

Proceedings of the
1st International Workshop on
Quantification and Classification
under Dataset Shift
(QCDS 2026)

Mirko Bunse, Pablo González, Eyke Hüllermeier,
Alejandro Moreo, and Fabrizio Sebastiani (eds.)

Preface

The 1st International Workshop on Quantification and Classification under Dataset Shift (QCDS 2026 – <https://qcds-2026.github.io/>) has been held in Napoli, IT, on September 7, 2026, as a satellite workshop of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2026).

Quantification and classification are two supervised learning tasks that, in real applications, are often hampered by changes in data distributions, i.e., by dataset shift. The two tasks differ in what trained models predict: while a classifier predicts the class label of each individual data point, a quantifier predicts the prevalence (i.e., relative frequency) of each class in a set of unlabelled data points. Both classifiers and quantifiers can suffer if dataset shift is at play, at least as long as they are not designed to handle the current type of shift robustly. Research has shown that a quantifier robust to dataset shift can facilitate robust classification, and a classifier robust to dataset shift can facilitate robust quantification.

QCDS 2026 aimed to engage the diverse expertise of the ECML/PKDD community; as dataset shift remains a fundamental challenge in real-world deployments, understanding the interplay between classification and quantification is more critical than ever. This workshop provided a collaborative forum for researchers and practitioners to bridge the gap between these two important fields, to share breakthroughs in machine learning methods robust to dataset shift, and to explore emerging applications.

QCDS 2026 was a follow-up of the Learning to Quantify (LQ) workshop series (see e.g., the 5th edition at <https://lq-2025.github.io/>), which concentrated exclusively on quantification. QCDS 2026 consisted of 8 oral presentations and 8 poster presentations, with each of these 16 submissions having gone through a rigorous peer-reviewing process by two or more reviewers each. The present volume contains the text of 15 of the presentations given at the workshop (for the other presentation the authors asked for their

paper not to be in the proceedings). We hope that the availability of the present volume will increase the interest in the subject of quantification and classification under dataset shift on the part of researchers and practitioners alike, and will contribute to making this topic better known to potential users of the corresponding technologies and to researchers interested in advancing the field.

Mirko Bunse
Pablo González
Eyke Hüllermeier
Alejandro Moreo
Fabrizio Sebastiani

Table Of Contents

Oral Presentations

<i>Zero-Shot Test-Time Canonicalization using Out-of-Distribution Scoring</i> Dominik Lindner, Johann Schmidt, Tom Siegl, Martin Becker, and Sebastian Stober	p. 1
<i>A Non-Aggregative Quantification Method for Graph Nodes</i> Alessio Micheli, Alejandro Moreo, Marco Podda, Fabrizio Sebastiani, and Domenico Tortorella	p. 17
<i>The Fragility of Value in Customer Churn: Predict-then-Optimize vs. Predict-and-Optimize Under Dataset Shift</i> Shimanto Rahman, Bram Janssens, and Matthias Bogaert	p. 33
<i>Dataset Shifts in Temporal Quantification</i> João Pedro Ortega, Luiz Fernando Luth Junior, Daniel Zonta Ojeda, André Maletzke, and Mirko Bunse	p. 49
<i>Uncertainty-Aware Classifier Accuracy Prediction under Prior Probability Shift</i> Lorenzo Volpi, Alejandro Moreo, and Fabrizio Sebastiani	p. 70
<i>Uncertainty-Aware Finite-Sample Quantification</i> Clemens Damke and Eyke Hüllermeier	p. 88
<i>Decision Tree Leaf Binning with Expectation Maximization for Quantification</i> David Kaftan, George Corliss, Anna Bershteyn, and Richard Povinelli	p. 104
<i>Out-of-Known-Safes Detection as a Benchmark for Runtime Monitors</i> Robbe De Groeve, Laurens Devos, and Mathias Verbeke	p. 120

Poster Presentations

<i>Assessing the Impact of Dataset Shift for Automatic Nephrops Norvegicus Burrows Detection and Classification Through Deep Learning</i> Oscar Papini, Marco Reggiannini, Enrico Cecapoli, Filippo Domenichetti, Lorenzo Zacchetti, Michela Martinelli, and Gabriele Pieri	p. 136
<i>Pre-Generation LLM Failure Prediction Under Distribution Shift</i> Yael Moros-Daval, Álvaro David Gómez Antón, Kexin Jiang-Chen, Cèsar Ferri, José Hernández-Orallo, and Fernando Martínez-Plumed	p. 148
<i>On Binary Quantification Calibration</i>	

Isabella Caroline Sachini Lorena, André Gustavo Maletzke, and Willian Zalewski.....	p. 164
<i>Drift Happens: An Empirical Study of Neural Architecture Robustness to Temporal Distribution Shift</i>	
Robin Holzinger and Riccardo Colletti.....	p. 183
<i>Beyond Single-Objective Accuracy: A Longitudinal Evaluation of Tabular Foundation Models under Dataset Shift</i>	
João Marcos Campos, Rafael Gomes, Diogo Chaves, Hugo Rocha, Mariangela Cherchiglia, Leonardo Rocha, Wagner Meira, and Marcos Gonçalves..	p. 222

The copyright (©) of all the papers in this volume is owned by the respective authors.

Organizers

- Mirko Bunse, University of Dortmund, DE
- Pablo González, University of Oviedo, ES
- Eyke Hüllermeier, Ludwig Maximilian University of Munich, DE
- Alejandro Moreo, Consiglio Nazionale delle Ricerche, IT
- Fabrizio Sebastiani, Consiglio Nazionale delle Ricerche, IT

Program Committee members

- Gustavo Batista, University of New South Wales, AU
- Clemens Damke, Ludwig Maximilian University of Munich, DE
- Juan José del Coz, University of Oviedo, ES
- Zahra Donyavi, University of New South Wales, AU
- Andrea Esuli, Consiglio Nazionale delle Ricerche, IT
- Cèsar Ferri, Universitat Politècnica de València, ES
- Peter Flach, University of Bristol, UK
- Devin Guillory, UC Berkeley, US
- Barbara Hammer, University of Bielefeld, DE
- Rafael Izbicki, Federal University of São Carlos, BR
- Mira Jürgens, Gent University, BE
- André Maletzke, Universidade Estadual do Oeste do Paraná, BR
- Olaya Pérez-Mon, University of Oviedo, ES
- Teodora Popordanovska, KU Leuven, BE
- Marco Saerens, Catholic University of Louvain, BE
- Tobias Schumacher, University of Mannheim, DE
- Dirk Tasche, North-West University, SA
- Lorenzo Volpi, Consiglio Nazionale delle Ricerche, IT
- Willem Waegeman, Ghent University, BE

Acknowledgments

PG’s work has been funded by Agencia de Ciencia, Competitividad Empresarial e Innovación Asturiana (Sekuens) within the program “Grants for research groups from organizations in the Principado de Asturias, 2024” (GRU-GIC-24-018). AM’s and FS’s work was partially supported by project “Future Artificial Intelligence Research” (FAIR), project “Quantification under Dataset Shift” (QuaDaSh), and project “Strengthening the Italian RI for Social Mining and Big Data Analytics” (SoBigData.it), all funded by the European Union under the NextGenerationEU funding scheme (CUP B53D22000980006, CUP B53D23026250001, CUP B53C22001760006, respectively). The organizers’ opinions do not necessarily reflect those of the European Commission.

Zero-Shot Test-Time Canonicalization using Out-of-Distribution Scoring

Dominik Lindner¹ (✉), Johann Schmidt¹, Tom Siegl², Martin Becker³, and
Sebastian Stober¹

¹ Artificial Intelligence Lab, Otto-von-Guericke University Magdeburg, Germany
{dominik.lindner, johann.schmidt, stober}@ovgu.de

² University of Rostock, Germany tom.siegl@uni-rostock.de

³ Becker Lab, University of Marburg, Germany martin.becker@uni-marburg.de

Abstract. Pretrained vision models often misclassify inputs that are rotated, scaled, or sheared, even though these affine transformations leave the object class unchanged. Robustness is usually restored either by building equivariance into the architecture or by retraining with augmentation, both of which require changing or retraining the model. Test-time canonicalization instead leaves the classifier untouched. It undoes the transformation of each input, mapping it to a canonical form near the training distribution before classification. Existing canonicalizers, however, rely on a narrow set of logit-based energy scores and bespoke search procedures, leaving the design space of scoring functions and optimizers unexplored. We reframe canonicalization as out-of-distribution (OOD) detection, which lets any OOD score serve as the energy minimized over transformations. Across benchmarks ranging from handwritten characters and sketches to natural images and 3D point clouds, we systematically evaluate around twenty OOD scores and nine search algorithms, finding that distance-based scores paired with random search and local refinement perform best overall. Because canonicalizing an already-aligned input can hurt accuracy, we add a gated mechanism that transforms an input only when its OOD score indicates this is needed, preserving most in-distribution accuracy while retaining the robustness gains on transformed inputs. Code is available at github.com/johschm/its.

Keywords: Canonicalization · Out-of-Distribution Detection · Robustness · Group Invariance.

1 Introduction

For a model to be reliable, its predictions should remain stable under class-preserving transformations of the input. A rotated, rescaled, or sheared image of an object still depicts the same object. Such transformations are common at deployment time, for instance in microscopy, aerial imagery, or sketch and document analysis, where the upright pose seen during training cannot be assumed.

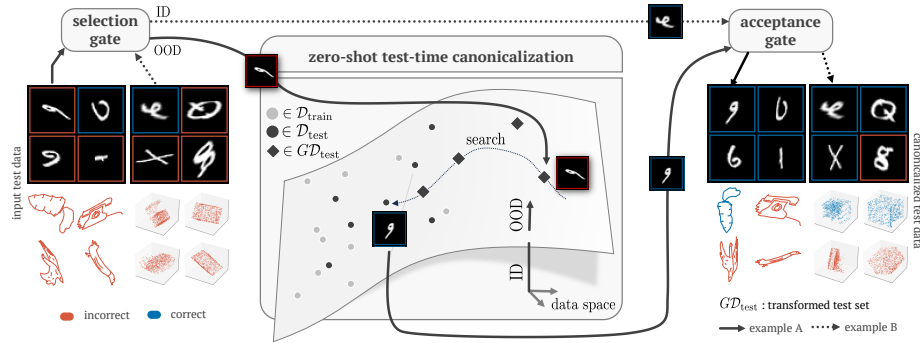


Fig. 1: At test time, in-distribution (ID) samples go directly to the downstream model, while out-of-distribution (OOD) samples enter a zero-shot canonicalization module that searches over transformations to minimize the OOD score. The result is passed on only if it lowers the score.

Yet many vision models are not robust to them [17,60], and retraining a deployed model for every such setting is often impractical. A strong form of robustness is invariance, where the predicted class remains unchanged under input transformations [26]. The most widely used method to increase robustness without altering the model architecture is *data augmentation*, which adds semantically redundant, transformed samples to the training set to encourage learning robust filters [24]. Invariance can be guaranteed by equivariant models [8] using group-pooling [7], which constrain the architecture through explicit inductive biases. This, unfortunately, is very constraining, as such an inductive bias has to be designed for the specific transformation and the model has to be trained from scratch. The downstream prediction can be made invariant without such constraints by averaging over all transformations [57]. Subsequent works reduce this to an average over a subspaces [42], but the computational cost remains prohibitive. This is closely related to Test-time Augmentation (TTA), where predictions are also averaged over transformed inputs. However, they assume a heuristic distribution over transformations and do not guarantee invariance [29].

An alternative approach to spatial robustness is *canonicalization*, which seeks to transform inputs into a canonical form by “undoing” group transformations prior to inference. Crucially, canonicalization does not constrain the downstream architecture, which can therefore be treated as fixed. Early canonicalizers learned to regress transformation parameters directly from data [25]. However, this can lead to fragile predictions even with modern backbones [46]. Later approaches formalized canonicalization as the selection of a canonical representation for an input sample by minimizing an equivariant scoring function over group transformations [28,40]. These methods typically rely on equivariant features or architectures for the canonicalizer [7] to ensure consistent scoring across transformed inputs. More recent work relaxes the requirement of equivariant canonicalizers by introducing training procedures for non-equivariant networks [41], but such approaches still require substantial training effort. Many learning-based canon-

icalizers further assume or benefit from training data that exhibits consistent alignment or a canonical pose distribution [40]. This assumption is often violated in real-world datasets, where object pose may vary significantly. Recent work addresses this by progressively re-aligning the training data during learning [48], but at the cost of additional training complexity. This raises a practical question: can canonicalization be obtained without training at all? Recent methods therefore formulate canonicalization as a test-time optimization problem, where a transformation is selected by minimizing a score defined directly on the downstream classifier’s outputs [47]. However, existing canonicalization methods are limited by their narrow choice of energy functions with costly optimization and a tendency to degrade in-distribution (ID) accuracy when inputs are already aligned. Existing approaches employ logit-based energies [47] or heuristic criteria [50] that favor transformations producing confident ID predictions.

We propose the hypothesis that OOD scores constitute suitable energy functions for canonicalization. The underlying assumption is that a canonical form lies in a high-density region of the training distribution and therefore attains a low OOD score, while transformed inputs move off this manifold and score higher. Minimizing an OOD score over the transformation group should thus recover a canonical form. This is a necessary rather than a sufficient condition. A low score favors transformations that appear typical, which need not coincide with the correct semantic pose, especially for classes with inherent pose ambiguity. We therefore treat it as an empirical hypothesis and test it across a broad range of scores and datasets. If it holds, the extensive body of OOD detection literature can be leveraged directly, broadening the design space for canonicalization. Moreover, the optimization strategy used to search over transformations is equally important. Prior work relies on bespoke search procedures tailored to specific scoring functions [47,51], including Lie-algebra-based methods for continuous transformation groups [50,29]. While effective, these approaches introduce additional complexity. Based on prior work, it remains unclear whether simpler global optimization strategies, such as simulated annealing or random search, which have shown benefits in other domains [45], can achieve competitive performance for common transformations such as affine transformations. Another issue of existing methods is the degradation of ID accuracy for aligned inputs [47], as incorrect or unnecessary transformations can distort features. This presents a practical challenge, since most real-world inputs are typically close to the training distribution and only a minority exhibit large transformations.

Contributions. Our main contributions to these problems are: (i) We systematically benchmark OOD scores as energy functions for canonicalization, addressing the limited exploration of energy functions in prior work. We find that the best-performing score varied across datasets. (ii) We show that prior search procedures can be replaced by simpler global optimizers (e.g., random search, simulated annealing) without sacrificing performance, with random search with local refinements working best overall. (iii) We introduce a lightweight OOD-based gating mechanism for canonicalization, resolving the accuracy trade-off in current frameworks by preserving ID and OOD performance.

2 Preliminaries

Let $(x, y) \sim \mathcal{D}$ denote input-label pairs, where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Many vision tasks admit class-preserving input transformations: a rotated image of a fish remains a fish. To study such transformations formally, we use group theory. The following introduces the key concepts needed (see Sec. A and [4,20] for more details). A *group* G is a set equipped with an associative binary operation, an identity element, and inverses for every element. We restrict to matrix Lie groups [20], so every $g \in G$ is a matrix, and the group operation is matrix multiplication. A *group action* of G on $\mathcal{X}$ is a map $(g, x) \mapsto g \cdot x$. A prominent example is the affine group $\text{Aff}_2(\mathbb{R})$, whose elements (any combination of rotation, scaling, shearing, and translation) act on pixel coordinates in homogeneous coordinates via 3×3 matrix multiplication. The *orbit* of x under G is $Gx = \{g \cdot x \mid g \in G\}$, which collects all transformed versions of x . A function $f: \mathcal{X} \rightarrow \mathcal{Y}$ is G -equivariant if $f(g \cdot x) = g \cdot f(x)$ for all $g \in G$ and $x \in \mathcal{X}$. In classification, the group acts trivially on labels, so equivariance reduces to G -invariance: $f(g \cdot x) = f(x) \quad \forall g \in G, x \in \mathcal{X}$. A G -invariant classifier assigns a single label to every orbit, which is precisely the robustness property we seek. We assume the classifier f to be not G -invariant, i.e. whose predictions can change under transformations in G , and we aim to render it G -invariant at test time. We do this by introducing a G -invariant function $h: \mathcal{X} \rightarrow \mathcal{X}$ such that $f(h(gx)) = f(h(x))$. We gain G -invariance without restricting f . This process is called *canonicalization* [28], where h maps input data to a specific $x \in Gx$ (the *canonical form*).

Definition 1 (Orbit Canonicalization [15]). *A canonicalization is a function $h: \mathcal{X} \rightarrow \mathcal{X}$ fulfilling the following invariance property for all $g \in G$ and $x \in \mathcal{X}$: $h(g \cdot x) = h(x)$ and $h(x) \in Gx$.*

As there is no trivial choice for $h(x)$, the most popular approaches [28,40] use optimization to find the canonical form of x . In this setting, each $g \cdot x \in Gx$ is assigned an energy and an optimizer seeks to find the point of least energy g^* , corresponding to the canonical form. The choice of energy function $E: \mathcal{X} \rightarrow \mathbb{R}$ is essential, as its minima should correspond to the canonical form of x and should be unique up to symmetries. The resulting $(g^*)^{-1} \cdot x$ is “undoing” the transformation applied to x , mapping it to its canonical form. Such that

$$h(x) = (g^*)^{-1} \cdot x \quad \text{with} \quad g^* = \arg \min_{g \in G} E(g^{-1} \cdot x). \quad (1)$$

For small groups this minimization can be done exactly, while continuous groups require methods to find approximate solutions. In principle E can be any function. What it does not guarantee is a unique minimizer per orbit: a generic OOD score may attain its minimum at several transformations that are not related by the stabilizer, in which case the canonical form is not well-defined and consistency across the orbit can fail [28]. We nonetheless forgo this formal uniqueness guarantee, since suboptimal solutions without such guarantees often perform well in practice [47], and we verify the resulting canonical forms empirically.

Table 1: Tested OOD scores, grouped by family (logit-based, activation-rectification, prototype-based, kNN-based; separated by rules). For each score we mark whether it requires stored training samples (TS), hyperparameter tuning (HP), or an auxiliary logit-based score (AS). The bottom row visualizes the relative inference-time increase (RT) of the full search relative to the Energy score, with bar height proportional to the increase. Bars below the axis indicate scores faster than Energy. Values are averaged across MNIST, EMNIST, TU Berlin and ModelNet10. Exact numbers are given in Tab. 1.

	<i>Energy</i>	<i>Entropy</i>	<i>GEN</i>	<i>MSP</i>	<i>MaxLogit</i>	<i>OpenMax</i>	<i>VIM</i>	<i>Laplace</i>	<i>ReAct</i>	<i>ASH</i>	<i>DICE</i>	<i>MD</i>	<i>RMD</i>	<i>SHE</i>	<i>Proto</i>	<i>PC-Proto</i>	<i>kNN</i>	<i>Trust Score</i>	<i>kNN Mix</i>	<i>PC-kNN Mix</i>
TS	-	-	-	-	-	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HP	-	-	✓	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AS	-	-	-	-	-	-	-	✓	✓	✓	✓	-	-	-	-	-	-	-	-	-
RT	0.0%	-3.0%	-1.5%	-3.5%	-1.7%	32.5%	1.8%	53.9%	-1.0%	2.2%	6.0%	13.3%	7.8%	2.2%	3.8%	9.3%	14.3%	52.8%	37.5%	43.5%

3 OOD-based Canonicalization

3.1 Searching for Canonical Forms

An OOD score $S : \mathcal{X} \rightarrow \mathbb{R}$ measures alignment with the training distribution, with lower values indicating that x is drawn from the distribution the classifier f was trained on. Our central insight is that, since canonical representatives reside near the data manifold [51], OOD scores are natural candidates for the energy E in Equation (1). This is a necessary but not sufficient condition. An OOD score rewards transformations that look typical, which may differ from the correct semantic pose for classes with inherent pose ambiguity. Whether it is sufficient in practice is the empirical question we study. We set $E(x) = S(x)$ and aim to find the lowest OOD score over all target transformations G . Throughout, we assume the transformation group G is specified in advance and contains the transformations the input may have undergone. This means that $g^* \cdot x$ is the most ID point. This connection opens the full landscape of OOD scores for canonical form estimation, whereas prior work was confined to very limited options [28,40,47,51]. Similarly, the choice of optimization strategy over G has received little attention, with prior work largely restricted to gradient-based optimization [40] or tree-searches [47] or Bayesian Optimization [51]. In this work, we study various choices for algorithms to find the minimum in Equation (1). We focus on heuristics rather than optimal solvers as for continuous groups this would be intractable. This leads to potentially suboptimal results (i.e., local minima g with $S((g^*)^{-1} \cdot x) \leq S(g^{-1} \cdot x)$). Even when S is convex in x , it is generally non-convex as a function of g , necessitating multi-start optimization rather than a single descent run [50]. These local minima can yield predicted canonical forms

that are themselves OOD [47,51]. This sub-optimality is particularly harmful when the found $g^{-1} \cdot x$ (local minima) is far from the true canonical form (global minima), degrading the classifier’s accuracy on standard inputs [47]. We address this with a gating mechanism for canonicalization functions h .

3.2 Gated Canonicalization

A practical failure mode of continuous canonicalization is the degradation of accuracy on data that is already in-distribution (which we show in the experiments). Unnecessary transformations perturb features and lower downstream confidence. To prevent this, we introduce *Gated Canonicalization*, which comprises two gates: a selection gate $\alpha_\tau(x)$ and an acceptance gate $\alpha(x)$. The former ensures that canonicalization is only applied if x is OOD. The latter ensures that the output of the canonicalization function reduces the OOD score, i.e., $S(\alpha_\tau(x)) \leq S(x)$. This is equivalent to always including the identity in the search, though stronger acceptance criteria may break this equivalence (see Sec. E.7). We define two canonicalization gates:

$$\text{selection gate } \alpha_\tau(x) = \begin{cases} (g^*)^{-1} \cdot x & \text{if } S(x) > \tau, \\ x & \text{otherwise,} \end{cases} \quad (2)$$

$$\text{acceptance gate } \alpha(x) = \begin{cases} (g^*)^{-1} \cdot x & \text{if } S((g^*)^{-1} \cdot x) < S(x), \\ x & \text{otherwise,} \end{cases} \quad (3)$$

where $\tau \in \mathbb{R}$ is a user-defined OOD threshold. In practice, τ is selected on a held-out validation set augmented with the same transformations, requiring no labels for the transformed test data. We study its effect in Sec. 5 and report validation-selected operating points in Sec. E.7. These mechanisms decouple the task of ID preservation from OOD robustness, treating the canonicalization process as a conditional corrective mapping. A visual overview is illustrated in Fig. 1.

4 OOD Scores and Search Methods

OOD Scoring Functions. We limit our focus to training-free OOD scores that require no architectural changes. Our central premise is that canonical representatives lie near the data manifold, so any reliable OOD score is a natural energy function for canonicalization. We evaluate four families of detectors. *Logit-based methods* operate on classifier outputs and require no stored features. We include Maximum Softmax Probability (MSP) [23], MaxLogit [22], Energy [36], Entropy, Generalized Entropy (GEN) [37], OpenMax [3], Virtual-logit Matching (VIM) [58], and the Laplace Approximation [9] with energy, entropy, and mutual information (MI) scores [33]. *Activation-rectification methods* modify intermediate features before computing a logit-based score, amplifying distributional differences. We include Rectified Activations (ReAct) [54], Activation Shaping (ASH) [12], and Directed Sparsification (DICE) [55]. *Prototype-based methods* measure

Table 2: Tested search algorithms. (*left*) For each method we mark whether it is a global search (GS), gradient-based (GB), and parallelizable (PL), and visualize the relative inference-time increase (RT) over Random Search as a bar (height proportional to the increase, averaged across MNIST, EMNIST, TU Berlin and ModelNet10). (*right*) Worst-case time complexity in Big-O.

	<i>RS</i>	<i>RS-LR</i>	<i>SA</i>	<i>GD</i>	<i>CD-S</i>	<i>CD</i>	<i>WCD</i>	<i>WCD-L</i>	<i>ITS</i>	Method	Complexity	
GS	✓	✓	✓	✓	—	—	—	—	—	RS	$\mathcal{O}(N)$	N : samples
GB	—	✓	—	✓	—	—	—	—	—	RS-LR	$\mathcal{O}(N + Lk)$	L : local pts
PL	✓	✓	✓	✓	—	—	—	—	—	SA	$\mathcal{O}(Rk)$	k : iterations
RT	0.0%	5.7%	34.8%	6.7%	11.2%	13.1%	19.3%	27.3%	24.76%	GD	$\mathcal{O}(Rk)$	R : runs/restarts
										WCD	$\mathcal{O}(M(d+W)c)$	M : samples/dim
										ITS	$\mathcal{O}(MHd)$	d : dims
												W : first-dim weight
												c : cycles
												H : hypotheses

distance to class representatives in feature space. We include Mahalanobis Distance (MD) [35], Relative Mahalanobis Distance (RMD) [43], Simplified Hopfield Energy (SHE) [61], and prototype distances using Euclidean, cosine, or a convex combination of both, applied either globally or per predicted class (PC-Proto). *kNN-based methods* score inputs by their average distance to the k nearest in-distribution embeddings [56]. We include variants using Euclidean and cosine distance, a convex mixture (kNN Mix), per-class versions (PC-kNN, PC-kNN Mix), and Trust Score [27]. These methods require storing training embeddings but no additional training. Tab. 1 summarizes the requirements of each detector. Further implementation details are in Sec. B.

Search Algorithms. We evaluate algorithms from global exploration to local exploitation. *Random Search (RS)* [52] samples the parameter space via Sobol sequences for uniform coverage. *RS with Local Refinement (RS-LR)* applies Adam-based gradient descent to the best candidate from global sampling. *Simulated Annealing (SA)* [31] uses neighborhood sampling [53] with multiple parallel runs to reduce latency. *Multi-start Gradient Descent (GD)*, as used in [50], runs Adam from several random initializations. As a non-global baseline inspired by ITS [47], we study *Coordinate Descent (CD)*, which sequentially optimizes each transformation parameter via uniform random sampling, iterating over multiple cycles. Variants include *CD-S* (one cycle), *Weighted CD (WCD)* (increased sampling density in the first dim.), and *WCD-L* (equally spaced samples). Tab. 2 summarizes properties and complexities with full details in Sec. C.

5 Experiments

We make our source code publicly available for the reproducibility. We refer the reader to Sec. D or our source code for more details on the used hyperparameters.

Finding. Across all benchmarks, random search with local refinement (**RS-LR**) combined with a distance-based OOD score performs best (Secs. 5.2 and 5.3). The single best score is dataset dependent but always distance-based: kNN for MNIST, PC-kNN for EMNIST, kNN Mix for ModelNet10, and PC-Proto for TU Berlin. On the validation set, the exact same detectors achieve the top scores across all datasets except ModelNet10, where **PC-kNN Mix** performs best. We refer to using the best configuration (RS-LR plus score) on the validation set as **OODC** (OOD-based Canonicalization) and use it throughout unless stated otherwise.

Benchmarks, Models, and Metrics. We evaluate on a deliberately diverse set of benchmarks: MNIST and EMNIST Balanced [6] (handwritten characters), TU Berlin [16] (single-line sketches), ModelNet10 [62] (3D point clouds), and SI-Score (rotation) [60], a robustness benchmark of ImageNet [11] foreground objects transformed onto varying backgrounds. For MNIST, EMNIST, and TU Berlin we construct transformed test sets by applying rotation, shear, and scaling. For ModelNet10 only rotation is applied, as scale and translation are normalized a priori. Crucially, the search draws candidate transformations from the same family and parameter ranges used to generate these transformed sets. This aligns the search space with the true corruption and likely benefits informed and random search differently. Under natural OOD, where the transformation family is unknown or only approximate, the advantage of any particular search strategy may shrink. Full augmentation ranges, downstream architectures, and training details are given in Sec. D. We report average top-1 test accuracy (mean over multiple runs with $1.96\times$ standard error to indicate statistical significance). To measure how close the search comes to the canonical form (the energetic minimum), we avoid the degenerate distance between predicted and ground-truth affine matrices and instead use a score-based distance,

$$\text{Scoring Distance (Err.): } \frac{S((g^*)^{-1}gx) - S((g_{\min}^*)^{-1}gx)}{S((g_{\max}^*)^{-1}gx) - S((g_{\min}^*)^{-1}gx)} \in [0, 1], \quad (4)$$

where g^* is the found solution, gx the transformed input, and $g_{\min}^*$, $g_{\max}^*$ the transformations with minimal and maximal score across all runs and methods.

5.1 How competitive is our OOD-based canonicalization?

We compared our OOD-based canonicalization framework on various benchmarks (including ID and OOD test sets) against multiple baselines. Results for MNIST, EMNIST, TU Berlin, and ModelNet10 are shown in Tab. 3. All retraining-based models were trained on augmented data. The STN [25] used a smaller version of the main model, with one-quarter of the channels. Test-time Augmentation averaged predictions over eight transformation samples, always including the identity. Equivariant architectures perform best among approaches requiring retraining, while the vanilla model leads among training-free variants but lacks robustness. OODC achieves the highest accuracy on transformed test

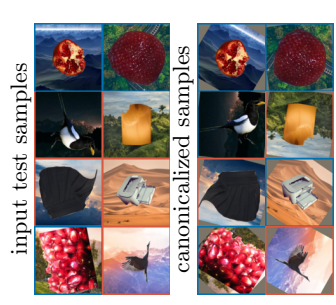
Table 3: Accuracy (%) on the original (top row, $\times$) and transformed (bottom row, $\checkmark$) test sets, indicated in column **Trf.** Methods are grouped into those that require retraining and those that do not. Retraining baselines: augmented training (**Aug.**), an equivariant SteerableCNN [8] or Vector Neuron model [10] (**Equi.**), a spatial transformer trained on augmented data (**STN**), and canonicalization with a learned energy model (**Learned**, Sec. E.3). Training-free methods: the vanilla classifier (**Van.**), Test-time Augmentation (**TTA**) [34], vanilla **ITS** [47], ITS with its logit energy (**ITS^a**), **ITS^b** with the same OOD score as OODC, and **OODC**. The best result within each group is shown in bold per row.

Dataset	Trf.	Requires Retraining				No Retraining					
		Aug.	Equi.	STN	Learned	Van.	TTA	ITS	ITS ^a	ITS ^b	OODC
MNIST	$\times$	99.2	99.3	99.3	98.6 $\pm$ 0.1	99.6	83.0	76.3	93.8	98.1	95.1 $\pm$ 0.1
	$\checkmark$	98.7	99.2	99.2	98.1 $\pm$ 0.1	37.7	49.8	60.4	75.5	85.0	89.8$\pm$0.3
EMNIST	$\times$	83.2	84.1	83.6	76.7 $\pm$ 0.2	88.4	44.2	54.1	71.2	78.0	64.4 $\pm$ 0.2
	$\checkmark$	82.1	83.9	82.8	75.6 $\pm$ 0.1	17.1	17.2	33.4	46.6	51.4	56.1$\pm$0.4
TU Berlin	$\times$	42.4	—	30.5	35.0 $\pm$ 0.8	51.6	18.9	36.6	39.7	40.5	34.0 $\pm$ 0.4
	$\checkmark$	40.7	—	28.5	31.2 $\pm$ 1.4	8.2	8.2	19.7	22.0	25.7	28.1$\pm$0.3
ModelNet10	$\times$	80.1	87.8	82.5	70.1 $\pm$ 0.5	91.2	47.7	33.6	32.4	86.7	60.6 $\pm$ 0.7
	$\checkmark$	83.9	87.8	82.9	68.2 $\pm$ 1.6	20.5	28.3	27.1	27.6	54.0	59.6$\pm$0.9

data among all training-free methods. On untransformed data, however, its accuracy falls well below the vanilla model. This is inherent to always-on canonicalization: every input is transformed, so already-aligned inputs are needlessly perturbed and their features distorted. The same effect appears for the STN, which also transforms unconditionally, indicating that the drop is the cost of unconditional robustness rather than a weakness of the OOD score. In Sec. 5.4 we show that gating restores most of the in-distribution accuracy while retaining the gains on transformed inputs, converting this apparent trade-off into a controllable one. Next, we evaluate our canonicalization framework on SI-Score (Rotation) [60] as a large-scale robustness benchmark using a ResNet-50 [21] and a ViT-B16 [13] backbone. For optimization, we use 17 equally spaced samples as we only consider rotation. Due to computational cost, we restricted OODC to PC-kNN here and additionally report the energy of the logits, without exploring other scores. Hyperparameters were optimized on rotated ImageNet[11] images (distinct from data used for embeddings). Feature-based OOD scores significantly outperform existing approaches, including ITS [47] (using their logit-based energy detector and iterative search), foreground fine-tuning for ViT [5], and Test Time Augmentation (TTA) over 16 equally spaced rotations (Fig. 2).

5.2 What Search Algorithm to choose for Canonicalization?

As testing all classifiers and combinations is computationally not feasible, we opted to evaluate the search on several selected score functions first. As scores



The figure shows two 4x4 grids of images. The left grid, labeled 'input test samples', shows various objects (like a red pepper, a yellow pepper, a black pepper, a white pepper, a red pepper, a yellow pepper, a black pepper, a white pepper) in different orientations. The right grid, labeled 'canonicalized samples', shows the same objects in a canonicalized orientation. Below the grids is a table of test accuracies for various methods.

	Method	ViT-B16	ResNet50
Baselines	Vanilla Classifier	38.8	48.0
	RS Augm. [47]	41.0	—
	Chefer et al. [5]	44.8	—
	Chefer et al. + AS [5]	46.2	—
	ITS [47]	49.0	—
	TTA [34]	49.5	54.9
Score	Energy of Logits	51.4 ± 0.2	48.3 ± 0.2
	PC-kNN	57.9 ± 0.1	60.7 ± 0.1

Fig. 2: Performance on the SI-Score (rotation) with canonicalized images on the left (blue correct, red incorrect classification), and test accuracies on the right.

Table 4: Comparison of search strategies on different datasets with 60 evaluations, showing random search with local refinement performing best overall.

Method	MNIST		EMNIST		ModelNet10		TU Berlin	
	Acc ↑	Err ↓	Acc ↑	Err ↓	Acc ↑	Err ↓	Acc ↑	Err ↓
CD	52.2 ± .2	0.42	80.2 ± .1	0.33	37.5 ± .4	0.52	24.4 ± .3	0.37
CD-S	52.2 ± .1	0.42	80.1 ± .2	0.33	36.0 ± .3	0.59	24.5 ± .3	0.37
GD	54.0 ± .2	0.57	85.6 ± .2	0.42	39.6 ± .5	0.55	—	—
ITS	55.7 ± .1	0.44	83.6 ± .1	0.34	39.2 ± .1	0.59	25.3 ± .1	0.34
SA	56.3 ± .2	0.56	85.0 ± .3	0.47	40.2 ± .4	0.54	21.3 ± .3	0.47
RS	57.4 ± .2	0.56	87.0 ± .2	0.46	40.3 ± .4	0.53	21.1 ± .3	0.47
RS-LR	59.9 ± .2	0.36	88.0 ± .2	0.27	41.9 ± .4	0.44	—	—
WCD	56.0 ± .2	0.32	82.6 ± .2	0.24	38.4 ± .5	0.52	26.1 ± .2	0.36
WCD-L	57.0 ± .1	0.30	83.0 ± .1	0.22	40.2 ± .4	0.48	28.7 ± .5	0.69

for this experiment, we use the negative energy of the logits [36], distance to the nearest embedding using cosine distance at the last layer (PC-kNN). In addition, we train a classifier to distinguish between data points that the downstream model predicts correctly and those it predicts incorrectly (called a trained energy model). The goal is to have a search algorithm that works with a wide range of possible scores, so we use the average accuracy across these scores as the metric. A main limiting factor of the searches is the computation time which is mainly decided by the number of model calls the methods make. For a fair comparison, the first experiment uses a fixed budget of 60 sample evaluations, chosen due to its divisibility. A forward pass counts as one evaluation, while a forward and backward pass counts as two to reflect the higher cost. The BiLSTM model requires unrolling, which is more expensive, so we do not apply gradient-based methods on the TU Berlin dataset. The hyperparameters for each method and dataset are optimized for 30 iterations, using the average accuracy across all scores, while ensuring that the computational budget is not exceeded. As shown in Tab. 4, the relative distance varied by at most 0.003 ($1.96 \times$ standard error),

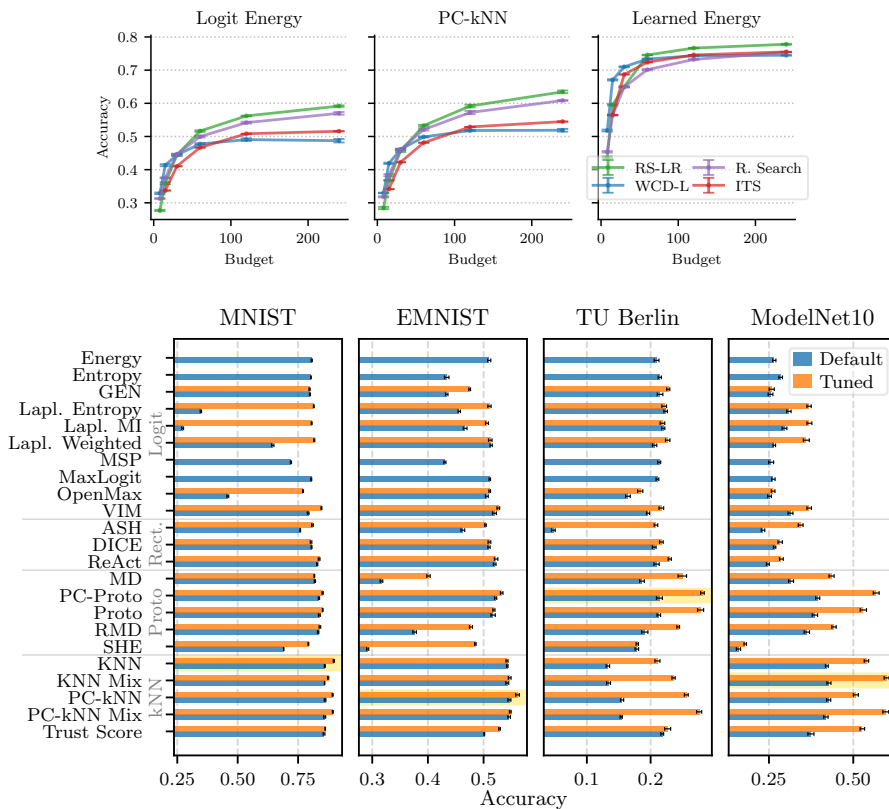


Fig. 4: Comparison of unsupervised OOD scores (best test score highlighted).

so it is omitted for readability. We found random search outperforms most other search algorithms. Coordinate descent-based methods find solutions closer to the minimum than random search, though often with lower accuracy, except on TU Berlin. Using local refinement can boost the accuracy slightly and find results closer to the minima, but typically not as close as WCD-L. As accuracy is typically prioritized, this makes it the preferable method within this budget. Next, we investigate if this holds for different budgets, re-optimizing hyperparameters for each. The results for previously well-performing methods on EMNIST are shown in Fig. 3 (cont. in Sec. E.5). On most datasets, RS-LR performs well with budgets above 30, though coordinate descent-based methods may outperform it at lower budgets. We found TU Berlin to be an exception where ITS works better. Overall, we recommend RS-LR for its simplicity, dimension-independent scaling, and strong performance at budgets where canonicalization is reliable.

5.3 What OOD Scoring Function to choose for Canonicalization?

As OOD scores exhibit varying performance depending on the dataset [59], we compare several detectors to examine whether this holds for canonicalization

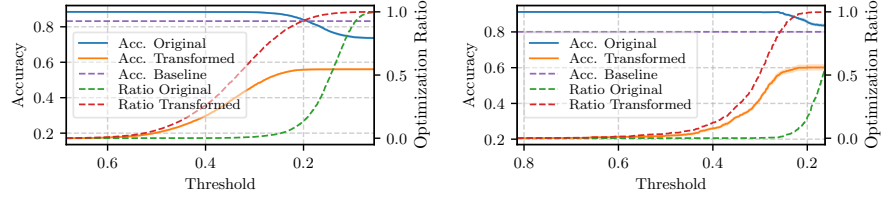


Fig. 5: Trade-off between accuracy on original and transformed data when varying the OOD threshold for EMNIST (left) and ModelNet10 (right), compared against the augmented model as the baseline.

and whether a well-chosen score can improve results. As the search strategy we use RS-LR as it worked well in previous experiments. Hyperparameters of the detectors were optimized in a two-stage process (see Sec. D.3). The results are shown in Fig. 4. Distance-based methods like kNN and prototype-based methods outperform logit-based methods, with kNN-based methods performing best overall. Laplace approximations and VIM also surpass logit-based methods but are beaten by distance-based approaches. Activation reshaping methods show similar performance to logit-based scores. In Sec. E we show example transformations by our method compared to our core baseline ITS. A concern with logit- and feature-based OOD detection is that information loss in deeper layers may hinder canonicalization. In Sec. E.3, we show that training a learned score function on later-layer features reduces accuracy, suggesting that feature-based methods may be limited compared to methods trained on the input data.

5.4 How does Gating trade ID vs OOD performance?

A working OOD detection setup should not degrade a model’s performance on ID data [59]. By thresholding the OOD score and only canonicalizing samples deemed OOD, we can control the trade-off between performance on transformed and untransformed data, accepting search results only when the score improves. Due to a lack of datasets with realistic mixtures of transformations, we only report the threshold’s effect on the standard and transformed test sets. Fig. 5 shows this trade-off for EMNIST and ModelNet10 using the best unsupervised OOD score per dataset. This shows that most performance on untransformed data can be retained while significantly improving it on transformed data. It remains below always-canonicalizing since some transformed inputs are skipped, which reduces computational costs. Other datasets show similar trends (Sec. E.7).

6 Conclusions

Zero-shot test-time canonicalization can and should take advantage of the vast space of existing out-of-distribution (OOD) scores and search heuristics. Selecting an appropriate OOD score significantly improves performance compared to

previous work [47]. We found that random search with local refinement coupled with distance-based OOD scores performs best overall, while the single best score is dataset dependent. Furthermore, existing canonicalization methods tend to degrade in-distribution (ID) performance as all inputs are always transformed. We introduce a gating mechanism that ensures that canonicalization is only executed when needed, significantly mitigating the negative effect on the ID performance. The framework requires no retraining and treats the classifier as a black box, so it can be attached to any pretrained model to increase robustness, with the inference-time search cost being the main practical consideration.

Limitations and Future Work. The 60-sample inference budget may limit real-time use. More sample-efficient optimizers such as evolutionary methods or Bayesian optimization are a natural next step. Deep features alone can be insufficient when spatial information is critical, so combining OOD scores with foundation-model energies [51] is promising. When transformations are known a priori and retraining is feasible, augmentation-based training can still be preferable, though test-time canonicalization offers a path toward closing this gap without such assumptions. Canonicalization could also be applied locally, to parts or individual objects rather than the whole frame, via part-sampling methods [49].

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

7 Acknowledgments

Dominik Lindner was funded by the IB (ZS/2026/08/204393). Johann Schmidt was funded by the ZIM (KK5540302MS4). Tom Siegl was cofunded by the BMFTR (01IS22077, 01ED2507) and DFG SFB 1713.

References

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: International Conference on Knowledge Discovery & Data Mining (ACM SIGKDD) (2019)
2. Amos, B., Xu, L., Kolter, J.Z.: Input convex neural networks. In: International Conference on Machine Learning (ICML) (2017)
3. Bendale, A., Boulton, T.E.: Towards Open Set Deep Networks. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
4. Bronstein, M.M., Bruna, J., Cohen, T., Velicković, P.: Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. 2104.13478, arXiv (2021)
5. Chefer, H., Schwartz, I., Wolf, L.: Optimizing relevance maps of vision transformers improves robustness. In: Conference on Neural Information Processing Systems (NeurIPS) (2022)
6. Cohen, G., Afshar, S., Tapson, J., André van Schaik: EMNIST: Extending MNIST to handwritten letters. In: International Joint Conference on Neural Networks (IJCNN) (2017)

7. Cohen, T., Welling, M.: Group equivariant convolutional networks. In: International Conference on Machine Learning (ICML) (2016)
8. Cohen, T.S., Welling, M.: Steerable CNNs. In: International Conference on Learning Representations (ICLR) (2017)
9. Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., Hennig, P.: Laplace Redux - Effortless Bayesian Deep Learning. In: Conference on Neural Information Processing Systems (NeurIPS) (2021)
10. Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., Guibas, L.J.: Vector neurons: A general framework for $so(3)$ -equivariant networks. In: International Conference on Computer Vision (ICCV) (2021)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2009)
12. Djurisic, A., Bozanic, N., Ashok, A., Liu, R.: Extremely Simple Activation Shaping for Out-of-Distribution Detection. In: International Conference on Learning Representations (ICLR) (2023)
13. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. International Conference on Learning Representations (ICLR) (2021)
14. Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.E., Lomeli, M., Hosseini, L., Jégou, H.: The faiss library. *IEEE Transactions on Big Data* (2025)
15. Dym, N., Lawrence, H., Siegel, J.W.: Equivariant frames and the impossibility of continuous canonicalization. In: International Conference on Machine Learning (ICML) (2024)
16. Eitz, M., Hays, J., Alexa, M.: How do humans sketch objects? *ACM Transactions on Graphics* (2012)
17. Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., Madry, A.: Exploring the landscape of spatial robustness. In: International Conference on Machine Learning (ICML) (2019)
18. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)
19. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: International Conference on Machine Learning (ICML) (2016)
20. Hall, B.C.: Lie Groups, Lie Algebras, and Representations: An Elementary Introduction. Springer International Publishing (2015)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
22. Hendrycks, D., Basart, S., Mazeika, M., Zou, A., Kwon, J., Mostajabi, M., Steinhardt, J., Song, D.: Scaling Out-of-Distribution Detection for Real-World Settings. In: International Conference on Machine Learning (ICML) (2022)
23. Hendrycks, D., Gimpel, K.: A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In: International Conference on Learning Representations (ICLR) (2017)
24. Huang, T., Sankar, C., Amini, P., Kottur, S., Geramifard, A., Razaviyayn, M., Beirami, A.: Dair: Data augmented invariant regularization. *ArXiv* (2021)
25. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: International Conference on Neural Information Processing Systems (NeurIPS) (2015)

26. Jang, H., Zaidi, S.S.A., Boix, X., Prasad, N., Gilad-Gutnick, S., Ben-Ami, S., Sinha, P.: Robustness to transformations across categories: Is robustness driven by invariant neural representations? *Neural Computation* (2023)
27. Jiang, H., Kim, B., Guan, M., Gupta, M.: To trust or not to trust a classifier. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2018)
28. Kaba, S.O., Mondal, A.K., Zhang, Y., Bengio, Y., Ravanbakhsh, S.: Equivariance with learned canonicalization functions. In: *International Conference on Machine Learning (ICML)* (2023)
29. Kim, J., Kaba, S.O., Park, J., Hong, S., Ravanbakhsh, S.: Inverting Data Transformations via Diffusion Sampling. *ArXiv* (2026)
30. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: *International Conference on Learning Representations (ICLR)* (2015)
31. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. *Science* (1983)
32. Kristiadi, A., Hein, M., Hennig, P.: Being bayesian, even just a bit, fixes overconfidence in ReLU networks. In: *International Conference on Machine Learning (ICML)* (2020)
33. Lafage, A., Laurent, O., Gabetni, F., Franchi, G.: Torch-uncertainty: A deep learning framework for uncertainty quantification. In: *NeurIPS Datasets and Benchmarks Track* (2025)
34. Laptev, D., Savinov, N., Buhmann, J.M., Pollefeys, M.: Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)
35. Lee, K., Lee, K., Lee, H., Shin, J.: A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2018)
36. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based out-of-distribution detection. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2020)
37. Liu, X., Lochman, Y., Zach, C.: GEN: Pushing the Limits of Softmax-Based Out-of-Distribution Detection. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2023)
38. Ma, G., Wang, Y., Lim, D., Jegelka, S., Wang, Y.: A Canonicalization Perspective on Invariant and Equivariant Learning. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2024)
39. MacKay, D.J.C.: Bayesian Interpolation. *Neural Computation* (1992)
40. Mondal, A.K., Panigrahi, S.S., Kaba, S.O., Rajeswar, S., Ravanbakhsh, S.: Equivariant adaptation of large pretrained models. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2023)
41. Panigrahi, S.S., Mondal, A.K.: Improved canonicalization for model agnostic equivariance. In: *CVPR Workshop on Equivariant Vision: From Theory to Practice* (2024)
42. Puny, O., Atzmon, M., Smith, E.J., Misra, I., Grover, A., Ben-Hamu, H., Lipman, Y.: Frame averaging for invariant and equivariant network design. In: *International Conference on Learning Representations (ICLR)* (2022)
43. Ren, J., Fort, S., Liu, J., Roy, A.G., Padhy, S., Lakshminarayanan, B.: A Simple Fix to Mahalanobis Distance for Improving Near-OOD Detection. *ArXiv* (2021)
44. Rudin, W.: *Principles of Mathematical Analysis*. McGraw-Hill (1976)
45. Schmidt, J., Köhler, B., Borstell, H.: Reviving simulated annealing lifting its degeneracies for real-time job scheduling. *Hawaii International Conference on System Sciences (HICSS)* (2024)

46. Schmidt, J., Siegl, T., Becker, M., Stober, S.: Geometrically constrained and token-based probabilistic spatial transformers. *ArXiv* (2026)
47. Schmidt, J., Stober, S.: Tilt your Head: Activating the Hidden Spatial-Invariance of Classifiers. In: *International Conference on Machine Learning (ICML)* (2024)
48. Schmidt, J., Stober, S.: Robust canonicalization through bootstrapped data re-alignment. *EurIPS: UPLB Workshop* (2025)
49. Schmidt, J., Stober, S., Denzler, J., Bodesheim, P.: Saccadic vision for fine-grained visual classification. *ArXiv* (2025)
50. Shumaylov, Z., Zaika, P., Rowbottom, J., Sherry, F., Weber, M., Schönlieb, C.B.: Lie algebra canonicalization: Equivariant neural operators under arbitrary lie groups. In: *International Conference on Learning Representations (ICLR)* (2025)
51. Singhal, U., Feng, R., Yu, S.X., Prakash, A.: Test-Time Canonicalization by Foundation Models for Robust Perception. In: *International Conference on Machine Learning (ICML)* (2025)
52. Sobol', I.: On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics* (1967)
53. Spall, J.C.: *Introduction to Stochastic Search and Optimization*. John Wiley & Sons, Inc. (2003)
54. Sun, Y., Guo, C., Li, Y.: ReAct: Out-of-distribution detection with rectified activations. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2021)
55. Sun, Y., Li, Y.: DICE: Leveraging sparsification for out-of-distribution detection. In: *European Conference on Computer Vision (ECCV)* (2022)
56. Sun, Y., Ming, Y., Zhu, X., Li, Y.: Out-of-distribution detection with deep nearest neighbors. In: *International Conference on Machine Learning (ICML)* (2022)
57. Tahmasebi, B., Jegelka, S.: Generalization bounds for canonicalization: A comparative study with group averaging. In: *International Conference on Learning Representations (ICLR)* (2025)
58. Wang, H., Li, Z., Feng, L., Zhang, W.: ViM: Out-Of-Distribution with Virtual-logit Matching. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2022)
59. Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., PENG, WENXUAN., Wang, H., Chen, G., Li, B., Sun, Y., Du, X., Zhou, K., Zhang, W., Hendrycks, D., Li, Y., Liu, Z.: OpenOOD: Benchmarking generalized out-of-distribution detection. In: *Conference on Neural Information Processing Systems (NeurIPS)* (2022)
60. Yung, J., Romijnders, R., Kolesnikov, A., Beyer, L., Djolonga, J., Houlsby, N., Gelly, S., Lucic, M., Zhai, X.: SI-Score: An image dataset for fine-grained analysis of robustness to object location, rotation and size. In: *ICML Shift Happens Workshop* (2022)
61. Zhang, J., Fu, Q., Chen, X., Du, L., Li, Z., Wang, G., Liu, x., Han, S., Zhang, D.: Out-of-distribution detection based on in-distribution data patterns memorization with modern hopfield energy. In: *International Conference on Learning Representations (ICLR)* (2023)
62. Zhirong Wu, Song, S., Khosla, A., Fisher Yu, Linguang Zhang, Xiaoou Tang, Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2015)
63. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the Continuity of Rotation Representations in Neural Networks. *ArXiv* (2020)

A Non-Aggregative Quantification Method for Graph Nodes

Alessio Micheli¹, Alejandro Moreo², Marco Podda¹, Fabrizio Sebastiani²,
and Domenico Tortorella¹

¹Dipartimento di Informatica
Università di Pisa
56127 Pisa, Italy

`{firstname.lastname}@unipi.it`

²Istituto di Scienza e Tecnologie dell’Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
`{firstname.lastname}@cnr.it`

Abstract. Node quantification is the task of estimating the prevalence of different node labels in arbitrary subsets of nodes extracted from graphs, under prior probability shift. Existing approaches to node quantification typically follow the aggregative paradigm, i.e., they train an intermediate node classifier, aggregate its predictions, and then correct the resulting prevalence estimates using various adjustment strategies. In this work we propose a non-aggregative method that estimates node prevalence values directly from sets of node representations, thereby bypassing the need for an intermediate node classifier. Notably, these aggregated representations are obtained from an untrained Graph Echo State Network (GESN), which ensures efficiency especially during inference. Our approach achieves performance comparable to aggregative methods while preserving the distinctive advantage of remaining applicable when inferring individual node labels is undesirable due to privacy concerns.

Keywords: Quantification, Supervised prevalence estimation, Quantification of linked data, Graph learning, Dataset shift

1 Introduction

Quantification [4, 12] is the supervised learning task of estimating the class prevalence values in a sample of unlabelled datapoints affected by dataset shift. This problem was first explicitly formulated (although without an explicit mention of dataset shift) about 20 years ago [7], and for the first 10 years it encountered limited interest on the part of the scientific community. However, this interest has progressively increased since then, also due to an increased awareness of the impact of dataset shift [23, 29] on the performance of machine-learned models.

While most quantification literature addresses “flat” data, a few works [2, 20, 22, 32] have tackled the case in which the (training and unlabelled) datapoints are

nodes in a graph. As is well-known from classification (quantification’s “sibling” task), the fact that datapoints are connected to each other is an important source of information, and classifiers capable of exploiting this information routinely outperform classifiers that treat the nodes as flat data [31]; the same is true of quantification, as has been shown in the above-cited papers.

When learning (either a classifier or a quantifier) in a networked data context, we usually assume that the training data and the unlabelled data are nodes of the *same* graph, and that the structure of the entire graph (i.e., which nodes are connected to which other nodes) *and* the covariates of the unlabelled nodes are known at training time; in other words, one difference between learning from flat data and learning from networked data is that this latter is usually tackled as a *transductive* task [11].

In this paper we contribute to the literature on quantification for graph nodes by presenting a new, *non-aggregative* (and transductive) method for estimating class prevalence values in subsets of unlabelled graph nodes. A non-aggregative quantification method is one that, unlike most quantification methods, does *not* rely on an intermediate classification step in which all the unlabelled datapoints are classified (either by a “hard” classifier that returns class labels, or by a “soft” classifier that returns numerical classification scores or, more specifically, posterior probabilities). Non-aggregative quantification methods are of independent interest, because (a) they are the full realization of the well-known *Vapnik’s principle* ([33] – see also [4, §4.4]), according to which we should solve our task (here: quantification) directly and not by first solving a more general task (here: classification), and (b) in a number of contexts (especially ones where privacy is a concern) drawing inferences at the aggregate level might be desirable but drawing inferences at the individual level might not [5].

An example application of our method is the prediction of election results [1], since in this application (a) we are interested in predictions at the aggregate level (in the form of prevalence values attributed to specific candidates or parties), and (b) we may see the individual datapoints (i.e., the voters) as linked by relations (e.g., kinship, co-work, relatedness in a social network). In this application, while we are interested in predictions at the aggregate level, we typically want to rule out predictions at the individual level; this indicates that, in this context, non-aggregative quantification methods are preferable to aggregative ones. Similar scenarios arise when sensitive data are involved; for example, in estimating the geographical incidence of cancer,

we are often interested in prevalence rates at the level of regions or communities while explicitly avoiding predictions about individual patients. Considering that individual labels should not be made available, non-aggregative quantification methods are particularly appealing.

While our method is natively multiclass, we here limit our analysis to the binary case. We should also mention that, while our method is not specific to any type of dataset shift, in our experiments we test it in scenarios characterized by *prior probability shift* (PPS – an important type of dataset shift), by using, both at training and at testing time, data where PPS has been artificially injected.

Our experiments show that our approach achieves performance comparable to aggregative methods, while preserving the distinctive advantage of remaining applicable when inferring individual node labels is undesirable.

The rest of the paper is organized as follows. Section 2 surveys related work. Section 3 briefly introduces notation and background, while Section 4 presents our new method for quantification of graph nodes. In Section 5 we explain our experimental setting, while Section 6 discusses the experimental results we have obtained. Section 7 concludes, pointing at avenues for future research.

2 Related work

Quantification for linked data. While the literature on quantification is now substantial (see [4] for a fairly recent overview), the papers that have addressed quantification for linked data are just a handful.

The earliest such work is [32], where the authors propose two methods. The first (dubbed CBQ, for “classification-based quantification”)¹ consists of individually classifying each unlabelled node of the graph via a relational classifier (i.e., a classifier capable of exploiting both the content of a node and its surrounding link structure) and then correcting the resulting prevalence estimates via the ACC (non-relational) quantification method. The second method (dubbed LBQ, for “link-based quantification”) consists of estimating the fraction of nodes k hops away from an anchor node v_i as a mixture of class-conditional such fractions; since all these fractions are either available or can be estimated from the training data, the parameters of the mixture (i.e., the class prevalences) can be estimated via standard algebra, after which the final prevalence estimate is computed as the median across the estimates obtained for each anchor node v_i . However, while the CBQ method can be experimentally compared with the method proposed in this paper, the LBQ method cannot, since it allows estimating class prevalence in the entire graph only, and not in random subgraphs, as our method (and all the other methods discussed in this section) instead allows.

This work was followed by [22], which proposes two aggregative methods, “Community Discovery for Quantification” (CDQ) and “Ego Networks for Quantification” (ENQ). Both methods first assign classes to unlabelled nodes using the network structure (CDQ via community detection, and ENQ via ego networks), and then correct the resulting estimates. To this end, CDQ uses frequency or density within communities, while ENQ uses the majority class in a node’s ego network, with isolated nodes defaulting to the training distribution. Both methods ignore node features and assume homophily.

The other two methods we survey are radically more recent. Micheli et al. [20] introduces XNQ, a method that combines unsupervised node embeddings computed by randomized recursive graph neural networks with Saerens et al. [30]’s (non-relational) quantification method, and show that, under prior probability

¹ In our previous paper [20], CBQ was erroneously called “wvRN”, and was used as a baseline under this name.

shift, XNQ largely outperforms Tang et al. [32]’s CBQ and Milli et al. [22]’s CDQ and ENQ. Damke and Hüllermeier [2, 3] propose a quantification method for graph nodes that explicitly addresses a type of shift different from prior probability shift, which the authors call *structural covariate shift*, defined as the situation in which the training data come from a different region of the graph than the test data. The method that they propose is based on “structural importance sampling”; while [2] adapts to structural covariate shift the ACC quantification method [8], [3] does the same with the (better performing) KDEy method [24].

Note that [22, 32], like other early quantification works, do not mention dataset shift, and do not state which type of dataset shift they address. The analysis of dataset shift is instead central to [2, 3, 20].

Non-aggregative quantification. In contrast to aggregative quantifiers, non-aggregative quantifiers do not rely on a surrogate classifier issuing label predictions for the test instances as an intermediate step towards class prevalence estimation. Representative examples of this family include ReadMe [14], ReadMe2 [15], HDx [13], and EDx [16]. Beyond being non-aggregative, these methods share a common trait, namely, they belong to the family of distribution-matching methods. That is, they estimate class prevalences by solving an optimization problem in which the prevalence values are adjusted so that a mixture of class-conditional distributions best matches the distribution observed in the test data [6].

Different methods adopt different representations of the underlying distributions and different matching criteria. For instance, ReadMe and EDx operate directly on the original covariates, ReadMe2 relies on document embeddings, and HDx models relies on per-feature histograms. Likewise, ReadMe and ReadMe2 formulate the matching problem as a least-squares regression task, HDx relies on the Hellinger Distance, and EDx minimizes the expected Energy Distance between distributions.

Note that, although a classifier is not trained as part of the quantification process, all methods described above still require instance-level labels in the training set. This approach is appealing because, in the spirit of Vapnik’s principle (see the Introduction), the available information is used directly to solve the target problem, without resorting to a more general task (namely, classification). Despite this appealing property, recent empirical evaluations have consistently shown that non-aggregative quantifiers tend to underperform aggregative ones. Nevertheless, the non-aggregative paradigm remains attractive because it naturally extends to settings in which supervision is available only at the sample level, i.e., where training samples are labelled exclusively by prevalence and individual labels are never observed. It is precisely in this direction that recent neural approaches have emerged.

In recent years, several neural architectures specifically designed to process set-valued data have been proposed. Although these architectures were not originally devised for quantification, they are naturally suited to prevalence esti-

mation because they are built upon permutation-invariant (unbiased towards ordering) and variable-size (unbiased towards sample size) representations.²

The first of these architectures is DeepSets [35], that relies on permutation-invariant pooling operators such as max, average, or median, to aggregate instances into a fixed-size representation. In [28], the authors proposed DQN, which adapts DeepSets to quantification problems. A more expressive alternative was later proposed in [17], which replaced the simple pooling operators of DeepSets with transformer-based attention mechanisms capable of modelling interactions among the elements of the set. The method, called “Set Transformers”, uses the attention mechanism of transformers omitting positional encodings (since the order of the instances is irrelevant). In order to reduce the combinatorial complexity, attention is computed with respect to a reduced set of learnable *inducing points* which act as latent representatives of the input space.

More recently, [27] proposed HistNetQ, which can be viewed as a neural counterpart of HDx. HistNetQ replaces traditional histograms with learnable histogram layers, where both the bin centers and bin widths are trainable parameters and the histogram operator itself is differentiable. Unlike classical non-aggregative quantifiers, HistNetQ was specifically designed for weakly supervised scenarios in which training samples are labelled only by prevalence and instance-level labels are unavailable. To mitigate the severe data scarcity that typically arises in this setting, HistNetQ introduces a sample-mixing strategy that generates additional training samples by combining existing ones and inferring silver prevalence labels from the resulting mixtures. As in HDx, however, HistNetQ ultimately relies on a factorized representation of the feature space through per-dimension histograms, which inevitably assumes independence among feature dimensions. To address this limitation, [26] proposed GMNet, which replaces histogram-based pooling with learnable multivariate Gaussian mixtures capable of modelling dependencies between feature dimensions.

3 Background and notation

In this section we introduce basic concepts and notation on both quantification and graph learning.

Quantification. Let $\mathcal{X}$ be a generic input domain and $\mathcal{Y}$ be a discrete and finite set of class labels. Assume a dataset of pairs $\{(\mathbf{x}, y) \mid \mathbf{x} \in \mathcal{X}, y \in \mathcal{Y}\}$ which gives point-wise values of some unknown function we wish to learn. For the sake of simplicity, we present our method for the binary case $\mathcal{Y} = \{\oplus, \ominus\}$, using $\oplus$ to denote the “positive class” and $\ominus$ to denote the “negative class”.

We use symbol S to denote a *sample set*, i.e., a non-empty subset of (labelled or unlabelled) elements from $\mathcal{X}$. Let $p_S(y)$ be the true prevalence of class y in sample set S (i.e., the fraction of items in S that belong to y). Note that $p_S(y)$ is just a shorthand of $\Pr(Y = y \mid \mathbf{x} \in S)$, where $\Pr(\cdot)$ indicates probability and Y is

² A function f is said to be permutation-invariant if $f(S) = f(\pi(S))$ for any permutation π of the input sequence S .

a random variable that ranges on $\mathcal{Y}$. In the binary case, since $p_S(\ominus) = 1 - p_S(\oplus)$, quantification may be viewed as the task of estimating the prevalence of the positive class only. A *binary quantifier* q is thus a predictor of the class prevalence $p_S(\oplus)$ in samples $S \subset \mathcal{X}$. We will use the notation $\hat{p}_S^q(y)$ to indicate that $\hat{p}_S(y)$ (the estimate of $p_S(y)$) has been computed by quantifier q .³

Graphs. We define an undirected graph G as a set of nodes $\mathcal{V} = \{v_i : 1 \leq i \leq |\mathcal{V}|\}$ and a set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges among them, encoded as a symmetric adjacency matrix $\mathbf{A} \in \{0, 1\}^{|\mathcal{V}| \times |\mathcal{V}|}$ whose entries $\mathbf{A}_{ij}$ are 1 if $(v_i, v_j) \in \mathcal{E}$ and 0 otherwise. We define the set of *neighbours* of a node v_i as $\mathcal{N}(v_i) = \{v_j : (v_i, v_j) \in \mathcal{E}\}$, and the set of node feature-based representations as $\mathbf{X} = \{\mathbf{x}_v \in \mathbb{R}^d : v \in \mathcal{V}\}$ for some $d \in \mathbb{N}$.

Node quantification. Graph node quantification obviously has similarities with the *node classification* task. We assume the set of nodes $\mathcal{V}$ to be partitioned into a set $\mathcal{V}_L$ of labelled nodes and a set $\mathcal{V}_U$ of unlabelled nodes. With this setup, node classification consists of inferring the labels of the nodes in $\mathcal{V}_U$ via a classifier trained on the nodes in $\mathcal{V}_L$. Typically, node classification is viewed as a *transductive* task, meaning that the learning algorithm has access to the entire structure of G (i.e., to the set $\mathcal{V}$ of nodes and to the set $\mathcal{E}$ of edges) as well as (i) to the set $\mathbf{X}$ of the feature-based representations of all the nodes in G , and (ii) to the set $\{y_v : v \in \mathcal{V}_L\}$ of the labels of $\mathcal{V}_L$. The goal is to predict the labels $\{y_v : v \in \mathcal{V}_U\}$. A graph node quantifier q has also a transductive nature, the difference from a node classifier being that q has the form $q : 2^{\mathcal{V}_U} \rightarrow [0, 1]$, since its goal is to predict the prevalence values $p_S(\oplus)$ of samples $S \in 2^{\mathcal{V}_U}$.

4 A non-aggregative quantification method for graph nodes

We view non-aggregative node quantification as a regression problem where a model f is tasked to predict the prevalence of the positive class $\hat{p}_S(\oplus) \in [0, 1]$. The model itself can be viewed as a composition of parametric functions $f_{(\theta, \phi)} = \text{reg}_\phi \circ \text{pool} \circ \text{emb}_\theta$, where emb_θ is a node embedder that leverages structural information from the entire graph G ; pool is a subgraph aggregator, i.e., a non-parametric function that generates an embedding for the nodes in S ; and reg_ϕ is a downstream regressor mapping subgraph embeddings into $[0, 1]$ prevalence estimates $\hat{p}_S(\oplus)$. Below, we describe each component separately.

4.1 Node embedder

Given the input graph G with each node $v \in \mathcal{V}$ having d -dimensional input features $\mathbf{x}_v \in \mathbb{R}^d$, the node embedder emb_θ provides a d' -dimensional embedding $\mathbf{h}_v \in \mathbb{R}^{d'}$. We employ a Graph Echo State Network (GESN) [9, 10] to efficiently

³ We omit the superscript when the quantification method is clear from the context.

provide node encoding without resorting to training its parameters $\theta = (\mathbf{W}_x \in \mathbb{R}^{d \times d'}, \mathbf{W}_h \in \mathbb{R}^{d' \times d'}, \mathbf{b}_x \in \mathbb{R}^{d'})$. Node embeddings are computed via

$$\mathbf{h}_v^t = \begin{cases} \mathbf{0} & \text{if } t = 0, \\ \tanh\left(\mathbf{W}_x \mathbf{x}_v + \sum_{u \in \mathcal{N}(v)} \mathbf{W}_h \mathbf{h}_u^{t-1} + \mathbf{b}_x\right) & \text{for } 1 \leq t \leq T \end{cases} \quad (1)$$

where $\mathbf{W}_x$ controls the influence of input features on node embeddings; $\mathbf{W}_h$ regulates the dynamics, influencing whether long-range or short-range interactions between nodes will be encoded in final embeddings $\mathbf{h}_v^T$ [21]; and T is the number of iterations, which must be at least as large as the input graph diameter [21]. The input-to-hidden projection $\mathbf{W}_x$, the recurrent weight matrix $\mathbf{W}_h$, and the bias term $\mathbf{b}_x$ are randomly initialized and left untrained. In particular, matrix entries are initialized i.i.d. as $\text{Unif}[-scale, +scale]$, while the respective scales are left to model selection as hyperparameters.

4.2 Subgraph aggregator

The aggregator **pool** computes a representation for the entire sample S from the set of embeddings representing the individual nodes in S , via

$$\mathbf{h}_S = \text{pool}(\{\mathbf{h}_v^T : v \in S\}) \quad (2)$$

In principle, **pool** can be any permutation-invariant operator, such as sum, max, mean, or any learned aggregation.

For simplicity, as the pooling operator we adopt the mean of the sample.

4.3 Downstream regressor

Finally, the regressor **reg** with parameters $\phi = (\mathbf{w}_{out} \in \mathbb{R}^{d'}, b_{out} \in \mathbb{R})$ transforms the sample embedding returned by **pool** into a prediction $\hat{p}_S(\oplus)$, via

$$\hat{p}_S(\oplus) = \sigma(\mathbf{w}_{out} \mathbf{h}_S + b_{out}) \quad (3)$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function of logistic regression, that squashes the output in the range $[0, 1]$.

The regressor can be learned by minimizing the mean-squared-error objective function over a training dataset $\mathcal{D}$

$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum_{S \in \mathcal{D}} (\hat{p}_S(\oplus) - p_S(\oplus))^2 \quad (4)$$

Alternatively, we propose to learn **reg** using the beta-regression parameterization

$$\begin{aligned} \hat{\alpha}_S &= \text{softplus}(\mathbf{w}_\alpha \mathbf{h}_S + b_\alpha) \\ \hat{\beta}_S &= \text{softplus}(\mathbf{w}_\beta \mathbf{h}_S + b_\beta) \\ \hat{p}_S(\oplus) &= \frac{\hat{\alpha}_S}{\hat{\alpha}_S + \hat{\beta}_S} \end{aligned} \quad (5)$$

in which case the objective function minimized during training is

$$\begin{aligned} \mathcal{L} = \sum_{S \in \mathcal{D}} [& -\log \Gamma(\hat{\alpha}_S + \hat{\beta}_S) + \log \Gamma(\hat{\alpha}_S) + \log \Gamma(\hat{\beta}_S) \\ & - (\hat{\alpha}_S - 1) \log p_S(\oplus) - (\hat{\beta}_S - 1) \log(1 - p_S(\oplus))] \end{aligned} \quad (6)$$

Here, $\Gamma(\cdot)$ is the standard Gamma function, which appears through the Beta function normalization constant

$$B(\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \quad (7)$$

Thus, the terms involving $\log \Gamma(\cdot)$ in the objective are the usual log-normalization terms of the Beta likelihood. Note that the trainable quantities are not $\hat{\alpha}_S$ and $\hat{\beta}_S$ directly, as these are sample-specific outputs induced by the weights $\mathbf{w}_\alpha$, b_α , $\mathbf{w}_\beta$, and b_β . During optimization, gradients flow through $\frac{\partial \mathcal{L}}{\partial \hat{\alpha}_S}$ and $\frac{\partial \mathcal{L}}{\partial \hat{\beta}_S}$, then through the softplus activation, and then back to the weights.

4.4 Training and inference

To train the downstream regressor, we generate synthetic labelled-node subsets with controlled class prevalence. This mirrors the Artificial Prevalence Protocol (APP) used in standard quantification for evaluation [4, §3.4.2] since, instead of relying on the natural class distribution in $\mathcal{V}_L$, we construct subsets whose prevalence values span the full binary prevalence range.

Let $\mathcal{V}_L^+ = \{v \in \mathcal{V}_L : y_v = \oplus\}$ and $\mathcal{V}_L^- = \{v \in \mathcal{V}_L : y_v = \ominus\}$ denote the positively and negatively labelled nodes, respectively. We fix the training subset size to $K = \min\{|\mathcal{V}_L^+|, |\mathcal{V}_L^-|\}$ which guarantees that the full range of class prevalence values can be used. Indeed, for any $m \in \{0, \dots, K\}$, one can sample m positive nodes and $(K - m)$ negative nodes without resorting to duplicate datapoints in the same sample. We thus define the feasible APP prevalence value grid induced by K as

$$\mathcal{R}_K = \left\{0, \frac{1}{K}, \frac{2}{K}, \dots, 1\right\}$$

For each training example, we sample $m \sim \text{Unif}\{0, \dots, K\}$. Conditional on m , we independently sample

$$\begin{aligned} S_m^+ &\sim \text{Unif}\{S \subseteq \mathcal{V}_L^+ : |S| = m\} \\ S_m^- &\sim \text{Unif}\{S \subseteq \mathcal{V}_L^- : |S| = (K - m)\} \end{aligned}$$

The resulting labelled-node subset is $S_m = S_m^+ \cup S_m^-$.

The pooled representation of S_m (i.e., the concrete implementation of Equation 2) is computed as

$$\bar{\mathbf{h}}_{S_m} = \frac{1}{|S_m|} \sum_{v \in S_m} \mathbf{h}_v^T$$

Algorithm 1: APP-based Prevalence Embedding Generation

Input: Labelled node representations $\{(\mathbf{h}_v^T, y_v) \mid v \in \mathcal{V}_L\}$, dataset size B
Output: Dataset $\mathcal{D} = \{(\bar{\mathbf{h}}_{S_i}, \bar{p}_{S_i}(\oplus))\}_{i=1}^B$

- 1 $\mathcal{V}_L^+ \leftarrow \{v \in \mathcal{V}_L \mid y_v = \oplus\}$ // positively labelled nodes
- 2 $\mathcal{V}_L^- \leftarrow \{v \in \mathcal{V}_L \mid y_v = \ominus\}$ // negatively labelled nodes
- 3 $K \leftarrow \min\{|\mathcal{V}_L^+|, |\mathcal{V}_L^-|\}$ // fixed subset size
- 4 $\mathcal{R}_{\text{APP}} \leftarrow \{0, 0.05, 0.10, \dots, 1\}$ // APP target grid
- 5 $\mathcal{D} \leftarrow \emptyset$
- 6 **for** $i \leftarrow 1$ **to** B **do**
 - // sample a target prevalence from the APP grid
 - 7 $\rho \sim \text{Unif}(\mathcal{R}_{\text{APP}})$
 - // map the target prevalence to a feasible class count
 - 8 $m_\rho \leftarrow \text{round}(K\rho)$
 - // sample a subset with m_ρ positives and $K - m_\rho$ negatives
 - 9 $S_i^+ \sim \text{Unif}\{S \subseteq \mathcal{V}_L^+ : |S| = m_\rho\}$
 - 10 $S_i^- \sim \text{Unif}\{S \subseteq \mathcal{V}_L^- : |S| = K - m_\rho\}$
 - 11 $S_i \leftarrow S_i^+ \cup S_i^-$
 - // create the pooled representation and prevalence target
 - 12 $\bar{\mathbf{h}}_{S_i} \leftarrow |S_i|^{-1} \sum_{v \in S_i} \mathbf{h}_v^T$
 - 13 $\bar{p}_{S_i}(\oplus) \leftarrow m_\rho/K$
 - 14 $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\bar{\mathbf{h}}_{S_i}, \bar{p}_{S_i}(\oplus))\}$
- 15 **return** $\mathcal{D}$

and the corresponding prevalence target is

$$\bar{p}_{S_m}(\oplus) = \frac{m}{K}$$

Repeating this procedure yields a training dataset $\mathcal{D} = \{(\bar{\mathbf{h}}_{S_i}, \bar{p}_{S_i}(\oplus))\}_{i=1}^B$, which is used to learn the parameters of the downstream regressor reg_ϕ , while the parameters of the node embedder are kept fixed.

In practice, when using the conventional APP target grid $\{0, 0.05, 0.1, \dots, 1\}$, one may draw a target prevalence ρ from this grid and set $m_\rho = \text{round}(K\rho)$. The realized prevalence is then m_ρ/K . This coincides exactly with ρ whenever $K\rho$ is an integer for all grid values, for example when K is a multiple of 20. Otherwise, the protocol realizes the closest feasible prevalence values induced by the fixed subset size K . At inference time, the learned regressor can be applied to any subset $S \subseteq \mathcal{V}_U$. Its node embeddings are pooled using the same aggregation rule, and the regressor returns the estimate

$$\hat{p}_S(\oplus) = \text{reg}_\phi(\bar{\mathbf{h}}_S)$$

The method therefore estimates the prevalence of a node subset directly, without training an intermediate node-level classifier. A practical realization of the graph-node APP protocol for training is given in Algorithm 1.

5 Experiments

We designed our experiments to answer the following research questions:

- Q1: Are non-aggregative node quantification methods competitive with aggregative node quantification methods?
- Q2: What is the influence of graph structure in quantification?

5.1 Datasets

Following [20], we selected two publicly available datasets for our comparison:

- CORA [19] is a citation network of scientific publications, where links represent citations. The original task is multi-class classification: given textual node features extracted from the abstract, a model is tasked to predict the scientific field to which a publication belongs, among 7 alternatives [34]. We binarized the task by assigning the $\oplus$ label to nodes with class 2, and class $\ominus$ to the remaining nodes.
- TOLOKERS is a network derived from the Toloka platform [18]. In the corresponding graph, nodes are workers that participate in crowd-sourcing projects, while links specify participation in the same project. The $\oplus$ class corresponds to whether a user was banned from the community [25].

We choose these two datasets because of several reasons. On the one hand, the CORA graph is homophilic (meaning, nodes with the same label are more often linked together),

while TOLOKER is heterophilic (the opposite scenario, where if two nodes have different label, they are more likely to be linked together). Therefore, these two datasets allow us to test how the connectivity patterns in the graph influence the predictions. On the other hand, while CORA is substantially a small and sparse graph (consisting of 2,000 nodes and 5,000 edges approximately), TOLOKER is 5 times larger and several orders of magnitude denser (11,000 nodes approx., over 1M edges). This allows us to study how the scale and density of the input graph influences the predictions.

5.2 Evaluation measures

To assess performance, we resort to the standard quantification practice of simulating PPS to evaluate how the quantifier estimates change as the test prevalences diverge from the training prevalences. In the binary case, PPS can be simulated through the *artificial prevalence protocol* (APP) [4, §3.4.2], which involves extracting from the unlabelled data test subsets with predetermined prevalence (according to a fixed grid such as $\{0.00, 0.05, \dots, 0.95, 1.00\}$). As performance metrics, we use the *Absolute Error* (AE) between the true prevalence and the estimated prevalence; in the binary case, AE is defined as

$$\text{AE}(p_S, \hat{p}_S) = |\hat{p}_S(\oplus) - p_S(\oplus)|. \quad (8)$$

In addition to AE, we also assess performance using the *relative absolute error* (RAE), which in the binary case is defined as

$$\text{RAE}(p_S, \hat{p}_S) = \frac{1}{2} \sum_{y \in \{\oplus, \ominus\}} \frac{|\hat{p}_S(y) - p_S(y)|}{p_S(y) + \varepsilon}. \quad (9)$$

Above, ε is a smoothing factor used to avoid division by zero when $p_S(y) = 0$. In our experiments we set $\varepsilon = 1/(2|S|)$, following [8], and we report the mean AE (MAE) and mean RAE (MRAE), which corresponds to the average AE and RAE values across all the test subsets S sampled via APP.

5.3 Evaluation protocols

The evaluation was performed via 5-fold cross validation. We partitioned the node set $\mathcal{V}$ (stratifying by node labels). During each CV round, we reserved one partition for evaluation and four partitions to select the best hyperparameter configuration using grid search with a hold-out (80/20) training-validation split. For the proposed non-aggregative quantifier, we choose the embedding dimension from $\{512, 1024, 2048, 4096\}$, the recurrent scale from $\{1.0, 10.0, 100.0, 1000.0\}$, and the input scale from $\{0.1, 0.5, 1.0\}$, while setting $T = 30$.

For the NoEdge baseline, we choose the number of layers from $\{1, 2, 3, 4\}$ and the embedding dimension from $\{64, 128, 256\}$. All downstream regressors are trained using gradient descent. We employed the Adam optimizer with learning rate 0.001 and weight decay 0.00001. In each CV round, the best model was selected according to the best validation AE. The chosen model was subsequently evaluated with APP using a grid of prevalence values $\{0, 0.05, 0.10, \dots, 1\}$; we generate 100 samples for each value in the grid, thus totaling 2100 samples, of 500 instances each, per each combination of dataset and method.

5.4 Baselines

In our experiments, we are interested in establishing:

- if relational information is useful to improve performance. To this end, we compare our non-aggregative GESN to a baseline which does not exploit relational information (termed NoEdge). Basically, the NoEdge baseline applies one or more hidden layers to the node features, without message passing.
- if non-aggregative GESN is competitive with state-of-the-art aggregative node quantifiers (and in particular, XNQ). To this end, we compare our results to those obtained by XNQ, although we note that this comparison is only useful as a reference, since the experimental setups differ.
- which downstream regressor works better in our learning setup, between linear regression in the $[0, 1]$ interval and Beta regression. To this end, we compare both NoEdge and GESN with a linear regression-based head, or with a Beta regression-based head.

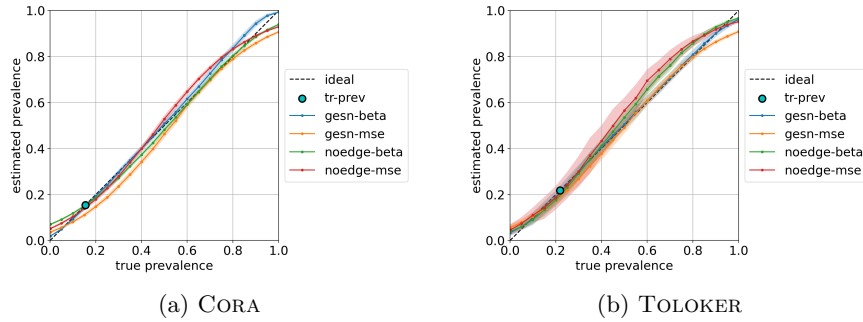


Fig. 1: Diagonal plots showing the estimated prevalence of class $\oplus$ versus its true prevalence. Curves represent mean estimates and shaded regions denote one standard deviation. Curves closer to the main diagonal (corresponding to perfect quantification) indicate better performance.

6 Results

Table 1 reports a comparison with aggregative baselines. Both GESN variants achieve performance close to XNQ, the state-of-the-art aggregative method, and GESN+Beta exhibits overlapping error bars with XNQ. The GESN-based non-aggregative methods also tend to outperform the remaining aggregative quantifiers. This comparison should nevertheless be interpreted with some caution. Our experimental setup differs from that of [20], since the non-aggregative methods considered here do not require external calibration data. Thus, the results are useful for contextualizing the proposed methods against aggregative alternatives, but they should not be read as a strictly like-for-like ranking.

Focusing on the non-aggregative methods, two trends emerge. First, the GESN-based methods outperform their NoEdge counterparts, suggesting that the graph structure provides useful information for prevalence estimation. Second, the Beta-regression variants generally improve over the linear-regression variants as downstream regressor. One possible explanation to this result is that the standard sigmoid-output regressor treats prevalence prediction as a point-estimation problem in $[0, 1]$, whereas the Beta parameterization explicitly models prevalence as a bounded response variable through sample-specific shape parameters. This added flexibility could be beneficial especially when prevalences are strongly skewed toward one class.

Finally, Figure 1 shows the diagonal plots (see [4]) of the models, in which the estimated prevalence $\hat{p}_S(\oplus)$ (y-axis) is displayed as a function of the true prevalence $p_S(\oplus)$ (x-axis) for CORA (panel (a)) and TOLOKER (panel (b)). These plots reveal two main findings: first, our proposed GESN-Beta model produces predictions closer to the diagonal, which represents the perfect quantifier; second, that the GESN-based variants typically exhibit lower variability with respect to NoEdge baselines.

Table 1: Results of our experiments. For the MAE and MRAE metrics, we report their 5-fold nested CV average $\pm$ standard deviation. Best results for each dataset are shown underlined for non-aggregative methods, **boldfaced** for best overall.

Dataset	Quantifier	MAE	MRAE
CORA	<i>Non-Aggregative</i> (this work)		
	NoEdge + LR	0.040 $\pm$ 0.010	2.840 $\pm$ 0.801
	NoEdge + Beta	0.072 $\pm$ 0.037	4.698 $\pm$ 1.697
	GESN + LR	0.028 $\pm$ 0.005	2.701 $\pm$ 0.287
	GESN + Beta	<u>0.019</u> $\pm$ 0.013	<u>0.714</u> $\pm$ 0.515
	<i>Aggregative</i> (taken from [20])		
	CBQ	0.037 $\pm$ 0.020	0.976 $\pm$ 1.758
	CDQ	0.100 $\pm$ 0.045	1.679 $\pm$ 1.101
	ENQ	0.029 $\pm$ 0.008	0.751 $\pm$ 0.987
	XNQ	0.015 $\pm$ 0.001	0.241 $\pm$ 0.374
TOLOKER	<i>Non-Aggregative</i> (this work)		
	NoEdge + LR	0.070 $\pm$ 0.011	2.617 $\pm$ 0.981
	NoEdge + Beta	0.052 $\pm$ 0.015	2.283 $\pm$ 1.057
	GESN + LR	0.044 $\pm$ 0.024	2.989 $\pm$ 0.617
	GESN + Beta	<u>0.036</u> $\pm$ 0.014	<u>1.483</u> $\pm$ 0.564
	<i>Aggregative</i> (taken from [20])		
	CBQ	0.079 $\pm$ 0.036	6.151 $\pm$ 8.407
	CDQ	0.358 $\pm$ 0.141	20.150 $\pm$ 12.42
	ENQ	0.099 $\pm$ 0.036	2.903 $\pm$ 1.398
	XNQ	0.034 $\pm$ 0.007	0.441 $\pm$ 0.253

7 Conclusions

In this paper we have presented a non-aggregative method for network quantification based on Graph Echo State Networks (GESNs). Unlike existing graph quantifiers, which typically follow the aggregative paradigm and rely on an intermediate node classifier, our approach estimates class prevalence values directly from sets of node representations. This makes it consistent with Vapnik’s principle and particularly appealing in scenarios where aggregate-level inferences are desirable but individual-level predictions are not.

Our experimental results show that the proposed approach achieves performance comparable to state-of-the-art aggregative quantifiers while avoiding the need for an intermediate classification step. Among the proposed variants, GESN combined with a Beta-regression head consistently delivered the strongest results. Our experiments also show that exploiting relational information from the graph structure is beneficial for prevalence estimation. Taken together, these results suggest that non-aggregative quantification is a viable alternative to aggregative approaches in graph-based settings.

Several directions for future work remain open. First, while our experiments focused on binary quantification under prior probability shift, extending the proposed framework to multiclass scenarios is a natural next step. Second, we aim to test our algorithm under different types of shift, including the “structural covariate shift” discussed in [2, 3]. Finally, it would be interesting to explore more sophisticated pooling operators and to investigate the applicability of the proposed approach to weakly supervised settings in which only sample-level prevalence labels are available and instance-level annotations are never collected.

Acknowledgments. Research partly supported by Project DEEP-GRAPH, funded by the Italian Ministry of University and Research, PRIN 2022 (project code: 2022YL-RBTT, CUP: I53C24002440006), and Project PAN-HUB, funded by the Italian Ministry of Health (POS 2014–2020, project ID: T4-AN-07, CUP: I53C22001300001).

Bibliography

- [1] Asokere, M., Wusu, A., Olabanjo, O.: Twitter (X) as an electoral barometer: Systematic evidence from sentiment analysis of Twitter data. *International Journal of Information Technology* pp. 1–24 (2025). <https://doi.org/10.1007/s41870-025-03039-1>
- [2] Damke, C., Hüllermeier, E.: Adjusted count quantification learning on graphs. In: *Proceedings of the 39th Annual Conference on Neural Information Processing Systems (NeurIPS 2025)*. Mexico City, MX (2025)
- [3] Damke, C., Hüllermeier, E.: Distribution matching for graph quantification under structural covariate shift. In: *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2025)*. pp. 403–419. Porto, PT (2025). https://doi.org/10.1007/978-3-032-05981-9_24
- [4] Esuli, A., Fabris, A., Moreo, A., Sebastiani, F.: *Learning to quantify*. Springer Nature, Cham, CH (2023). <https://doi.org/10.1007/978-3-031-20467-8>
- [5] Fabris, A., Esuli, A., Moreo, A., Sebastiani, F.: Measuring fairness under unawareness of sensitive attributes: A quantification-based approach. *Journal of Artificial Intelligence Research* **76**, 1117–1180 (2023). <https://doi.org/10.1613/jair.1.14033>
- [6] Firat, A.: Unified framework for quantification. arXiv:1606.00868v1 [cs.LG] (2016)
- [7] Forman, G.: Counting positives accurately despite inaccurate classification. In: *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*. pp. 564–575. Porto, PT (2005). https://doi.org/10.1007/11564096_55
- [8] Forman, G.: Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery* **17**(2), 164–206 (2008). <https://doi.org/10.1007/s10618-008-0097-y>
- [9] Gallicchio, C., Micheli, A.: Graph echo state networks. In: *Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN 2020)*.

- pp. 3967–3974. Barcelona, ES (2010). <https://doi.org/10.1109/IJCNN.2010.5596796>
- [10] Gallicchio, C., Micheli, A.: Fast and deep graph neural networks. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI 2020). pp. 3898–3905. New York, US (2020). <https://doi.org/10.1609/AAAI.V34I04.5803>
 - [11] Gammerman, A., Vovk, V.G., Vapnik, V.: Learning by transduction. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI 1998). pp. 148–155. Madison, US (1998)
 - [12] González, P., Castaño, A., Chawla, N.V., del Coz, J.J.: A review on quantification learning. *ACM Computing Surveys* **50**(5), 74:1–74:40 (2017). <https://doi.org/10.1145/3117807>
 - [13] González-Castro, V., Alaiz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the Hellinger distance. *Information Sciences* **218**, 146–164 (2013). <https://doi.org/10.1016/j.ins.2012.05.028>
 - [14] Hopkins, D.J., King, G.: A method of automated nonparametric content analysis for social science. *American Journal of Political Science* **54**(1), 229–247 (2010). <https://doi.org/10.1111/j.1540-5907.2009.00428.x>
 - [15] Jerzak, C.T., King, G., Strezhnev, A.: An improved method of automated nonparametric content analysis for social science. *Political Analysis* **31**(1), 42–58 (2023). <https://doi.org/10.1017/pan.2021.36>
 - [16] Kawakubo, H., du Plessis, M.C., Sugiyama, M.: Computationally efficient class-prior estimation under class balance change using energy distance. *IEICE Transactions on Information Systems* **99-D**(1), 176–186 (2016). <https://doi.org/10.1587/transinf.2015EDP7212>
 - [17] Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., Teh, Y.W.: Set transformer: A framework for attention-based permutation-invariant neural networks. In: Proceedings of the 36th International Conference on Machine Learning (ICML 2019). pp. 3744–3753. Long Beach, US (2019)
 - [18] Likhobaba, D., Pavlichenko, N., Ustalov, D.: Toloker graph: Interaction of crowd annotators (2023). <https://doi.org/10.5281/zenodo.7620795>
 - [19] McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of Internet portals with machine learning. *Information Retrieval* **3**(2), 127–163 (2000). <https://doi.org/10.1023/a:1009953814988>
 - [20] Micheli, A., Moreo, A., Podda, M., Sebastiani, F., Simoni, W., Tortorella, D.: Efficient quantification on large-scale networks. *Machine Learning* **114**(12), Article 270 (2025). <https://doi.org/10.1007/s10994-025-06915-w>
 - [21] Micheli, A., Tortorella, D.: Addressing heterophily in node classification with graph echo state networks. *Neurocomputing* **550**, 126506 (2023). <https://doi.org/10.1016/j.neucom.2023.126506>
 - [22] Milli, L., Monreale, A., Rossetti, G., Pedreschi, D., Giannotti, F., Sebastiani, F.: Quantification in social networks. In: Proceedings of the 2nd IEEE International Conference on Data Science and Advanced Analytics (DSAA 2015). pp. 596–605. Paris, FR (2015). <https://doi.org/10.1109/dsaa.2015.7344845>

- [23] Moreno-Torres, J.G., Raeder, T., Alaíz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. *Pattern Recognition* **45**(1), 521–530 (2012). <https://doi.org/10.1016/j.patcog.2011.06.019>
- [24] Moreo, A., González, P., del Coz, J.J.: Kernel density estimation for multi-class quantification. *Machine Learning* **114**(4) (2025). <https://doi.org/10.1007/s10994-024-06726-5>
- [25] Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., Prokhorenkova, L.: A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In: *Proceedings of the 11th International Conference on Learning Representations (ICLR 2023)*. Kigali, RW (2023)
- [26] Pérez-Mon, O., del Coz, J.J., González, P.: Quantification via Gaussian latent space representations. *Neural Networks* **201**, 108886 (2026). <https://doi.org/https://doi.org/10.1016/j.neunet.2026.108886>
- [27] Pérez-Mon, O., Moreo, A., del Coz, J.J., González, P.: Quantification using permutation-invariant networks based on histograms. *Neural Computing and Applications* **37**, 3505–3520 (2025). <https://doi.org/10.1007/s00521-024-10721-1>
- [28] Qi, L., Khaleel, M., Tavanapong, W., Sukul, A., Peterson, D.A.M.: A framework for deep quantification learning. In: *Proceedings of the 2020 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2020)*. pp. 232–248. Ghent, BE (2020). https://doi.org/10.1007/978-3-030-67658-2_14
- [29] Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D. (eds.): *Dataset shift in machine learning*. The MIT Press, Cambridge, US (2009). <https://doi.org/10.7551/mitpress/9780262170055.001.0001>
- [30] Saerens, M., Latinne, P., Decaestecker, C.: Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure. *Neural Computation* **14**(1), 21–41 (2002). <https://doi.org/10.1162/089976602753284446>
- [31] Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. *AI Magazine* **29**(3), 93–106 (2008)
- [32] Tang, L., Gao, H., Liu, H.: Network quantification despite biased labels. In: *Proceedings of the 8th Workshop on Mining and Learning with Graphs (MLG 2010)*. pp. 147–154. Washington, US (2010). <https://doi.org/10.1145/1830252.1830271>
- [33] Vapnik, V.: *Statistical learning theory*. Wiley, New York, US (1998)
- [34] Yang, Z., Cohen, W.W., Salakhutdinov, R.: Revisiting semi-supervised learning with graph embeddings. In: *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*. pp. 40–48. New York, NY (2016)
- [35] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R., Smola, A.J.: Deep sets. In: *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*. pp. 3391–3401. Long Beach, US (2017)

The Fragility of Value in Customer Churn: Predict-then-Optimize vs. Predict-and-Optimize Under Dataset Shift

Shimanto Rahman^{1,2}, Bram Janssens¹, and Matthias Bogaert^{1,2}

¹ Ghent University, Ghent, Belgium

{shimanto.rahman, bram.janssens, matthias.bogaert}@UGent.be

² FlandersMake@UGent-corelab CVAMO, Ghent, Belgium

Abstract. Customer churn prediction (CCP) models are essential tools for identifying consumers at risk of attrition. However, traditional CCP approaches prioritize statistical accuracy over financial return, often ignoring the inherent value disparity among individual customers. This limitation has driven the adoption of cost-sensitive learning, broadly categorized into two approaches: *Predict-then-Optimize* (PtO), which trains standard probabilistic models and applies economic decision rules post-hoc, and *Predict-and-Optimize* (PaO), which directly integrates misclassification costs into the training objective. While recent literature has evaluated these approaches on static datasets, models in real-world business applications are typically deployed into non-stationary environments operating under static, unyielding decision policies. This research investigates the economic fragility of cost-sensitive CCP approaches under dataset shift. Utilizing a Gaussian copula architecture on an empirical telecommunications dataset, we decouple marginal behavior from feature covariance to systematically induce covariate and label drift. These statistical perturbations mirror real-world phenomena such as economic recessions and shifts in consumer pricing models. Our findings reveal a fundamental operational trade-off: while PtO maximizes in-distribution profitability, its reliance on probabilistic calibration makes it vulnerable to scale-based shifts. Conversely, PaO approaches establish a defensive moat, utilizing stable, uncalibrated score densities to maintain profitability during structural covariance drift. This study provides a comprehensive taxonomy of economic model decay, offering vital guidance for deploying cost-sensitive classifiers in volatile business applications.

Keywords: Predict-then-Optimize · Predict-and-Optimize · Cost-Sensitive Learning · Customer Churn Prediction · Dataset Shift.

1 Introduction

Customer Relationship Management (CRM) heavily relies on retaining existing clientele, as acquiring new customers incurs substantially higher costs than preserving current ones [10]. To facilitate this, organizations deploy Customer

Churn Prediction (CCP) models, which analyze historical behavior to identify individuals with a high probability of terminating their relationship with the company [17]. While traditional CCP approaches treat all customers equally by optimizing purely for statistical accuracy (e.g., maximizing the Area Under the ROC Curve), this assumption is fundamentally misaligned with the financial realities of business applications [18]. Because customers exhibit heterogeneity in their Customer Lifetime Value (CLV), ranking interventions solely based on attrition probability frequently results in suboptimal and unprofitable retention campaigns [11].

To resolve this misalignment, the literature has established two distinct cost-sensitive learning approaches [16]. The first is the **Predict-then-Optimize (PtO)** approach. In this framework, standard classification algorithms are trained using proper scoring rules (e.g., cross-entropy) to maximize predictive accuracy [6], and economic optimization is deferred to the decision-making stage. Theoretical decision rules, such as Bayes Minimum Risk thresholds, are then applied directly to the model’s calibrated probabilities [4].

The second approach is the **Predict-and-Optimize (PaO)** paradigm. This stream argues that training a model independently of its operational context is economically suboptimal [16]. Instead, PaO integrates asymmetric cost matrices directly into the model training phase to explicitly minimize the Average Expected Cost [3] or maximize profit metrics like the Expected Maximum Profit [18]. Because bespoke cost-sensitive loss functions are improper scoring rules, PaO models output uncalibrated economic scores rather than true conditional probabilities. This requires practitioners to abandon theoretical Bayesian bounds and instead deploy empirical decision thresholds tuned on historical validation data.

While the literature rigorously debates the in-distribution superiority of these two approaches, a critical operational vulnerability remains unaddressed: their robustness under dataset shift. In real-world business applications, retention models are deployed with static, fixed threshold policies. When macroeconomic shocks induce distribution drift, the financial viability of these immutable policies decays. To address this gap, we utilize a continuous Gaussian copula framework to rigorously decouple marginal behavioral drift from underlying covariance structures, allowing us to investigate the following research questions:

1. Do the cost-aligned representations learned by PaO models exhibit greater robustness to shifting feature dependencies than the purely statistical representations of PtO models?
2. Under a systematic decline in customer usage, which failure mechanism proves more detrimental: the degradation of probabilistic calibration in PtO models, or the misalignment of empirical score densities in PaO models?
3. Can either approach maintain profitability during a sudden macroeconomic shift in the prior probability of churn while operating under fixed deployment policies?

2 Related Work

Customer attrition is a critical concern in business analytics, prompting a shift from accuracy-driven metrics to cost-sensitive strategies [18]. Traditional CCP models implicitly assume symmetric misclassification costs, fundamentally ignoring the economic reality that the financial upside of successfully retaining a true churner vastly outweighs the cost of erroneously incentivizing a loyal customer [17]. Early efforts to resolve this imbalance introduced class-dependent costs, which aggregate these tradeoffs by estimating the average benefit of retention against the global average cost of a campaign intervention. While recognizing this baseline asymmetry provides significant structural advantages, treating all targeted customers identically remains detrimental in industries characterized by high customer value heterogeneity, such as Business-to-Business sectors [11]. In these volatile environments, it is highly beneficial to further individualize the economic penalties utilizing instance-dependent costs. To conceptualize the evolution and implementation of these cost-sensitive methodologies, Vanderschueren et al. [16] proposed a unifying two-stage framework that categorizes the literature into the Predict-then-Optimize and Predict-and-Optimize approaches.

In the Predict-then-Optimize (PtO) approach, the training phase relies on proper scoring rules to estimate true conditional probabilities, deferring economic trade-offs to the downstream deployment stage [16]. Because the outputs of PtO models are assumed to be well-calibrated probabilities, the optimal business policy can be determined analytically post-hoc. For instance, applying theoretical Bayes Minimum Risk thresholds to standard classifier probabilities drastically improves financial savings compared to arbitrary, accuracy-based cut-offs [4]. However, the operational success of the PtO approach is inextricably linked to the model’s probabilistic calibration remaining perfectly intact upon deployment.

Conversely, the Predict-and-Optimize (PaO) paradigm asserts that training a model independently of its ultimate operational context yields suboptimal economic boundaries [16]. This approach integrates misclassification costs directly into the machine learning algorithm. Because true positive benefits and false positive costs in customer churn are inherently instance-dependent—varying drastically based on a customer’s individual Customer Lifetime Value [11]—algorithms such as Cost-Sensitive Logistic Regression [8] and Cost-Sensitive Decision Trees [2] alter their internal splitting criteria or gradients to explicitly minimize expected financial loss. Similarly, other methodologies directly integrate the Expected Maximum Profit Measure for Customer Churn (EMPC) into the training process utilizing evolutionary learning [15,9]. A profound theoretical consequence of direct PaO optimization is the loss of probability calibration. Even when practitioners attempt to make standard proper scoring rules (such as cross-entropy) cost-sensitive via asymmetric weighting, the optimization landscape fundamentally shifts to minimize financial risk rather than to elicit the true conditional probability. Because these bespoke economic loss functions deviate from standard maximum likelihood estimation and thus become mathematically improper, PaO models inherently output uncalibrated risk scores, forcing practitioners to rely on empirical decision policies tuned on historical validation data.

While both approaches aggressively optimize for economic value, existing literature overwhelmingly evaluates them under the assumption that the deployment data is drawn from the identical distribution as the training data. However, empirical studies in general churn management demonstrate that this static assumption is operationally flawed. For instance, Risselada et al. [14] established that the “staying power” of churn prediction models is notoriously low; predictive performance deteriorates rapidly post-estimation due to temporal shifts in customer behavior and changing parameter significance, fundamentally necessitating frequent model retraining. Despite this recognized reality, research addressing structural covariate drift in the feature space remains remarkably scarce within the cost-sensitive literature. To rigorously benchmark policy decay without destroying the complex dependencies of empirical churn data, modern methodologies employ Gaussian copulas [12,13]. This architecture allows for the precise, isolated induction of distinct drift typologies reflecting real-world business disruptions. By isolating marginal drift (representing macroscopic usage decline of customers, such as an economic recession) from covariance drift (representing the decoupling of customer behaviors, such as shifts to flat-rate pricing models), this framework provides a controlled environment to answer a critical operational question: under dataset shift, what fails first—the theoretical probabilistic calibration of a PtO model, or the empirical score density of a PaO model?

3 Methodology

To systematically evaluate the operational fragility of cost-sensitive classifiers under dataset shift, we ground our experimental framework in customer churn prediction. Churn management provides an ideal empirical setting due to its highly asymmetric financial consequences—wasting retention incentives on loyal customers versus forfeiting the lifetime value of actual defectors [17]. This section formalizes the economic parameters of this proactive retention scenario, details the translation of predictive scores into operational threshold policies, and outlines the synthetic drift architecture utilized to simulate adverse macroeconomic shocks.

3.1 Cost in Churn Management

We model a standard proactive campaign where an enterprise attempts to prevent attrition by offering predicted churners ($y_i = 1$) a financial incentive to remain. The economic consequences of a model’s interventions are formalized in a cost matrix relative to a baseline of inaction (Table 1). This relative formulation isolates the net impact of the campaign, treating financial gains as negative costs.

Adopted from Janssens et al. [11], the cost of a False Positive ($C_{FP,i}$) represents wasted expenditure comprising an individualized incentive cost (d_i) and a

Table 1. The instance-dependent cost matrix for a proactive customer retention campaign. Financial benefits (retained customer value) are expressed as negative costs. For class-dependent costs, the individualized parameters (CLV_i and d_i) are replaced by their global averages.

	Actual Positive ($y = 1$)	Actual Negative ($y = 0$)
Pred. Positive ($\hat{y} = 1$)	$C_{TP,i} = (1 - \gamma)f$ $-\gamma(CL V_i - d_i - f)$	$C_{FP,i} = d_i + f$
Pred. Negative ($\hat{y} = 0$)	$C_{FN,i} = 0$	$C_{TN,i} = 0$

fixed administrative contact cost (f). Conversely, a True Positive ($C_{TP,i}$) represents the expected net gain of successfully retaining a churner, explicitly factoring in their Customer Lifetime Value (CLV_i) and their probability of accepting the offer (γ). Following established specifications [18,11], we set $f = 1$, a static acceptance rate of $\gamma = 0.3$, and scale the retention incentive proportionally to individual value ($d_i = 0.05 \cdot CLV_i$).

To rigorously evaluate the impact of this economic granularity, we contrast this instance-dependent formulation against a class-dependent baseline. In the class-dependent approach, the individualized CLV_i is replaced by the global average Customer Lifetime Value ($\overline{CLV}$) across the training population. This aggregation yields static, uniform penalty terms (C_{TP} and C_{FP}) that treat all successful retentions and wasted incentives symmetrically, irrespective of a customer’s specific economic profile.

To optimize the models in the PaO approach, the objective function minimizes the Average Expected Cost (AEC). Given N instances and a continuous model output $\hat{y}_i \in [0, 1]$, the empirical risk is evaluated using the cost parameters corresponding to the chosen granularity (utilizing either the individualized $C_{TP,i}$ and $C_{FP,i}$, or the static C_{TP} and C_{FP}):

$$\text{AEC} = \frac{1}{N} \sum_{i=1}^N [y_i \hat{y}_i C_{TP,i} + (1 - y_i) \hat{y}_i C_{FP,i}] \quad (1)$$

3.2 The Two Decision Approaches: Theoretical vs. Empirical Thresholds

Because the PtO and PaO approaches optimize fundamentally different loss functions during training, they output distinct types of prediction scores. This architectural divergence necessitates specific operational policies to translate these scores into binary business actions (Figure 1).

Predict-then-Optimize (Theoretical Policy) PtO models minimize standard log-loss during training, outputting calibrated conditional probabilities ($\hat{p}_i$).

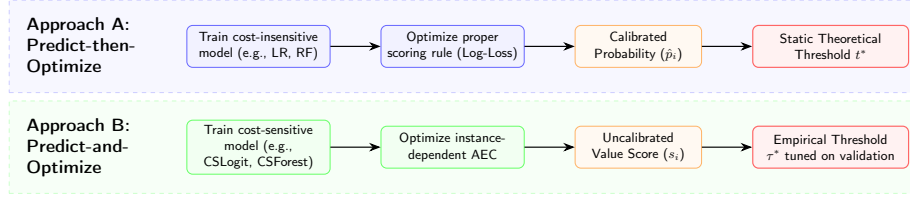


Fig. 1. The architecture of threshold policies. PtO models leverage proper scoring rules to establish theoretical thresholds. Conversely, PaO models output uncalibrated scores, mandating the use of empirical thresholds derived from historical validation data.

This enables the use of exact theoretical decision rules. By applying Bayes Minimum Risk [4], we first calculate the optimal, instance-specific theoretical threshold for each customer in the validation set:

$$t_i^* = \frac{C_{FP,i}}{-C_{TP,i} + C_{FP,i}} \quad (2)$$

However, to deploy a fixed, operational policy on unseen test data, we extract the median of these individual boundaries to establish a single static campaign threshold, denoted as t^* . An intervention is subsequently executed in the deployment environment strictly if the calibrated prediction $\hat{p}_i > t^*$.

Predict-and-Optimize (Empirical Policy) PaO models explicitly minimize the Average Expected Cost (AEC) during training. Because the AEC is an improper scoring rule, the resulting model outputs uncalibrated economic value scores ($s_i \in [0, 1]$) rather than probabilities. Consequently, applying the Bayesian formula (Eq. 2) to these scores is mathematically invalid. Instead, the PaO approach requires an empirical policy: the uncalibrated scores are systematically swept over a historical validation fold to identify the static, empirical threshold (τ^*) that minimizes the total AEC in-distribution.

3.3 Base Models and Empirical Dataset

To ensure a robust comparative analysis across varying geometric algorithmic properties, we select four standard machine learning architectures: Logistic Regression (LR), Decision Trees (DT), Random Forests (RF), and LightGBM (GBM). These algorithms serve as the cost-insensitive baselines within the PtO approach. They are evaluated directly against their specialized, cost-sensitive counterparts within the PaO approach: Cost-Sensitive Logistic Regression (CSLogit), Cost-Sensitive Decision Trees (CSTree), Cost-Sensitive Random Forests (CSForest), and Cost-Sensitive Gradient Boosting (CSBoost) [8,2].

We utilize the Iranian Churn Dataset [1], encompassing 3,150 customers and 13 attributes over a 12-month period. This specific dataset was selected because it provides continuous financial variables necessary for computing precise

CLV—an absolute prerequisite for instance-dependent cost-sensitive optimization. Furthermore, it contains highly granular continuous usage features (e.g., call durations, SMS frequency) required to effectively simulate structural feature decoupling and scale-based data drifts. To mirror proactive retention cycles, predictive features are aggregated over the first 9 months. The target label designates the churn state at the end of the 12th month, reflecting an operational 3-month intervention window.

3.4 Experimental Setup and Synthetic Drift Architecture

To guarantee statistical robustness and systematically benchmark policy decay, the experimental framework utilizes a continuous 13-dimensional Gaussian copula [12]. This generative architecture allows us to decouple the marginal cumulative distribution functions (CDFs) of empirical features from their multivariate dependency structure, creating a controlled sandbox to test out-of-distribution resilience.

As illustrated in Figure 2, the evaluation lifecycle is executed across 10 independent iterations, proceeding through two distinct chronological phases.

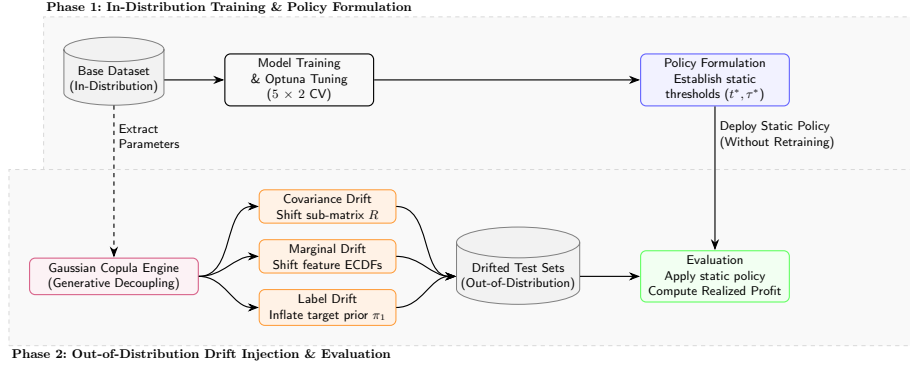


Fig. 2. The longitudinal evaluation lifecycle. Base models and operational policies are strictly optimized and established on the in-distribution dataset. The Gaussian Copula engine subsequently spawns out-of-distribution test sets to benchmark the economic decay of those static policies.

Phase 1: In-Distribution Training and Policy Formulation. First, the copula engine generates an empirical base dataset representing stable, in-distribution historical data. For each base dataset, hyperparameter optimization is conducted across all model architectures via the Tree-structured Parzen Estimator (TPE) [5]. To align with the operational deployment environment, the objective function strictly minimizes the Average Expected Cost (AEC) using a 5×2 cross-validation procedure.

Crucially, the final operational policies are explicitly calculated and established on this base dataset prior to out-of-distribution deployment. For PtO

models, the theoretical Bayes threshold (t^*) is derived from the calibrated baseline probabilities. For PaO models, the empirical thresholds (τ^*) are tuned via cross-validation over the baseline economic scores.

Phase 2: Out-of-Distribution Drift Injection and Evaluation. Once the static operational policies are established, the models are subjected to dataset shift. Utilizing the mathematical decoupling properties of the Gaussian copula, we spawn multiple out-of-distribution test sets from the original base distribution, systematically inducing three specific types of macroeconomic drift:

1. **Covariance Drift (Structural Shock):** We simulate the sudden decoupling of customer usage behaviors, such as shifts to flat-rate pricing models. We hold the marginal distributions constant and isolate the sub-matrix R of highly correlated usage and financial features within the copula’s correlation structure. We apply a Fisher z-transformation to negatively scale the off-diagonal Pearson correlation coefficients by a shift strength of 0.80, utilizing Higham’s alternating projections algorithm to ensure the modified matrix remains positive semi-definite [7].
2. **Marginal Drift (Scale Shock):** We simulate an absolute macroeconomic decline, such as an economic recession causing a global contraction in telecommunications usage. We hold the correlation dependency matrix R entirely constant. Instead, we isolate the continuous usage variables and scale their absolute margins by a shift strength of 0.80, systematically shifting the Empirical CDFs without altering internal feature relationships.
3. **Label Drift (Base-Rate Shock):** We simulate a sudden influx of churners caused by aggressive competitor marketing. We artificially inflate the global prior probability of the target class (π_1) by a factor of 1.20. We resample the test set to reflect this new prior distribution while leaving the conditional feature distributions $P(X|Y)$ mathematically untouched.

To evaluate true economic resilience, the static policies formulated in Phase 1 are deployed directly onto these drifted test sets.

3.5 Evaluation Metrics

To evaluate the true economic resilience of the competing operational policies under adverse dataset shift, our primary metric is the Realized Profit. Subsequently, to empirically disentangle the underlying failure mechanisms driving any observed financial decay, we track specific diagnostic metrics tailored to the mathematical vulnerabilities of both the PtO and PaO approaches.

1. Realized Profit: Ultimately, the success of a proactive churn campaign is measured by its net financial return. Let $\mathcal{T}$ represent the subset of customers selected for intervention by the fixed deployment policy (i.e., customers exceeding the theoretical t^* or the empirical τ^*). For each drifted test set, we simulate a deterministic business outcome by sampling a true underlying campaign acceptance vector $a_i \sim \text{Bernoulli}(0.3)$ for $i \in \mathcal{T}$. The Realized Profit aggregates the net financial gains and wasted expenditures generated across the targeted

portfolio by explicitly separating accepted offers from rejected ones:

$$\text{Realized Profit} = \sum_{i \in \mathcal{T}} [y_i a_i (CLV_i - d_i - f) - y_i (1 - a_i) f - (1 - y_i) C_{FP,i}] \quad (3)$$

While Realized Profit establishes the macroscopic financial impact, we utilize the following three diagnostic metrics to isolate the exact architectural failures driving that profit loss:

2. Brier Score (Calibration Loss): The PtO approach relies on absolute probabilistic accuracy to validate its theoretical Bayes threshold. We compute the Brier Score—the mean squared difference between the predicted probabilities $\hat{p}_i$ and the actual outcomes y_i —to precisely measure the destruction of calibration under drift.

$$BS = \frac{1}{N} \sum_{i=1}^N (\hat{p}_i - y_i)^2 \quad (4)$$

3. Wasserstein Distance (Density Misalignment): The PaO approach relies on its uncalibrated score density remaining stable so the fixed empirical threshold (τ^*) remains valid. We calculate the 1D Wasserstein Distance (Earth Mover’s Distance) between the validation scores S_{val} and the drifted test scores S_{test} . A high Wasserstein distance proves the score distribution has shifted, rendering the static threshold obsolete.

4. Target Volume Drift (ΔV): This metric translates theoretical density shifts into concrete operational terms. Let $\mathcal{T}$ be the subset of targeted customers. ΔV measures the directional percentage-point deviation in the proportion of the customer base selected for intervention between the validation and test phases:

$$\Delta V = \frac{|\mathcal{T}_{test}|}{N_{test}} - \frac{|\mathcal{T}_{val}|}{N_{val}} \quad (5)$$

This metric tracks whether threshold staleness causes a model to erroneously inflate its total campaign volume (a positive ΔV) or collapse its targeted cohort (a negative ΔV).

4 Results

The empirical evaluation highlights a fundamental operational trade-off in cost-sensitive classification: the modeling approach that maximizes initial financial return in-distribution exhibits the highest sensitivity to structural dataset shift.

To establish a clear macroscopic understanding of these decay mechanisms, the initial analysis presents an aggregated view. Specifically, the first two subsections evaluate the overarching performance of the Predict-then-Optimize (PtO) and Predict-and-Optimize (PaO) paradigms by averaging outcomes across all underlying base algorithms (e.g., Logistic Regression, Random Forests, Gradient Boosting) and cost granularities (class-dependent versus instance-dependent optimization). Subsequent sections systematically disentangle these dimensions to isolate the structural robustness of specific algorithmic architectures and the exact impact of cost granularity.

4.1 Predictive Performance and Pipeline Degradation

To establish the baseline differences between the two pipelines, Table 2 reports the average realized profit under in-distribution conditions across all algorithms compared to the three out-of-distribution drift scenarios.

Table 2. Average realized profit in units per customer across operational pipelines. Values in parentheses indicate the absolute drop or gain relative to the baseline.

Pipeline	Baseline	Covariance (0.8)	Marginal (0.8)	Label (1.2)
PtO (Cost-Insensitive)	4.48	2.96 (-1.52)	1.78 (-2.70)	5.17 (+0.69)
PaO (Cost-Sensitive)	3.28	2.92 (-0.36)	2.16 (-1.12)	4.23 (+0.95)

Under strictly in-distribution conditions, the PtO pipeline yields a superior average baseline profit of 4.48 units per customer, outperforming the PaO pipeline (3.28 units). Because PtO relies on proper scoring rules to generate calibrated probabilities, applying theoretical Bayes Minimum Risk thresholds yields mathematically optimal decisions when the test data matches the training distribution.

However, this reliance on exact calibration introduces severe vulnerabilities under distribution shift. When subjected to structural covariance drift (a shift strength of 0.80), PtO models suffer an average profit reduction of 1.52 units. In contrast, PaO models demonstrate substantial robustness, degrading by only 0.36 units. By explicitly integrating asymmetric costs into the loss function during Empirical Risk Minimization, PaO representations prioritize economically vital feature interactions, maintaining their ordinal ranking even when peripheral correlations decouple.

This vulnerability is most pronounced under marginal drift. When the absolute scale of continuous features (e.g., usage metrics) contracts, the degradation of the PtO approach is so severe that its profit drops below that of the PaO approach (1.78 vs. 2.16 units). Conversely, when macroeconomic base-rates change (Label Drift of 1.20), both approaches exhibit an absolute increase in realized profit. It is important to note that this aggregate increase is structurally driven by the evaluation framework: because average profit is normalized across the entire customer base, an artificially inflated proportion of churners inherently raises the total pool of retainable value. Nevertheless, PtO models successfully capture significantly more of this upside. Because their outputs are probabilities, they naturally scale with base-rate influxes, allowing the Bayes threshold to instantly expand the targeted cohort and generate 5.17 units of profit, outpacing the PaO approach.

4.2 Mechanisms of Policy Decay

To isolate why these approaches fail, we track diagnostic metrics tailored to their respective operational policies (Table 3). Because the PaO approach optimizes an improper scoring rule, its outputs are uncalibrated economic scores. Consequently, evaluating it with a Brier score is mathematically invalid. Instead, we evaluate PtO degradation via calibration loss (Brier Score), and PaO degradation via threshold staleness (Wasserstein Distance and Target Volume Drift).

Table 3. Diagnostic metrics isolating the failure mechanisms of each approach. PtO decay is measured via calibration loss (Brier Score). PaO decay is measured via score density shift (Wasserstein Distance) and target volume drift (ΔV).

Drift Scenario	PtO Diagnostics		PaO Diagnostics	
	Base Brier	Drift Brier	Wasserstein	Target Vol. Drift (ΔV)
Covariance (0.8)	0.037	0.076	0.006	0.000
Marginal (0.8)	0.037	0.079	0.015	0.018
Label (1.2)	0.037	0.046	0.015	0.020

Under covariance drift, the source of the PtO approach’s profit collapse is evident: its Brier score more than doubles from 0.037 to 0.076, rendering the theoretical Bayes thresholds invalid. In contrast, the PaO models exhibit a Wasserstein distance of only 0.006, indicating that the uncalibrated score density remained highly stable. This stability results in zero target volume drift ($\Delta V = 0.000$), meaning the campaign size neither erroneously inflated nor collapsed.

Under marginal drift, both approaches exhibit significant density warping. For the PaO approach, this misalignment causes the static empirical threshold to erroneously inflate the campaign volume ($\Delta V = +0.018$), effectively wasting retention budget on false positives and driving the mutual collapse in profitability observed in Table 2. Conversely, under label drift, the positive volume shift ($\Delta V = +0.020$) reflects a structurally valid and necessary expansion of the campaign to accommodate the macroeconomic influx of true churners.

4.3 Algorithmic Variance and Structural Robustness

Disaggregating the approaches reveals that economic robustness depends heavily on the geometric properties of the underlying base algorithms, as visualized in Figure 3.

LightGBM (GBM) exhibits the highest variance. Standard GBM achieves the maximum baseline profit (5.83) but collapses entirely under marginal drift (-2.91 absolute reduction). Cost-Sensitive Boosting (CSBoost) mitigates this volatility, trading peak baseline performance to restrict losses under covariance drift (reducing the drop from -2.58 to -0.85).

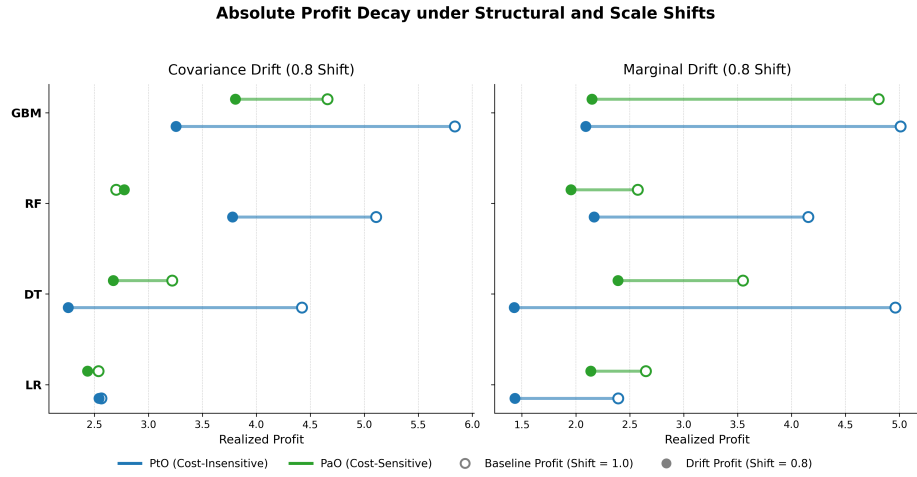


Fig. 3. Absolute profit decay under negative dataset shift across base algorithms. Hollow markers denote baseline profitability, while solid markers denote realized profit under shift. The length of the line visualizes the severity of the degradation.

Cost-Sensitive Random Forests (CSForest) exhibit the highest resistance to structural shock. While a standard Random Forest’s profit drops by -1.33 under covariance drift, CSForest maintains consistent performance, effectively neutralizing the impact of feature decoupling. Conversely, standard Decision Trees produce brittle step-function probabilities that are highly sensitive to absolute feature scaling, resulting in a severe profit reduction (-3.53) under marginal drift. Implementing the Cost-Sensitive Tree (CSTree) resolves this vulnerability by adjusting the splitting criterion, mitigating the loss to -1.15 . This heightened resilience across both CSForest and CSTree can likely be attributed to their architectural divergence from gradient-based methods. While CSBoost penalizes misclassifications by altering continuous gradients, the tree-based models fundamentally alter the recursive partitioning process by replacing standard metrics (such as Gini impurity) with a direct cost impurity measure. This discrete, non-gradient optimization mechanism appears structurally more robust to the continuous spatial warping induced by dataset shift.

4.4 Impact of Cost Granularity: Class vs. Instance Optimization

The analysis identifies a significant interaction effect regarding the granularity of the cost parameters utilized during model training (Figure 4).

During scale-based shocks (Marginal Drift), Instance-Dependent optimization demonstrates clear superiority. Relying on average class costs caused algorithms to disproportionately drop customers below the operational threshold as raw usage metrics contracted. Integrating the exact, individualized Customer Lifetime Value (CLV) acts as a mathematical anchor. The elevated gradient

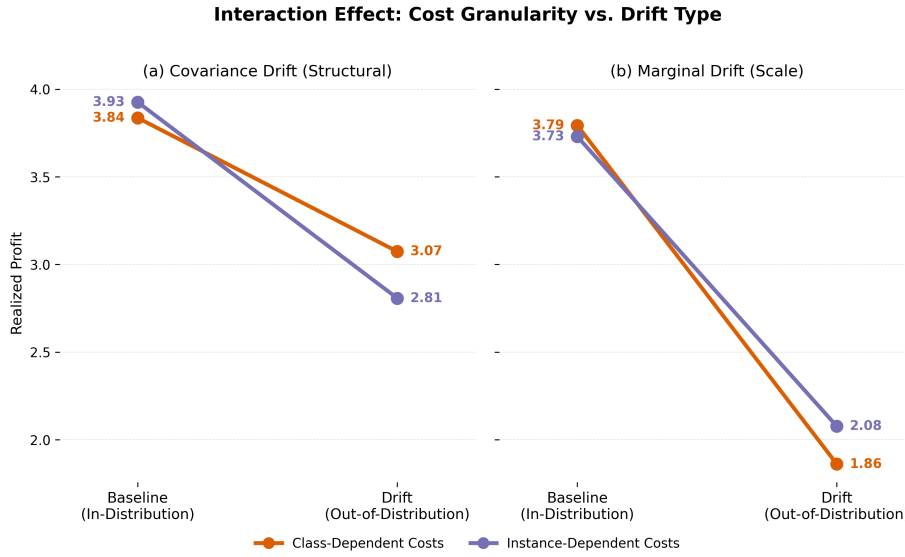


Fig. 4. The interaction effect between cost granularity and drift type. Panel (a) demonstrates the stabilizing effect of Class-Dependent costs under Covariance drift. Panel (b) illustrates the robustness provided by Instance-Dependent costs under Marginal drift.

penalty for misclassifying a high-value customer forces the model to maintain targeted interventions, systematically reducing financial losses compared to Class-Dependent models.

However, during structural shocks (Covariance Drift), Class-Dependent optimization yields greater stability for high-variance architectures such as Trees and Forests. When internal feature correlations decouple, optimizing for hyper-specific individual CLVs introduces excess variance, causing the algorithms to overfit to the disrupted feature space. In this context, utilizing a global average cost provides a macro-level smoothing effect that restricts the algorithm from capturing spurious interactions, thereby preserving out-of-distribution profitability.

5 Discussion

The results of this study recontextualize the consensus surrounding cost-sensitive classification. Recent large-scale benchmarking, such as Vanderschueren et al. (2022) [16], concludes that the Predict-then-Optimize (PtO) approach is the universally dominant strategy, positing that task-specific loss functions (Predict-and-Optimize) do not yield better downstream decisions.

While our baseline results align with this in-distribution supremacy, our longitudinal evaluation reveals a critical operational blind spot. PtO’s profitability is strictly contingent upon perfect probabilistic calibration—a property highly

susceptible to environmental decay. Under structural dataset shifts, the prerequisite for PtO’s success (the validity of the theoretical Bayes threshold) collapses. Consequently, the true value of the PaO approach lies not in maximizing in-distribution returns, but in establishing a robust, uncalibrated score density that defends against structural uncertainty.

5.1 Revisiting Class vs. Instance Optimization

Prior literature strongly advocates integrating instance-dependent rather than class-dependent costs to minimize overall expected costs [16]. Our findings validate but significantly amend this recommendation depending on the geometry of the environmental shock.

Under scale-based shocks (Marginal or Label Drift), instance-dependent optimization is superior. The individualized Customer Lifetime Value (CLV) provides a mathematical anchor; the heightened gradient penalties associated with misclassifying high-value individuals force the model to maintain targeted interventions even as overall usage metrics systematically decline.

Conversely, under structural shocks (Covariance Drift), we identify a critical divergence from the established literature. When historical feature correlations decouple, optimizing for hyper-specific individual CLVs introduces excess variance, causing high-capacity algorithms to overfit the disrupted feature space. In these highly volatile environments, class-dependent costs prove superior. It provides a macro-level smoothing effect that restricts algorithms from capturing spurious interactions within decoupled feature sets.

5.2 Strategic Implications for Model Deployment

Selecting an operational approach must transcend static performance benchmarking and instead align with the anticipated volatility of the deployment environment. For instance, when external market dynamics—such as a competitor launching an aggressive acquisition campaign—alter target class prevalence without fundamentally changing conditional feature distributions, PtO architectures maintain a distinct advantage. Because these models output proper probabilities, they naturally scale with shifting priors. This mathematical property allows the Bayes Minimum Risk threshold to dynamically expand the targeted cohort to capture the macroeconomic influx of true churners. Conversely, PaO models rely on static empirical thresholds that artificially constrain intervention volumes under base-rate inflation, resulting in substantial forfeited revenue.

However, the probabilistic calibration that makes PtO models resilient to base-rate fluctuations becomes a severe liability when the environment undergoes structural disruptions. If an industry-wide transition, such as moving from pay-per-use to flat-rate subscription models, breaks the historical correlations between predictive features, PtO calibration severely degrades. To preserve profitability under this covariance drift, practitioners must shift toward PaO approaches utilizing Cost-Sensitive Ensembles (e.g., CSForest). Furthermore, applying class-dependent costs during training provides a necessary macro-level

regularization effect, preventing high-capacity algorithms from overfitting to the newly decoupled feature space.

This regularization strategy must be deliberately inverted when the environmental shock manifests as a scale contraction rather than feature decoupling. When populations experience systematic contractions in absolute feature scales—such as a macroeconomic downturn causing widespread reductions in discretionary service usage—standard models generate artificially suppressed risk scores, effectively halting proactive campaigns. Enterprises must counter this marginal drift by deploying PaO models trained specifically with instance-dependent costs. The individualized financial penalty functions as an operational safeguard, forcing the algorithm to continue targeting high-value customers despite the macroscopic decay in their behavioral metrics.

6 Conclusion

This research demonstrates that the evaluation of cost-sensitive classifiers must extend beyond static, in-distribution performance metrics to encompass structural robustness over time. By rigorously decoupling dataset shift into marginal, covariance, and label drift utilizing a continuous Gaussian copula architecture, we map the distinct failure mechanisms of the two dominant operational approaches.

Our findings confirm that there is no universally dominant strategy for cost-sensitive classification. The Predict-then-Optimize paradigm maximizes potential financial return under stable conditions but suffers severe economic degradation when its probabilistic calibration is compromised by structural or scale-based shocks. Conversely, the Predict-and-Optimize paradigm sacrifices peak baseline performance to establish a resilient, uncalibrated score density that withstands feature decoupling. Ultimately, this research provides a strategic framework to guide practitioners in selecting the optimal modeling approach based on anticipated changes in their competitive environment. Rather than relying on a singular methodology, organizations can proactively safeguard their retention campaigns by deliberately aligning the algorithmic properties and cost-granularity of their classifiers with the specific macroeconomic and behavioral shifts anticipated in production.

7 Limitations and Future Work

While this study isolates the mechanisms of policy decay by evaluating static operational policies, it inherently assumes a worst-case deployment scenario where models remain unmonitored. In mature operational environments, thresholds are periodically audited. Future research should investigate the financial viability of dynamic thresholding techniques under dataset shift—specifically, determining whether continuously updating the empirical threshold (τ^*) of a PaO model using recent batches of unlabelled data can successfully restore profitability without necessitating complete model retraining. Additionally, extending this

experimental framework to other financial domains characterized by asymmetric misclassification costs, such as credit scoring and fraud detection, warrants further empirical investigation to establish the generalizability of these findings.

References

1. Iranian Churn. UCI Machine Learning Repository (2020), DOI: <https://doi.org/10.24432/C5JW3Z>
2. Bahnsen, A.C., Aouada, D., Ottersten, B.: Example-dependent cost-sensitive decision trees. *Expert Systems with Applications* **42**(19), 6609–6619 (2015)
3. Bahnsen, A.C., Aouada, D., Ottersten, B.: A novel cost-sensitive framework for customer churn predictive modeling. *Decision Analytics* **2**(1), 5 (2015)
4. Bahnsen, A.C., Stojanovic, A., Aouada, D., Ottersten, B.: Cost sensitive credit card fraud detection using bayes minimum risk. In: 2013 12th international conference on machine learning and applications. vol. 1, pp. 333–338. IEEE (2013)
5. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. *Advances in neural information processing systems* **24** (2011)
6. Hand, D.J.: Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine learning* **77**(1), 103–123 (2009)
7. Higham, N.J.: Computing the nearest correlation matrix—a problem from finance. *IMA journal of Numerical Analysis* **22**(3), 329–343 (2002)
8. Höppner, S., Baesens, B., Verbeke, W., Verdonck, T.: Instance-dependent cost-sensitive learning for detecting transfer fraud. *European Journal of Operational Research* **297**(1), 291–300 (2022)
9. Höppner, S., Stripling, E., Baesens, B., vanden Broucke, S., Verdonck, T.: Profit driven decision trees for churn prediction. *European journal of operational research* **284**(3), 920–933 (2020)
10. Jahromi, A.T., Stakhovych, S., Ewing, M.: Managing b2b customer churn, retention and profitability. *Industrial Marketing Management* **43**(7), 1258–1268 (2014)
11. Janssens, B., Bogaert, M., Bague, A., Van den Poel, D.: B2boost: Instance-dependent profit-driven modelling of b2b churn. *Annals of Operations Research* **341**(1), 267–293 (2024)
12. Nelsen, R.B.: An introduction to copulas. Springer (2006)
13. Patki, N., Wedge, R., Veeramachaneni, K.: The synthetic data vault. In: 2016 IEEE international conference on data science and advanced analytics (DSAA). pp. 399–410. IEEE (2016)
14. Risselada, H., Verhoef, P.C., Bijmolt, T.H.: Staying power of churn prediction models. *Journal of Interactive Marketing* **24**(3), 198–208 (2010)
15. Stripling, E., vanden Broucke, S., Antonio, K., Baesens, B., Snoeck, M.: Profit maximizing logistic model for customer churn prediction using genetic algorithms. *Swarm and Evolutionary Computation* **40**, 116–130 (2018)
16. Vanderschueren, T., Verdonck, T., Baesens, B., Verbeke, W.: Predict-then-optimize or predict-and-optimize? an empirical evaluation of cost-sensitive learning strategies. *Information sciences* **594**, 400–415 (2022)
17. Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B.: New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European journal of operational research* **218**(1), 211–229 (2012)
18. Verbraken, T., Verbeke, W., Baesens, B.: A novel profit maximizing metric for measuring classification performance of customer churn prediction models. *IEEE transactions on knowledge and data engineering* **25**(5), 961–973 (2012)

Dataset Shifts in Temporal Quantification

João Pedro Ortega¹, Luiz Fernando Luth Junior¹, Daniel Ojeda¹,
André Maletzke¹, and Mirko Bunse²

¹ Western Parana State University, Foz do Iguaçu, Brazil

{joao.ortega1,luiz.junior90,daniel.ojeda,andre.maletzke}@unioeste.br

² Lamarr Institute for Machine Learning and Artificial Intelligence,
TU Dortmund University, 44227 Dortmund, Germany

mirko.bunse@cs.tu-dortmund.de

Abstract. Quantification aims to estimate class distributions in unlabeled data. When data are observed over time, this task becomes more challenging, as temporal processes may induce systematic forms of dataset shift. Existing temporal quantifiers rely on forecasting-based adjustments, such as moving averages or Kalman filters, applied to the outputs of standard quantifiers. Although effective in some scenarios, these methods do not explicitly characterize the type of temporal shift affecting the data. In this paper, we study temporal quantification through the lens of time-dependent dataset shift. We introduce time as an explicit random variable and use Bayesian networks to distinguish different temporal shift mechanisms. This formalization explains when existing forecasting-based adjustments are expected to perform well and when they are not. We propose a temporal extension to distribution matching, show that our proposal is Fisher consistent under temporal joint shift, and instantiate it with T-PACC, a temporal variant of Probabilistic Adjusted Classify & Count. Experiments on synthetic and real-world datasets show that forecasting-based adjustments excel when standard quantifiers are consistent across time, while T-PACC achieves superior performance when the relation between features and labels changes. Our results indicate that temporal quantification requires methods aligned with the specific temporal shift mechanism that underlies the data.

Keywords: quantification · temporal joint shift · temporal dataset shift.

1 Introduction

Quantification estimates the distributions of classes in unlabeled data sets based on models that are trained from labeled data [3,6]. A central challenge in quantification arises from *dataset shift*, that is, some discrepancy between the data used for training and the data encountered during deployment. Dataset shift can be categorized into several types based on a distinct set of properties that remain invariant between training and deployment [12,16,20]. These invariances can be exploited to learn quantifiers that transfer reliably from the training distribution to various deployment distributions [22].

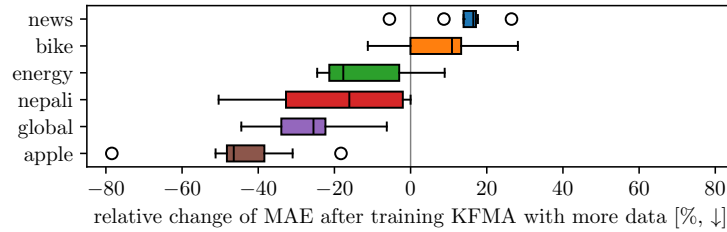


Fig. 1. Relative change in mean absolute prediction error (MAE) on the test set after doubling each quantifier’s training data. Each box summarizes multiple quantifiers using KFMA’s temporal adjustment [13]. Values below zero indicate successful learning; values above zero indicate a poor transfer between training and testing distributions.

The challenge of dataset shift becomes particularly pronounced when data are observed over time, being affected by long-term trends, periodical variations, or external events. In this setting, which we refer to as *temporal quantification*, time introduces additional structure beyond the standard quantification setting: observations are embedded in temporal processes that induce systematic, time-dependent forms of dataset shift. Monitoring class distributions over time is relevant in diverse application areas such as the social sciences [11], ocean monitoring [10], and public health [19].

The state-of-the-art (SOTA) in temporal quantification is to adjust the outputs of non-temporal quantifiers through time series forecasting, as recently studied by Li et al. [13]. Their systematical evaluation of various quantifiers and time series forecasters, given a fixed split of the data, finds the overall best performance in classification-based and label shift-consistent quantifiers that are paired with a Kalman Filter Moving Average (KFMA) forecaster.

We ask: how well does the SOTA actually transfer from the training distribution to the testing distributions? To answer this question, we have re-run the benchmark by Li et al. [13] with a slight modification: instead of using the original split of the data, we repeat the experiment using twice the amount of data for training each quantifier. If the quantifier transfers robustly, doubling the training data should lead to a reduction in the prediction error on the test set. However, Fig. 1 shows that the observed changes in prediction error vary substantially across datasets; in 2 out of 6 datasets, the SOTA exhibits a substantial *increase* of the prediction error, reflecting poor transferability.

In this work, we attribute this variability across datasets to the presence of different types of time-dependent dataset shift. We define these novel types of shift through Bayesian networks [18], extending previous definitions [12] with temporal dependencies. On this basis, we propose Fisher-consistent methods [21] for temporal quantification. By associating the datasets with suitable shift types, we can select the most transferable quantifier to minimize prediction errors. Our findings reveal that current methods such as KFMA are designed to address only specific shift types, promising high performance in some temporal settings but limiting their applicability in others.

2 Background and Related Work

Our temporal analysis builds on several findings from prior work on quantification and dataset shift in non-temporal settings. We begin by introducing Bayesian networks as a general tool for characterizing different types of dataset shift, with particular attention to the case of label shift and including an original proof of Fisher consistency for distribution matching quantifiers. Finally, we review previous work on temporal quantification.

2.1 Bayesian Networks and Dataset Shift

Bayesian networks are probabilistic graphical models that encode statements on conditional independence between random variables. They are a convenient means of expressing assumptions about the data, particularly those that concern potential dependencies and independencies of features and labels from some given *context*. In the simplest case, there is only one training context c_{trn} and one testing context c_{tst} [12] but quantification usually deals with multiple testing contexts $\{c_i\}_{i=1}^n$ that are defined by the respective application area. A product sentiment quantifier, for instance, is trained with some text corpus and tasked with predicting the distribution of sentiment towards several products [8]; a plankton quantifier is trained with some microscope images and tasked with predicting the distribution of plankton species within several water samples [10]. Here, each product or water sample defines one prediction context.

Since the context is defined by the respective object of study, practitioners are naturally interested in the class distribution $\mathbb{P}(Y \mid C = c)$ within each context. Quantification tries to accurately estimate this class distribution based on a corresponding set of unlabeled data $\{x_i \sim \mathbb{P}(X \mid C = c)\}_{i=1}^n$, where X and Y have some kind of relation. We emphasize that their relation might change across contexts, complicating the estimation of $\mathbb{P}(Y \mid C = c)$ from data.

Bayesian networks can represent our assumptions about potential changes between contexts. Let us consider a network with vertices $\{X, Y, C\}$ and only one edge, be it either $X \rightarrow Y$ or $Y \rightarrow X$. Since this network does not include any outgoing edge from C , it asserts that the context C does not affect either the features X or the labels Y , i.e., $\mathbb{P}(X, Y \mid C) = \mathbb{P}(X, Y)$. In fact, this Bayesian network represents the complete (and desirable) *absence of dataset shift*, where quantification would be trivial. Quantification becomes challenging once dataset shift happens, i.e., once C has outgoing edges.

2.2 Label Shift

The most common assumption in quantification research is the assumption of *label shift*, also known as prior probability shift. Here, C has an outgoing edge to Y , expressing that the class distribution can change between contexts.

Fig. 2 illustrates this assumption as a Bayesian network. It also contains the edge $Y \rightarrow X$, encoding the conditional independence

$$\mathbb{P}(X \mid Y, C) = \mathbb{P}(X \mid Y) \quad (1)$$



Fig. 2. Bayesian network representing label shift, as of Kull and Flach [12]. The context $C \in \{\text{training}, \text{testing}_1, \text{testing}_2, \dots\}$ determines the distribution of class labels Y , which, in turn, determines the distribution of features X .

meaning that C has no implication on X given Y [12,18]. Most importantly, this independence means that $\mathbb{P}(X \mid Y, C = c_i) = \mathbb{P}(X \mid Y, C = c_j) \forall (i, j)$, that is, any knowledge on $\mathbb{P}(X \mid Y)$ can be *transferred between contexts*.

We find it worthwhile to relate the notation from Eq. 1 to a different notation that other researchers have used to represent label shift [9]. This other notation differs between two probability distributions P and Q , with P representing the training distribution and Q representing the distribution of the current testing context (or vice versa). In this notation, Eq. 1 is $Q(X \mid Y) = P(X \mid Y)$, emphasizing that the class-conditional distributions are identical in training and testing. This distinction between P and Q is determined by the context C : specifically, $P = \mathbb{P}(\cdot \mid C = c_{\text{trn}})$ and $Q = \mathbb{P}(\cdot \mid C = c_{\text{tst}})$ [12]. The advantage of our context-based notation is a larger convenience in discussing more than just two contexts, as we do later in this paper.

2.3 Quantification Under Label Shift

Distribution matching is a broad family of quantification methods that explicitly leverage the independence of $\mathbb{P}(X \mid Y)$ from C . They represent all data through some feature mapping $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ (where possibly $\mathcal{Z} \subseteq \mathbb{R}^d$), such that $\hat{\Phi}(X \mid C = c) = \frac{1}{m} \sum_{i=1}^m \Phi(x_i)$ represents the data for which the class distribution is being predicted and $\hat{\Phi}(X \mid Y = y) = \frac{1}{|D_y|} \sum_{x \in D_y} \Phi(x)$ represents the training data D_y of every class $y \in \mathcal{Y}$. Given any such mapping, a prediction of the class distribution is found by minimizing some loss $\ell : \mathcal{Z} \times \mathcal{Z} \rightarrow \mathbb{R}$ between $\hat{\Phi}(X \mid C = c)$ and the prediction-weighted sum of all $\hat{\Phi}(X \mid Y = y)$, i.e.,

$$\hat{\mathbb{P}}(Y \mid C=c) = \arg \min_{\mathbb{P}} \ell \left(\hat{\Phi}(X \mid C=c), \sum_{y \in \mathcal{Y}} \hat{\Phi}(X \mid Y=y) \cdot \mathbb{P}(Y=y \mid C=c) \right) \quad (2)$$

where Φ and ℓ admit many possible definitions. In fact, any choice of feature map and loss function yields a different distribution matching method [2,5].

2.4 Fisher Consistency

Fisher consistency states that a quantification method would recover the true class distribution $\mathbb{P}(Y \mid C = c)$ if it were applied not to a finite sample but to the unlabeled distribution $\mathbb{P}(X \mid C = c)$ itself [21]. Verifying this property for a finite-sample estimator requires analyzing its *population analogue*—its hypothetical behavior when working directly on populations $\mathbb{P}$ rather than finite samples

$\{x_i \sim \mathbb{P}\}_{i=1}^m$. Fortunately, distribution matching is already designed at the population level to begin with; finite-sample estimates are only derived from these definitions so that the population analogue is available by construction.

Definition 1 (Fisher Consistency). *Let h be the population analogue of some quantifier. We say that this quantifier is Fisher consistent for a family of distributions $\mathcal{P}$ if and only if, for all $\mathbb{P} \in \mathcal{P}$ and all contexts $c \in \mathcal{C}$,*

$$h(\mathbb{P}(X \mid C = c)) = \mathbb{P}(Y \mid C = c)$$

While this definition is applicable to any type of shift, each corresponding to a family of distributions $\mathcal{P}$, we now establish distribution matching to be Fisher consistent under label shift. The following result is an original contribution of ours. All proofs are given in Appendix A.

Theorem 1 (Fisher Consistency Under Label Shift). *Consider a distribution matching quantifier, as defined by Eq. 2. Under the identifiability condition that, for all $\alpha \in \mathbb{R}^{|\mathcal{Y}|}$,*

$$\sum_{y \in \mathcal{Y}} \Phi(X \mid Y = y) \cdot \alpha_y = 0 \quad \Rightarrow \quad \alpha_y = 0 \quad \forall y \in \mathcal{Y} \quad (3)$$

this quantifier is Fisher consistent under label shift.

2.5 Quantification Over Time

Monitoring temporal changes in class distributions has already been envisioned in the seminal paper by Forman [6], which introduced quantification as a machine learning task in its own right. Only recently has the topic been picked up systematically. In their work “Quantification over Time” [13], Li, Gharakheili, and Batista establish a framework for temporal adjustments of quantifier predictions through time series forecasting. Their framework estimates the class distribution at the current time step by combining two sources of information: the prediction from a standard quantifier and a time-series forecaster based on previous estimates. The best-performing method in this framework, KFMA, employs a moving-average forecaster and a Kalman filter that dynamically weights the forecast against the current quantifier output.

Prior work on temporal quantification largely falls into two patterns within the above framework. One line applies a quantifier independently to each time period. Hopkins and King [11] for instance, estimate the distribution of political opinions over time with a method designed to correct the bias of classify and count approaches, but the individual time periods are still quantified without an explicit temporal adjustment. A second line adjusts predictions of quantifiers through moving averages. O’Connor et al. [17] derives daily sentiment ratios by counting lexicon-based positive and negative Twitter messages, and by smoothing these ratios with moving averages to compare them with pooling time series. Wen et al. [23] similarly compute daily ratios of positive to negative terms in online course discussions forums and use a three-day moving average to monitor students’ opinions over time.

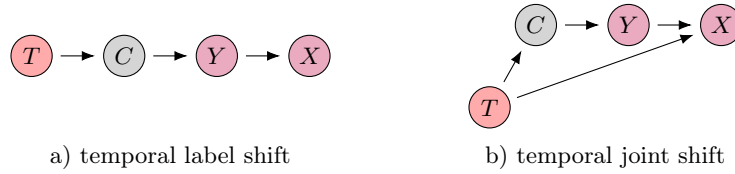


Fig. 3. Bayesian networks of temporal shift types in which the time T determines the context C , the context determines the class label Y , and the class label determines the features X . These shift types differ in the impact of T on X , with substantial implications on addressing temporal quantification: (a) T can only affect X indirectly, through C and Y ; (b) T has a direct influence on X .

Unfortunately, the circumstances under which any of the above methods thrives or fails remain unclear. This is because the literature on temporal quantification is still lacking a characterization of time-dependent dataset shift and corresponding consistency guarantees.

3 Proposal

We now introduce time as an additional random variable T . Its relation to context, labels, and features can take different forms, corresponding to different types of time-dependent dataset shift. We define these types of shift through Bayesian networks and analyze their implications on data distributions and consistency. Most importantly, our analysis reveals novel methods for quantification under what we call “temporal joint shift”.

Our Bayesian networks formalize a common assumption in prior work on temporal quantification: that the time T determines the context C . To see that this assumption is ubiquitous, note that (i) Li et al. [13] employ the first n time steps for training and predict the class distribution in each of the remaining time steps separately; (ii) Hopkins and King [11] also estimate the class distribution separately in each time step; (iii) Maletzke et al. [14] consider a temporal framework that estimates class distributions under the assumption of discrete and well-defined contexts; and (iv) Forman [6] lists, as an example use case of quantification, estimating “the number of news articles about terrorism each month”. In each of these examples, every testing and training context spans a separate range of time, such that $T \rightarrow C$.

Fig. 3 presents Bayesian networks for the case where $Y \rightarrow X$, another prominent assumption in temporal and non-temporal quantification. We detail the implications of these two networks and develop methods for handling them in subsections 3.1 and 3.2. The opposite $X \rightarrow Y$ setting is analyzed in Sec. 3.3.

3.1 Temporal Label Shift

Fig. 3.a) defines a minimal extension of label shift with the additional, temporal dependency $T \rightarrow C$. This extension comprises the conditional independences

$$\mathbb{P}(X | Y, C, T) = \mathbb{P}(X | Y) \quad (4)$$

$$\mathbb{P}(Y | C, T) = \mathbb{P}(Y | C) \quad (5)$$

where Eq. 4 makes $\mathbb{P}(X | Y)$ independent from both context and time. Hence, we can safely transfer this conditional probability from a training context to all other contexts—just as in non-temporal label shift. Indeed, Eqs. 4 and 5 allow us to draw a conclusion of a high practical value: from the perspective of consistency, temporal label shift can be solved in precisely the same way as regular (i.e., non-temporal) label shift, without further temporal adjustments.

Corollary 1 (Fisher Consistency Under Temporal Label Shift). *Consider a non-temporal distribution matching quantifier, as defined by Eq. 2. Under the identifiability condition of Eq. 3, this quantifier is also Fisher consistent under temporal label shift.*

On the surface, this result seems to challenge the use of post-prediction adjustments that are based on time series forecasting [13] under temporal label shift; after all, it states that no such adjustment is necessary for establishing consistency. However, it also states that the underlying base quantifiers, which are being adjusted, provide a functional basis for successful time series forecasting. On small data volumes, the forecasting component keeps the predictions aligned with the prior assumption that two consecutive points in time have similar class distributions, thereby reducing prediction error. On larger data volumes, the impact of the forecasting component can be reduced through a hyper-parameter, to establish consistency. In this sense, we understand forecasting adjustments like moving averages and Kalman filters as an ideal approach to the *regularization* of quantifier predictions under temporal label shift. Sec. 4 will support this understanding by revealing that forecasting adjustments excel precisely on those datasets where standard quantifiers exhibit consistent behavior.

Hopkins and King [11] have already leveraged the fact that standard quantifiers are consistent under temporal label shift, albeit without further discussion.

3.2 Temporal Joint Shift

Fig. 3.b) introduces yet another dependency, $T \rightarrow X$, through which the relation between features and labels can shift over time. Hence, we call this network *temporal joint shift*. It comprises the conditional independency from Eq. 5 and

$$\mathbb{P}(X | Y, C, T) = \mathbb{P}(X | Y, T) \quad (6)$$

which implies that Eq. 1—representing a temporally static relation between X and Y —no longer holds in general. Without this equation, we cannot expect a standard quantifier to be consistent. This inadequacy marks a central distinction between temporal joint shift and temporal label shift. However, Eq. 6 also hints at a suitable approach for quantification: to learn a time-dependent relation between X and Y , instead of a static one.

Following this hint, we propose an extension of standard distribution matching for temporal joint shift. We further condition the class-wise representations on time, replacing the static $\hat{\Phi}(X | Y = y)$ from Eq. 2 with some context-dependent $\hat{\Phi}(X | Y = y, C = c)$ computed for the time spanned by each context. This replacement yields the new estimate

$$\hat{\mathbb{P}}(Y|C=c) = \arg \min_{\mathbb{P}} \ell \left(\hat{\Phi}(X|C=c), \sum_{y \in \mathcal{Y}} \hat{\Phi}(X|Y=y, C=c) \cdot \mathbb{P}(Y=y|C=c) \right) \quad (7)$$

where

$$\hat{\Phi}(X | Y=y, C=c) = \frac{1}{m} \sum_{i=1}^m \hat{\Phi}(X | Y=y, T=t_i) \quad (8)$$

is computed over all time points t_i that are contained in the current context c . For this computation, we need to learn the function $\hat{\Phi}(X | Y, T)$, which returns a vector in $\mathcal{Z}$ for any $t \in \mathcal{T}$ and $y \in \mathcal{Y}$. We establish the following.

Theorem 2 (Fisher Consistency Under Temporal Joint Shift). *Consider a temporal distribution matching quantifier, as defined by Eqs. 7 and 8. Under the identifiability condition that, for all $\alpha \in \mathbb{R}^{|\mathcal{Y}|}$ and for all $c \in \mathcal{C}$,*

$$\sum_{y \in \mathcal{Y}} \Phi(X | Y = y, C = c) \cdot \alpha_y = 0 \quad \Rightarrow \quad \alpha_y = 0 \quad \forall y \in \mathcal{Y} \quad (9)$$

this quantifier is Fisher consistent under temporal joint shift.

T-PACC As an example of temporal distribution matching, consider a temporal variant of Probabilistic Adjusted Classify & Count [1], which we call T-PACC. Like the original method, T-PACC employs the probabilistic outputs of a classifier as a feature mapping Φ and the mean squared error as a loss function ℓ . Unlike the original, it does not just learn a static representation $\hat{\Phi}(X | Y)$ but a time-dependent one, $\hat{\Phi}(X | Y, T)$, that is used in Eq. 7.

We emphasize that any other distribution matching method can be adapted for temporal joint shift, by modeling the given representation as being time-dependent. Further details on T-PACC are given in Appendix B.

3.3 Temporal Covariate Shift

Fig. 4 presents two additional networks of temporal dataset shift that extend *covariate shift* [12], a shift with edges $C \rightarrow X \rightarrow Y$. With the inclusion of additional edges from T , we therefore refer to *temporal covariate shift*.

The first of these two cases, defined as $T \rightarrow C \rightarrow X \rightarrow Y$ bears similarity to the case of temporal label shift we discussed in Sec. 3.1: since time determines only the context, it does not have any further implications on $\mathbb{P}(Y | X)$ or $\mathbb{P}(X | C)$. Therefore, we can expect any consistency result on regular covariate shift to also hold under temporal covariate shift.

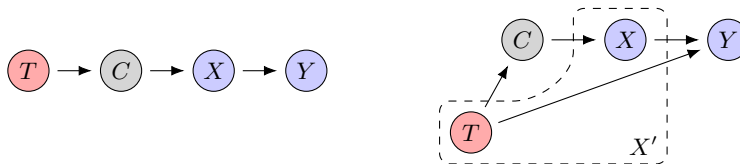


Fig. 4. Bayesian networks of temporal dataset shift when X determines Y . Both of these cases reduce to regular, non-temporal covariate shift.

In the second case, we define an extended feature set $X' = (X, T)$. Once we include time as an additional feature, we have again established a situation where the posterior distribution $\mathbb{P}(Y \mid X')$ is independent of the context. Therefore, once using the extended feature set X' , also the second case of temporal covariate shift reduces to regular, non-temporal covariate shift. In summary, neither of the two cases of temporal covariate shift requires a special treatment of time on the quantification level. In both cases, covariate shift-consistent quantifiers [22] should be used, like the un-adjusted Probabilistic Classify & Count [1] method.

4 Experiments

In the following, we explore the types of dataset shift that most likely affect the latest benchmark data [13] and we generate synthetic datasets that simulate the various types of temporal shifts. Finally, we empirically evaluate our proposed methods in comparison to the SOTA in temporal quantification.

4.1 Shift Types in Benchmark Data

Already in the beginning of this paper, Fig. 1 revealed that the SOTA temporal quantifier, KFMA, exhibits different behaviors across datasets: doubling the size of the training set sometimes decreases and sometimes increases the prediction error of KFMA. An increase in prediction error is observed in 2 of the 6 evaluated datasets in Li et al. [13] research:

Bike³ Hourly records of bike rentals from the Capital Bikeshare system over two years, described by 13 weather and calendar features. The binary target used by Li et al. [13] is whether the rental count is high or low.

News⁴ Statistics on Mashable articles over two years, described by 58 features covering content statistics, keyword and reference metadata, and attributes derived through natural language processing. The binary target used by Li et al. [13] is whether the number of article shares is high or low.

We now attempt to explain the irregular behavior of these datasets by characterizing the shift types that all 6 datasets potentially exhibit.

³ <https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset>

⁴ <https://archive.ics.uci.edu/dataset/332/online+news+popularity>

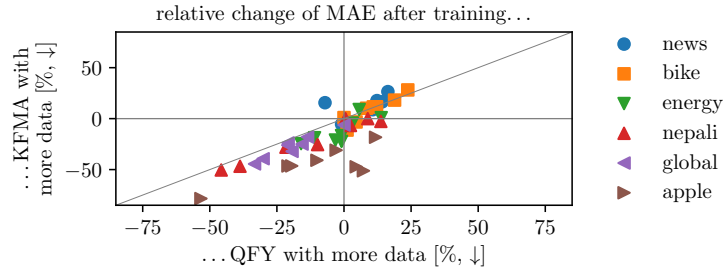


Fig. 5. Testing MAE changes when doubling the training data. Each mark shows one KFMA quantifier: its MAE change (y-axis) versus the MAE change of its unadjusted quantifier (x-axis).

Hypothesis H1—Where the State of the Art Excels Since KFMA adjusts the predictions of various baseline quantifiers, we can relate its changes in prediction errors to the changes that the corresponding un-adjusted baselines exhibit as a result of the enlarged training data. This relation is informative about the underlying shift because we can expect two things. First, in case of label shift (be it temporal or non-temporal), even a standard quantifier should reduce its error as a result of the enlarged training set. This expectation is justified through the consistency guarantees established by Theorem 1 and Corollary 1. Second, a temporal adjustment like KFMA should reduce the baseline’s prediction error even further if the label shift is temporal.

Fig. 5 establishes this relation between baseline and KFMA-adjusted change in prediction error. It becomes apparent that the change in baseline performance is an excellent predictor of the change in KFMA’s performance: if the baseline benefits from the enlarged training data, then KFMA will benefit even more. This behavior is perfectly compatible with temporal label shift. It is observed on all four datasets where the SOTA excels.

Hypothesis H2—Where the State of the Art Fails For the News and Bike datasets, we hypothesize the presence of some dataset shift other than label shift. To test our hypothesis, we take a deeper look into the time-dependent relations between X and Y that both datasets exhibit. Our particular attention is on $\mathbb{P}(\hat{Y} | Y = y, C = c)$, the evolution of a classifier’s misclassification rates across contexts and, hence, over time. We emphasize that PACC and T-PACC use precisely these misclassification rates as their data representation, i.e., $\Phi(X | \cdot) = \mathbb{P}(\hat{Y} | \cdot)$ for some classifier $\hat{Y} = \phi(X)$, and that standard quantifiers assume $\mathbb{P}(\hat{Y} | Y = y, C = c)$ to be independent of the context.

Fig. 6 presents the results of this investigation, showing a substantial periodicity of the Bike rental data over one year. This behavior is plausible for the use case it covers: since the classifier $\hat{Y}$ is trained only on the first few months of data, it is likely to fail in the other seasons, where bikes are rented under different circumstances, and likely to recover in the first months of the next year. Since the class distribution also changes over time (not plotted here), the behavior of

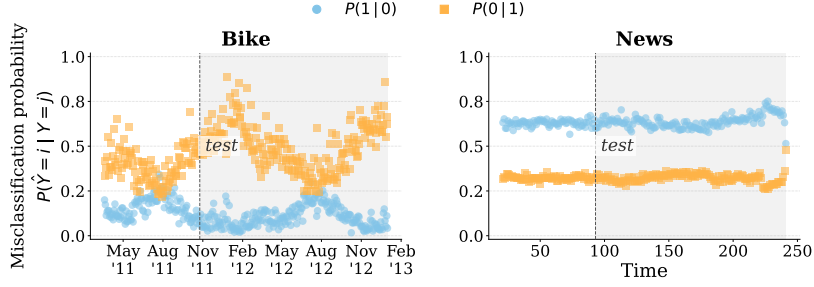


Fig. 6. Temporal evolution of class-conditional misclassification probabilities produced by a logistic regression classifier for the Bike and News datasets. The blue series shows $\mathbb{P}(\hat{Y}=1 \mid Y=0, C=c)$ and the orange series shows $\mathbb{P}(\hat{Y}=0 \mid Y=1, C=c)$ for contexts c that correspond to time steps. While at least the Bike series indicate a relation between features and labels that varies over time, regular non-temporal quantifiers assume static misclassification probabilities $\mathbb{P}(\hat{Y} \mid Y)$ instead, which are invariant across time. The vertical dashed line separates the training phase from the testing phase.

the Bike data is most compatible with temporal joint shift, suggesting the use of our temporal distribution matching framework.

The News data behaves differently, showing roughly constant misclassification probabilities. Therefore, we cannot assert a strong presence of temporal joint shift, leaving open (after ruling out the presence of label shift during the discussion of Fig. 5) the possibility of covariate shift. We therefore hypothesize that the News data most likely benefits from some covariate shift-robust method like PCC [22]. Looking at the classifier’s learned parameters, we see that the most important features are previous shares of an article’s keywords and references. These features indeed shift over time, supporting our claim of covariate shift.

4.2 Synthesizing Shift Types

To evaluate quantification methods under various shift types, we generate multiple synthetic datasets for temporal quantification. The first step in generating each of these datasets is to draw, for every context c , a ground-truth class distribution $\mathbb{P}(Y \mid C)$. We draw this ground-truth from a Beta distribution $\text{Beta}(\alpha \cdot p^{(c)}, \alpha \cdot (1 - p^{(c)}))$, where $p^{(c)}$ controls the expected class prevalences and $\alpha = 50$ determines the concentration around this expectation. The second step is to realize each $\mathbb{P}(Y \mid C)$ in a sample of $m = 1000$ labels and, for each label y , to draw a corresponding feature vector x from $\mathbb{P}(X \mid Y = y, C = c)$. We realize each $\mathbb{P}(X \mid Y, C)$ as a single multivariate normal distribution per class.

This setup is deeply inspired by the Artificial Prevalence Protocol (APP) [6] but introduces two novelties. First, our parametrization of the Beta distribution allows us to alter the expected class distribution over time while the original APP would draw from a fixed $\text{Beta}(1, 1)$ distribution, making all class distributions equally likely. Second, our context-dependent $\mathbb{P}(X \mid Y, C)$ allows us to alter even the relation between features and labels over time while the original APP would

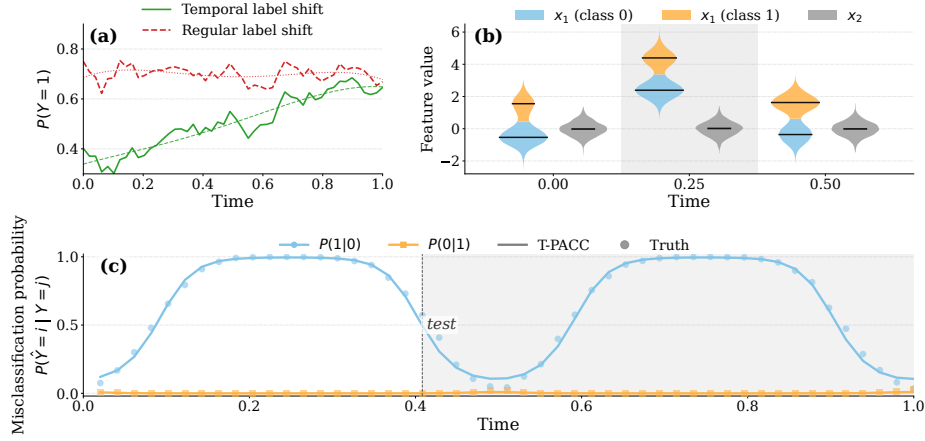


Fig. 7. Dataset shifts in our synthetic data: **(a)** class prevalence $\mathbb{P}(Y=1|C=c)$ over time with and without temporal dependence, corresponding to LS_{syn} (absence of temporal dependency) and to TLS_{syn} and TJS_{syn} (presence of temporal dependency); **(b)** distributions of the features x_1 (blue/orange for class 0/1) and x_2 (gray for both classes) at three time steps of TJS_{syn} ; **(c)** misclassification probabilities over time, as a result of TJS_{syn} , in terms of true empirical estimates (scatter) and T-PACC predictions (lines); here, the vertical line separates training from test.

draw from a fixed, context-independent $\mathbb{P}(X | Y)$. These novel options allow us to synthesize three datasets with different types of dataset shift.

LS_{syn} —Label Shift We fix $\mathbb{P}(X | Y, C) = \mathbb{P}(X | Y)$ and $p^{(c)} = 0.7$, keeping both the class-wise feature distributions and the Beta distribution constant (like in the standard APP). As a result, only $\mathbb{P}(Y | C)$ changes across contexts, albeit without any temporal dependency. Therefore, the generated data represents a regular, non-temporal label shift through which we assess quantification performance in the absence of temporal dependencies.

TLS_{syn} —Temporal Label Shift We still fix $\mathbb{P}(X | Y, C) = \mathbb{P}(X | Y)$ but now alter $p^{(c)}$ over time, linearly interpolating from $p^{(\text{first})} = 0.3$ to $p^{(\text{last})} = 0.7$. This interpolation introduces a systematic and smooth dependency of the context’s class distribution on time, resulting in a temporal label shift.

TJS_{syn} —Temporal Joint Shift We still alter $p^{(c)}$ like above but now also alter $\mathbb{P}(X | Y, C)$, by shifting the means of both class-wise normal distributions over time. This combination of altering both $p^{(c)}$ and $\mathbb{P}(X | Y, C)$ introduces a systematic and smooth dependency of both the context’s labels and features on time, resulting in a temporal joint shift.

Fig. 7 illustrates our synthetic datasets in terms of the shifts they exhibit in their label and feature spaces. We use these datasets in addition to existing benchmark data when evaluating our proposals against the SOTA.

Table 1. MAE values of methods on various datasets. **Bold** values indicate the best-performing methods for each dataset.

Quantifiers	Bike	Bike×2	News	News×2	LS _{syn}	TLS _{syn}	TJS _{syn}
Regular Quantifiers For Label Shift							
ACC (LR)	0.1780	0.1930	0.3253	0.3020	0.0029	0.0030	0.4146
DyS (LR)	0.2113	0.2110	0.2143	0.2490	0.0024	0.0026	0.3993
EDy (LR)	0.1827	0.2030	0.2175	0.2440	0.0023	0.0024	0.3982
PACC (LR)	0.1982	0.2220	0.2243	0.2480	0.0024	0.0024	0.4009
ACC (RF)	0.2443	0.2550	0.2248	0.2230	0.0026	0.0036	0.4126
DyS (RF)	0.2622	0.2650	0.2282	0.2560	0.0026	0.0028	0.4128
EDy (RF)	0.2604	0.3090	0.2039	0.2330	0.0027	0.0027	0.4102
PACC (RF)	0.2648	0.3280	0.1981	0.2240	0.0027	0.0027	0.4106
KFMA—Quantifiers Adjusted For Temporal Label Shift							
ACC (LR)	0.1667	0.1840	0.0960	0.1110	0.0029	0.0030	0.2181
DyS (LR)	0.1918	0.1940	0.1515	0.1910	0.0024	0.0026	0.2372
EDy (LR)	0.1725	0.1930	0.1532	0.1790	0.0023	0.0024	0.2328
PACC (LR)	0.1885	0.2110	0.1597	0.1740	0.0024	0.0024	0.2264
ACC (RF)	0.1843	0.1780	0.2162	0.2040	0.0026	0.0038	0.2171
DyS (RF)	0.1871	0.1660	0.1805	0.2130	0.0026	0.0030	0.2198
EDy (RF)	0.2087	0.2470	0.1696	0.1980	0.0027	0.0030	0.2192
PACC (RF)	0.2134	0.2730	0.1637	0.1910	0.0027	0.0029	0.2183
Regular Quantifiers For Covariate Shift							
PCC (LR)	0.2587	0.2756	0.0849	0.0857	0.0073	0.0084	0.2383
PCC (RF)	0.2075	0.2229	0.0866	0.0917	0.0081	0.0066	0.2372
Quantifiers For Temporal Joint Shift (Ours)							
T-PACC (LR)	0.4029	0.1196	0.4764	0.3381	0.0023	0.0029	0.0860
T-PACC (RF)	0.3838	0.1990	0.5185	0.4018	0.0030	0.0027	0.0677

4.3 Evaluation

We evaluate quantification methods from four groups: (i) regular quantifiers that have been designed for label shift, namely ACC, PACC, DyS, and EDy [4,6,7,15]; (ii) the same quantifiers, now with a temporal KFMA post-processing [13]; (iii) PCC [1], a quantifier with desirable properties under covariate shift [22]; and (iv) our own method T-PACC, an instance of the temporal distribution matching framework that we designed for temporal joint shift. For each quantifier, we consider two base classifiers: Logistic Regression (LR) and Random Forest (RF), both with default hyperparameters. Quantification error is assessed using the Mean Absolute Error (MAE), averaging the absolute difference between the true and estimated class prevalences across all testing bags. Overall, this setup is equivalent to the one used by Li et al. [13]. We discuss the results, which are summarized in Table 1, in correspondence with our hypotheses **H1** and **H2**.

Regarding **H1**, the synthetic data provide evidence that regular label shift-robust quantifiers and KFMA-adjusted quantifiers are effective under both temporal and non-temporal label shift. On the LS_{syn} and TLS_{syn} datasets all methods achieve remarkably low MAE values. This behavior is consistent with Theorem 1 and Corollary 2, which establish that standard quantifiers are already consistent in these settings. In fact, LS_{syn} and TLS_{syn} are deliberately designed to evoke this expected behavior. Beyond this expectation, we further observe that our T-PACC method matches the performance of the SOTA on LS_{syn} and

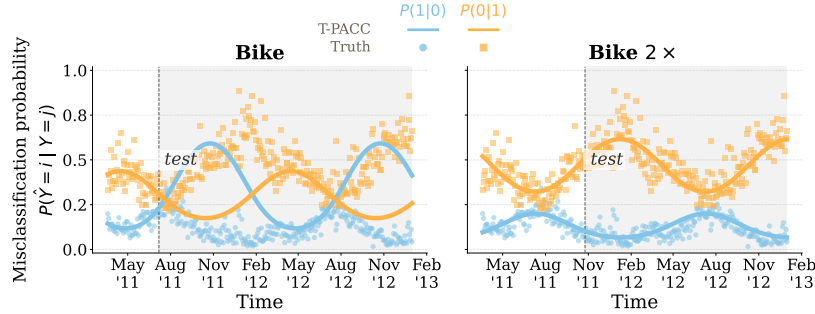


Fig. 8. Misclassification probabilities for the Bike dataset trained on the original amount (left) and the doubled amount (right) of training data. Scatter points show the true classifier behavior and solid lines show the T-PACC predictions. The vertical lines mark the beginning of the respective test phases.

TLS_{syn} , indicating that its temporal modeling does not compromise performance if the relation between features and labels remains stable over time.

Hypothesis **H2**, in turn, highlights a gap in the current literature on temporal quantification: existing approaches are not designed to explicitly handle shifts in the relation between features and labels. This limitation is revealed by the overall inability of label shift-robust and KFMA-adjusted quantifiers to improve their performance when the amount of the quantifier’s training data is doubled on the Bike and News datasets. In the case of News, the best performance is achieved by the covariate shift-robust PCC quantifier, supporting our interpretation that this dataset is mainly affected by covariate shift.

Concerning the Bike and Bike $\times 2$ datasets, we emphasize that T-PACC urgently requires the doubled training data of Bike $\times 2$ to quantify accurately in terms of the MAE. Fig. 8 traces this behavior down to an incorrect prediction of the Bike seasonality if only the original amount of training data—which doesn’t yet fully characterize this seasonality—is available. As soon as the training data is doubled, T-PACC has sufficient information to model the seasonality and, hence, the time-dependent class prevalences to beat the SOTA by a large margin. All other quantifiers exhibit the opposite behavior: their performance even declines as a result of doubling the training data, suggesting their inherently weak transferability across time if confronted with temporal joint shift.

5 Conclusion

We established that temporal quantification can suffer from different types of time-dependent dataset shift. Each of these types has to be addressed with a different family of quantification methods that enjoys consistency under the respective dataset shift at hand.

Our results confirm that the SOTA forecasting-based adjustments, such as KFMA, are effective under *temporal label shift*. For this type of shift, regular

label shift-robust quantifiers already provide a reliable basis for temporal adjustments to act as an effective regularizer over consecutive predictions. However, the forecasting-based adjustment framework becomes insufficient under other types of dataset shift, like temporal covariate shift and temporal joint shift.

While *temporal covariate shift* can potentially be addressed with existing covariate shift-robust quantifiers like PCC, no existing method is able to handle *temporal joint shift*. Therefore, we proposed temporal distribution matching as a novel framework that explicitly models time-dependent changes in the relation between features and labels. Our experiments demonstrate that T-PACC, an instantiation of our proposal that is based on the well-known PACC method, achieves superior performance on data affected by temporal joint shift.

Limitations We evaluated only PCC, T-PACC, and KFMA as representative methods respectively designed for covariate shift, temporal joint shift, and temporal label shift. Our evaluation was limited to two challenging datasets used by Li et al. [13] and to synthetic datasets with isolated types of shift. Whether T-PACC generally requires more data than other methods remains open.

Outlook Future work should address these limitations, extend our temporal distribution matching framework to other base quantifiers, and explore richer temporal representations like neural networks for temporal joint shift. Our general proposal of context-dependent distribution matching should also be explored for other types of dataset shift beyond temporal ones.

Acknowledgments. This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project number 576571177, the Araucária Foundation (Call 14/2024), the AGEUNI Agreement (41/2024), and the State Secretariat of Science, Technology and Higher Education of Paraná (SETI) (TED 091/2025).

References

1. Bella, A., Ferri, C., Hernández-Orallo, J., Ramírez-Quintana, M.J.: Quantification via probability estimators. In: Int. Conf. on Data Min. pp. 737–742. IEEE (2010). <https://doi.org/10.1109/ICDM.2010.75>
2. Bunse, M.: Unification of algorithms for quantification and unfolding. In: Worksh. on Mach. Learn. for Astropart. Phys. and Astron. pp. 459–468. Gesellschaft für Informatik e.V. (2022). https://doi.org/10.18420/INF2022_37
3. Esuli, A., Fabris, A., Moreo, A., Sebastiani, F.: Learning to Quantify. Springer (2023). <https://doi.org/10.1007/978-3-031-20467-8>
4. Esuli, A., Moreo, A., Sebastiani, F.: LeQua@CLEF2022: Learning to quantify. In: Adv. in Inform. Retr. pp. 374–381. Springer (2022). https://doi.org/10.1007/978-3-030-99739-7_47
5. Firat, A.: Unified framework for quantification. arXiv (2016), <http://arxiv.org/abs/1606.00868>
6. Forman, G.: Counting positives accurately despite inaccurate classification. In: Europ. Conf. on Mach. Learn. pp. 564–575. Springer (2005). https://doi.org/10.1007/11564096_55

7. Forman, G.: Quantifying counts and costs via classification. *Data Mining and Knowl. Discov.* **17**(2), 164–206 (2008). <https://doi.org/10.1007/s10618-008-0097-y>
8. Gao, W., Sebastiani, F.: From classification to quantification in tweet sentiment analysis. *Soc. Netw. Analys. and Min.* **6**(19), 1–22 (2016). <https://doi.org/10.1007/s13278-016-0327-z>
9. González, P., Castaño, A., Chawla, N.V., del Coz, J.J.: A review on quantification learning. *ACM Comput. Surv.* **50**(5), 74:1–74:40 (2017). <https://doi.org/10.1145/3117807>
10. González, P., Castaño, A., Peacock, E.E., Díez, J., Del Coz, J.J., Sosik, H.M.: Automatic plankton quantification using deep features. *J. of Plankton Res.* **41**(4), 449–463 (2019). <https://doi.org/10.1093/plankt/fbz023>
11. Hopkins, D.J., King, G.: A method of automated nonparametric content analysis for social science. *Amer. J. of Polit. Sci.* **54**(1), 229–247 (2010). <https://doi.org/10.1111/j.1540-5907.2009.00428.x>
12. Kull, M., Flach, P.: Patterns of dataset shift. In: *Int. Worksh. on Learn. Over Multiple Contexts* (2014), http://dmip.webs.upv.es/LMCE2014/Papers/lmce2014_submission_10.pdf
13. Li, F., Gharakheili, H.H., Batista, G.: Quantification over time. In: *Europ. Conf. on Mach. Learn. and Knowl. Discov. in Databases*. pp. 282–299. Springer (2024). https://doi.org/10.1007/978-3-031-70362-1_17
14. Maletzke, A., Reis, D., Cherman, E., Batista, G.: On the need of class ratio insensitive drift tests for data streams. In: *Second international workshop on learning with imbalanced domains: theory and applications*. pp. 110–124. PMLR (2018)
15. Maletzke, A.G., dos Reis, D.M., Cherman, E.A., Batista, G.E.A.P.A.: Dys: A framework for mixture models in quantification. In: *AAAI Conf. on Artif. Intell.* pp. 4552–4560. AAAI Press (2019). <https://doi.org/10.1609/AAAI.V33I01.33014552>
16. Moreno-Torres, J.G., Raeder, T., Alaíz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. *Pattern Recognit.* **45**(1), 521–530 (2012). <https://doi.org/10.1016/j.patcog.2011.06.019>
17. O’Connor, B., Balasubramanyan, R., Routledge, B.R., Smith, N.A.: From tweets to polls: Linking text sentiment to public opinion time series. In: *Int. Conf. on Weblogs and Social Media*. The AAAI Press (2010), <http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/view/1536>
18. Pearl, J.: *Causality: models, reasoning, and inference*. Cambridge Univ. Press (2009)
19. Souza, V.M., dos Reis, D.M., Maletzke, A.G., Batista, G.E.: Challenges in benchmarking stream learning algorithms with real-world data. *Data Min. and Knowl. Discov.* **34**(6), 1805–1858 (2020). <https://doi.org/10.1007/S10618-020-00698-5>
20. Storkey, A., et al.: When training and test sets are different: Characterizing learning transfer. *Dataset shift in machine learning* **30**(3-28), 6 (2009)
21. Tasche, D.: Fisher consistency for prior probability shift. *arXiv* (2017), <http://arxiv.org/abs/1701.05512>
22. Tasche, D.: Invariance assumptions for class distribution estimation. In: *Int. Worksh. on Learn. to Quantify*. pp. 56–71 (2023), <https://lq-2023.github.io/proceedings/CompleteVolume.pdf>
23. Wen, M., Yang, D., Rosé, C.P.: Sentiment analysis in MOOC discussion forums: What does it tell us? In: *Int. Conf. on Educ. Data Min.* pp. 130–137 (2014), http://www.educationaldatamining.org/EDM2014/uploads/procs2014/longpapers/130_EDM-2014-Full.pdf

A Proofs

A.1 Proof of Theorem 1 (Fisher Consistency Under Label Shift)

To establish Fisher consistency under label shift, we first need to establish the following property of feature mappings on the population level.

Lemma 1. *Consider any feature mapping $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ and let its extension to conditional distributions be $\Phi(X | \cdot) = \mathbb{E}_{x \sim \mathbb{P}(X=x|\cdot)} \Phi(x) \in \mathcal{Z}$. Then, Eq. 1 implies that, for any context $c \in \mathcal{C}$,*

$$\Phi(X | C = c) = \sum_{y \in \mathcal{Y}} \Phi(X | Y = y) \cdot \mathbb{P}(Y = y | C = c)$$

Proof. We prove the claim by a direct rearrangement, using Eq. 1.

$$\begin{aligned} \Phi(X | C = c) &\stackrel{\text{Def.}}{=} \mathbb{E}_{x \sim \mathbb{P}(X=x|C=c)} \Phi(x) \\ &= \mathbb{E}_{(x,y) \sim \mathbb{P}(X=x, Y=y|C=c)} \Phi(x) \\ &\stackrel{\text{Eq. 1}}{=} \mathbb{E}_{(x,y) \sim \mathbb{P}(X=x|Y=y) \cdot \mathbb{P}(Y=y|C=c)} \Phi(x) \\ &= \mathbb{E}_{y \sim \mathbb{P}(Y=y|C=c)} \mathbb{E}_{x \sim \mathbb{P}(X=x|Y=y)} \Phi(x) \\ &\stackrel{\text{Def.}}{=} \mathbb{E}_{y \sim \mathbb{P}(Y=y|C=c)} \Phi(X | Y = y) \\ &= \sum_{y \in \mathcal{Y}} \Phi(X | Y = y) \cdot \mathbb{P}(Y = y | C = c) \quad \square \end{aligned}$$

Note that Lemma 1, while making a claim on the population level, deals with quantities that can be estimated with the given unlabeled data. Specifically, $\Phi(X | C = c)$ can be estimated with the unlabeled data from c and $\Phi(X | Y = y)$ can be estimated with the training data. The expectation of these estimates obeys the equality established by the lemma, such that, with increasing amounts of data, $\mathbb{P}(Y | C = c)$ is increasingly likely to minimize any loss between the two estimated hand sides.

To ensure that this solution is the only one, we further require the identifiability condition from Eq. 3. This condition essentially states that the classes are distinguishable in the mapped feature space $\mathcal{Z}$. Having this intuition established, we now turn to the main proof of distribution matching being Fisher consistent under label shift.

Proof (Theorem 1). Let

$$\mathcal{P} = \{\mathbb{P} : \mathbb{P}(X | Y, C) = \mathbb{P}(X | Y)\}$$

be the set of all distributions that suffer exclusively from label shift, as defined by Eq. 1. Hence, we assert that Lemma 1 holds for all $\mathbb{P} \in \mathcal{P}$. Consequently, there it as least one $\mathbb{P}(Y | C = c)$ which solves the equation from Lemma 1.

Considering any two solutions $\mathbb{P}_a$ and $\mathbb{P}_b$, we employ the identifiability condition from Eq. 3 to see that

$$\begin{aligned} & \sum_{y \in \mathcal{Y}} \Phi(X | Y=y) \mathbb{P}_a(Y=y | C=c) \stackrel{\text{Lemma 1}}{=} \sum_{y \in \mathcal{Y}} \Phi(X | Y=y) \mathbb{P}_b(Y=y | C=c) \\ \Rightarrow & \sum_{y \in \mathcal{Y}} \Phi(X | Y=y) \left(\mathbb{P}_a(Y=y | C=c) - \mathbb{P}_b(Y=y | C=c) \right) = 0 \\ \stackrel{\text{Eq. 3}}{\Rightarrow} & \mathbb{P}_a(Y=y | C=c) - \mathbb{P}_b(Y=y | C=c) = 0 \quad \forall y \in \mathcal{Y} \end{aligned}$$

such that $\mathbb{P}_a = \mathbb{P}_b$ and, hence, the solution is unique. Replacing the finite sample estimates $\hat{\Phi}$ in distribution matching, as defined by Eq. 2, with representations of populations Φ yields precisely this solution. $\square$

A.2 Proof of Corollary 1 (Fisher Consistency Under Temporal Label Shift)

The result follows directly from Theorem 1, once we establish that the equality of Lemma 1 also holds under temporal label shift. Formally, the proof is as follows.

Proof (Corollary 1). Let

$$\mathcal{P}' = \{ \mathbb{P} : \mathbb{P}(X | Y, C, T) = \mathbb{P}(X | Y) \wedge \mathbb{P}(Y | C, T) = \mathbb{P}(Y | C) \}$$

be the set of all distributions that suffer from temporal label shift. Moreover, let $\mathcal{P} \neq \mathcal{P}'$ be the set of all distributions that suffer from non-temporal label shift, as defined in the proof of Theorem 1.

Recognize that Eq. 4 directly implies Eq. 1. Consequently, we have $\mathcal{P}' \subseteq \mathcal{P}$ and can assert that Lemma 1—holding for all $\mathbb{P} \in \mathcal{P}$ —also holds for all $\mathbb{P} \in \mathcal{P}'$. With Lemma 1 holding true, the remainder of this proof is analogous to the proof of Theorem 1. $\square$

A.3 Proof of Theorem 2 (Fisher Consistency Under Temporal Joint Shift)

We proceed in analogy to Sec. A.1, starting by first establishing a property of feature mappings on the population level.

Lemma 2. *Consider any feature mapping $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ and let its extension to conditional distributions be $\Phi(X | \cdot) = \mathbb{E}_{x \sim \mathbb{P}(X=x|\cdot)} \Phi(x) \in \mathcal{Z}$. Then, Eqs. 5 and 6 imply that, for any context $c \in \mathcal{C}$,*

$$\Phi(X | C=c) = \sum_{y \in \mathcal{Y}} \Phi(X | Y=y, C=c) \cdot \mathbb{P}(Y=y | C=c) \quad (10)$$

where

$$\Phi(X | Y=y, C=c) = \int_{\mathcal{T}} \Phi(X | Y=y, T=t) \cdot \mathbb{P}(T=t | C=c) dt \quad (11)$$

Proof. We prove the claim by a direct rearrangement, using Eqs. 5 and 6.

$$\begin{aligned}
\Phi(X \mid C = c) &\stackrel{\text{Def.}}{=} \mathbb{E}_{x \sim \mathbb{P}(X=x|C=c)} \Phi(x) \\
&= \mathbb{E}_{(x,y,t) \sim \mathbb{P}(X=x, Y=y, T=t|C=c)} \Phi(x) \\
&= \mathbb{E}_{(x,y,t) \sim \mathbb{P}(X=x|Y=y, T=t, C=c) \cdot \mathbb{P}(Y=y, T=t|C=c)} \Phi(x) \\
&\stackrel{\text{Eq. 6}}{=} \mathbb{E}_{(x,y,t) \sim \mathbb{P}(X=x|Y=y, T=t) \cdot \mathbb{P}(Y=y, T=t|C=c)} \Phi(x) \\
&\stackrel{\text{Eq. 5}}{=} \mathbb{E}_{(x,y,t) \sim \mathbb{P}(X=x|Y=y, T=t) \cdot \mathbb{P}(Y=y|C=c) \cdot \mathbb{P}(T=t|C=c)} \Phi(x) \\
&= \mathbb{E}_{y \sim \mathbb{P}(Y=y|C=c)} \mathbb{E}_{t \sim \mathbb{P}(T=t|C=c)} \mathbb{E}_{x \sim \mathbb{P}(X=x|Y=y, T=t)} \Phi(x) \\
&\stackrel{\text{Def.}}{=} \mathbb{E}_{y \sim \mathbb{P}(Y=y|C=c)} \mathbb{E}_{t \sim \mathbb{P}(T=t|C=c)} \Phi(X \mid Y=y, T=t) \\
&= \sum_{y \in \mathcal{Y}} \mathbb{P}(Y=y|C=c) \int_{\mathcal{T}} \Phi(X \mid Y=y, T=t) \cdot \mathbb{P}(T=t|C=c) dt \quad \square
\end{aligned}$$

The main proof for Fisher consistency under temporal joint shift works in complete analogy to the proof that Appendix A.1 established for regular label shift. Here, we just need to replace the label shift identifiability condition with the temporal identifiability condition from Eq. 9 and the label shift lemma with Lemma 2.

Proof (Theorem 2). Let

$$\mathcal{P} = \{\mathbb{P} : \mathbb{P}(X \mid Y, C, T) = \mathbb{P}(X \mid Y, T) \wedge \mathbb{P}(Y \mid C, T) = \mathbb{P}(Y \mid C)\}$$

be the set of all distributions that suffer from temporal joint shift, as defined by Eqs. 5 and 6. Hence, we assert that Lemma 2 holds for all $\mathbb{P} \in \mathcal{P}$. Consequently, there it as least one $\mathbb{P}(Y \mid C = c)$ which solves Eq. 10 from Lemma 2.

Considering any two solutions $\mathbb{P}_a$ and $\mathbb{P}_b$, we employ the identifiability condition from Eq. 9 to see that

$$\begin{aligned}
&\sum_{y \in \mathcal{Y}} \Phi(X|Y=y, C=c) \mathbb{P}_a(Y=y|C=c) \stackrel{\text{Lemma 2}}{=} \sum_{y \in \mathcal{Y}} \Phi(X|Y=y, C=c) \mathbb{P}_b(Y=y|C=c) \\
&\Rightarrow \sum_{y \in \mathcal{Y}} \Phi(X \mid Y=y, C=c) \left(\mathbb{P}_a(Y=y \mid C=c) - \mathbb{P}_b(Y=y \mid C=c) \right) = 0 \\
&\stackrel{\text{Eq. 9}}{\Rightarrow} \mathbb{P}_a(Y=y \mid C=c) - \mathbb{P}_b(Y=y \mid C=c) = 0 \quad \forall y \in \mathcal{Y}
\end{aligned}$$

such that $\mathbb{P}_a = \mathbb{P}_b$ and, hence, the solution is unique. Replacing the finite sample estimates $\hat{\Phi}$ in temporal distribution matching, as defined by Eq. 7, with representations of populations Φ yields precisely this solution. $\square$

B T-PACC

We exemplify our temporal distribution matching through *T-PACC*, a temporal variant of Probabilistic Adjusted Classify & Count. Like any adaptation of an

existing distribution matching method to temporal joint shift, T-PACC needs to learn a time-dependent representation $\hat{\Phi}(X | Y, T)$ and this task can be approached with many different models.

For simplicity, our experiments take this representation to be modeled as one logistic regression per class, $\sigma(\langle \beta_y, \bar{t} \rangle + \alpha_y) \in \mathcal{Z}$, where σ is the soft-max operator, $\beta_y \in \mathbb{R}^{|\mathcal{Y}| \times d}$ is a matrix of model coefficients, $\alpha_y \in \mathbb{R}^{|\mathcal{Y}|}$ is a bias vector, and $\bar{t} \in \mathbb{R}^d$ is a vector of features that we compute from the time $t \in \mathcal{T}$. For instance, the feature vector $\bar{t} = (\sin t_\pi, \cos t_\pi)$ with $t_\pi = t - \pi \lfloor \frac{t}{\pi} \rfloor$ would allow the logistic regression to learn the periodical impact of some time period π on the representation, be this period a day, a year, or a hyper-parameter. The soft-max activation is a natural choice for T-PACC because Φ represents probabilistic classifier outputs, which have the same image as this activation function.

T-PACC is detailed further in Algorithm 1. We emphasize that any other distribution matching algorithm can be adapted to temporal joint shift in the same way that is outlined there. We emphasize further that, instead of logistic regressions with suitable input features, $\hat{\Phi}(X | Y, T)$ could also be modeled as a deep neural network that automatically extracts suitable features during the learning process.

Algorithm 1 Temporal Distribution Matching—T-PACC

Input: a probabilistic classifier $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$, target data $\{(x_i, t_i) \sim \mathbb{P}(X, T | C=c)\}_{i=1}^{m'}$, and training data $\{(x_i, y_i, t_i) \sim \mathbb{P}(X, Y, T | C=c_{\text{trn}})\}_{i=1}^m$.

Output: a prediction $\hat{\mathbb{P}}(Y|C=c)$

Training Phase—determine the parameters of the time-dependent representation:

- 1: **for** $y \in \mathcal{Y}$ **do**
- 2: $\alpha_y, \beta_y \leftarrow \arg \min_{\alpha, \beta} \sum_{i=1}^m \mathbb{1}_{y_i=y} H(\Phi(x_i), \sigma(\langle \beta, \bar{t}_i \rangle + \alpha))$ {cross entropy loss}
- 3: **end for**

Temporal Forecasting Phase—determine the representation of the current context:

- 1: **for** $y \in \mathcal{Y}$ **do**
- 2: $\hat{\Phi}(X | Y=y, C=c) \leftarrow \frac{1}{m'} \sum_{i=1}^{m'} \sigma(\langle \beta_y, \bar{t}_i \rangle + \alpha_y)$ {integration over target data}
- 3: **end for**

Quantification Phase—predict the class distribution:

- 1: **return** $\arg \min_{\mathbb{P}} \|\hat{\Phi}(X|C=c), \sum_{y \in \mathcal{Y}} \hat{\Phi}(X|Y=y, C=c) \cdot \mathbb{P}(Y=y|C=c)\|_2^2$
-

C Dataset Characteristics

Table 2 presents the datasets and the sampling employed in the experiments. The Bike and News benchmark datasets were used with the default preprocessing and sampling strategy as in Li et al. [13], and also with doubled validation size (Bike $\times 2$, News $\times 2$). The remaining datasets are the synthetic generated ones, described in Subsection 4.2.

Table 2. Experimental dataset sampling.

Dataset	Classes	Classifier Training Bags	Quantifier Training Bags	Testing Bags
Bike	2	38	55	273
Bike $\times$ 2	2	38	110	218
News	2	21	36	185
News $\times$ 2	2	21	72	149
LS _{syn}	2	1	19	30
TLS _{syn}	2	1	19	30
TJS _{syn}	2	1	19	30

Uncertainty-Aware Classifier Accuracy Prediction under Prior Probability Shift

Lorenzo Volpi[✉], Alejandro Moreo[✉], and Fabrizio Sebastiani[✉]

Istituto di Scienza e Tecnologie dell’Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
`{firstname.lastname}@isti.cnr.it`

Abstract. *Classifier accuracy prediction (CAP)* is the task of estimating the accuracy that a trained classifier will have on a set of datapoints characterized by *dataset shift*, i.e., data sampled from a distribution different from the distribution from where the training data originated. We here propose three new algorithms for CAP under *prior probability shift*, an important type of dataset shift; our first two algorithms are built on top of surrogate quantifiers, while the third is a Bayesian algorithm. Importantly, these methods do not merely return a point estimate (i.e., the estimated accuracy of the classifier on the unlabelled data), but also return a measure of their uncertainty in this estimate. Systematic experiments on a number of datasets and against a number of baseline CAP algorithms show that our methods are competitive with strong baselines, with one of them frequently achieving the best balance between point estimation accuracy and interval quality.

Keywords: Classifier accuracy prediction, Dataset shift, Prior probability shift, Label shift, Quantification, Point estimates, Uncertainty quantification, Confidence intervals

1 Introduction

The accuracy that a classifier will obtain when applied to new unlabelled data is usually estimated by cross-validation. However, the assumption that this procedure returns *reliable* estimates rests on the assumption that the labelled test data used in the cross-validation process are representative of the target unlabelled data. In other words, cross-validation relies on the *IID assumption*, which states that the joint distribution $P(X, Y)$ from which the labelled data are sampled and the joint distribution $Q(X, Y)$ from which the unlabelled data are sampled, are the same. When $P(X, Y) \neq Q(X, Y)$ we say we are instead in the presence of *dataset shift* [21, 25], and *classifier accuracy prediction* (CAP, also known as *unsupervised accuracy estimation* [5, 18], or *out-of-distribution error*

prediction [31]) can no more be carried out by cross-validation, thus becoming an open problem.¹

Until a few years ago, little research had been conducted on CAP under dataset shift, but in recent years the scientific community has investigated the problem more vigorously, and to deliver algorithms that address it [9, 11, 17, 18, 28, 29]. Most solutions proposed so far (and the present paper is no exception) address *prior probability shift* (PPS, a.k.a. *label shift*), a type of dataset shift that has sometimes been called *the paradigmatic* case of dataset shift [33].

Most CAP works published so far return, given a classifier h and a set σ of unlabelled datapoints, a *point estimate* of the accuracy that h is going to have on σ . However, since the returned estimate is characterized by uncertainty, it would be desirable to *quantify* this uncertainty [14]; measuring the uncertainty of the produced estimate is one of the goals of the present paper.

The contributions of this paper are threefold. First, we propose two new methods for CAP under PPS, dubbed *SQS-H* and *SQS-S*; these methods use a *quantifier* (i.e., a model trained to estimate the prevalence values of the classes in a set of unlabelled datapoints; see [7]) as an auxiliary model for selecting validation sets that resemble the target distribution. Second, we propose *BayCAP*, a Bayesian CAP method that samples sets from a posterior distribution over target contingency tables and turns these sets into posterior sets of classifier accuracy values. Third, we analyse the ability of these methods to measure the uncertainty of the returned classifier accuracy estimate.

Our paper is organized as follows. Section 2 summarizes related work in CAP under dataset shift, and in uncertainty quantification for CAP. In Section 3 we set the notation we use, and review our underlying distributional assumptions and the reasons why cross-validated estimation of accuracy is unreliable under prior probability shift. In Section 4, we describe two new CAP methods that rely on surrogate quantifiers, while in Section 5 we propose a fully Bayesian approach to CAP. Section 6 describes the experiments we have carried out, while Section 7 wraps up, pointing at avenues for further research.

2 Related work

Research on CAP under dataset shift has grown substantially in recent years. Earlier proposals include methods based on confidence scores, domain-shift diagnostics, nearest-neighbour structure, or distribution matching [2–6, 9, 12, 15, 20, 26, 31, 32]. We focus here on the methods that are closest to our setting, i.e., CAP under PPS, and on CAP methods that either estimate contingency tables or quantify uncertainty.

The *Difference of Confidence* (DoC) method [11] estimates the accuracy of a classifier from changes in the confidence scores it returns. DoC constructs shifted validation sets, measures how the average confidence changes with respect to the

¹ Throughout this paper, by CAP we will indicate a task where no access to labels from distribution $Q(X, Y)$ is assumed. Other works (e.g., [30]) assume instead that a limited number of labels from $Q(X, Y)$ are available to help in the prediction.

original validation set, and trains a regressor to map these confidence differences to differences in accuracy.

More recent work has moved from directly estimating the value of a single accuracy measure, to estimating the values of the entries of the contingency table produced by the classifier. This is useful because several evaluation measures can then be computed from the same estimated table entry values. QuAcc [29] treats the entries of the contingency table as classes and estimates their prevalence by means of a PPS-robust quantifier. LEAP [28] instead solves a system of linear equations that the contingency table entry values must satisfy under PPS, also using a PPS-robust quantifier for estimating class prevalence values required by the equations.

Confidence-Based Performance Estimation (CBPE – [17]) also targets the contingency table, but models its entries as random variables whose values derive from the classifier’s confidence scores. This gives CBPE a natural mechanism for measuring the uncertainty of the prediction. However, its guarantees rely on the assumption that the classifier is calibrated on the target distribution. CBPE is therefore the closest existing work to the uncertainty-aware perspective adopted in the present paper, since it models uncertainty over quantities from which evaluation measures can be derived. Nevertheless, its uncertainty model is frequentist rather than Bayesian, and thus does not provide a natural mechanism for incorporating prior knowledge about the target contingency table or the desired evaluation measures. To the best of our knowledge, no Bayesian treatment of uncertainty in CAP has been proposed before the present paper.

3 Preliminaries

3.1 Background and notation

In this section we introduce basic concepts and notation on classifier accuracy prediction and uncertainty quantification.

Let $\mathcal{X} = \mathbb{R}^d$ denote the input space (a space of vectors of covariates), and let $\mathcal{Y} = \{\omega_1, \dots, \omega_K\}$ denote the output space (a space of classes). We assume we are in the presence of prior probability shift (PPS), i.e., that the training and test data are drawn from distributions P and Q , respectively, such that $P(Y) \neq Q(Y)$ and $P(X|Y) = Q(X|Y)$.

In this paper we assume by default that our classifiers are probabilistic, i.e., that they generate a vector of class specific posterior probability estimates (one for each class). This is not too restrictive, since most classifiers generate a vector of class-specific scores that can be converted into posterior probabilities estimates via calibration. In this paper, though, we do not make any hard assumption on how well-calibrated these classifiers are.

Let $h : \mathcal{X} \rightarrow \Delta^{K-1}$ denote a probabilistic classifier mapping input $x \in \mathcal{X}$ into a vector $\hat{p} = h(x) = (\hat{p}_1, \dots, \hat{p}_K)$ of posterior probability estimates, where $\hat{p}_k =$

$\hat{P}(Y = \omega_k | X = x)$, and where Δ^{K-1} denotes the standard probability simplex

$$\Delta^{K-1} = \{p \in \mathbb{R}^K : p_k \geq 0, \sum_{k=1}^K p_k = 1\}$$

and where one can obtain a crisp prediction by simply applying the Maximum a Posteriori (MAP) rule $\hat{y} = \arg \max_{\omega_k \in \mathcal{Y}} \hat{P}(\omega_k | x)$.

A key property, which will be instrumental in the following sections, is that for any deterministic and measurable function f on $\mathcal{X}$, the stationarity $P(X|Y) = Q(X|Y)$ implies that $P(f(X)|Y) = Q(f(X)|Y)$; see also [19, Lemma 1] and [27]. As a direct consequence, if we take f to be a classifier h , this property implies the stationarity of the (crisp and soft) classifier predictions, i.e., it implies that $P(\hat{Y}|Y) = Q(\hat{Y}|Y)$ and $P(Z|Y) = Q(Z|Y)$, with $Z = h(X)$. We write C to denote the matrix of class-conditional classification rates, with $C_{ij} = P(\hat{Y} = \omega_i | Y = \omega_j)$. We denote by $\pi^P = (\pi_1^P, \dots, \pi_K^P) \in \Delta^{K-1}$ the class prevalence values of the source distribution, with $\pi_k^P = P(Y = \omega_k)$; analogously, we write $\pi^Q = (\pi_1^Q, \dots, \pi_K^Q) \in \Delta^{K-1}$ for the target distribution, with $\pi_k^Q = Q(Y = \omega_k)$.

We assume that classifier h (i.e., the classifier whose accuracy we aim to estimate) has already been trained on a set L of labelled datapoints which may no longer be available. We assume access to a validation set V of labelled datapoints (which will allow us to train our accuracy predictors), with L and V drawn from the same distribution P . Given a measure of accuracy A , CAP consists of predicting the accuracy of h on an unlabelled test set U drawn from a distribution Q related to P via PPS. We write $A(h, U)$ to denote the empirical value of measure A on set U . We restrict our attention to evaluation measures A that can be computed from a contingency table T , and write $A(T)$ when the measure is applied directly to such a table.

Note that our goal is to estimate the performance of an already trained classifier h , and not to improve its performance on the target distribution Q . In this paper, we are interested not only in the accuracy of classifier accuracy point estimates, but also in characterizing the uncertainty associated with such estimates.

3.2 Why Can't We Trust Cross-Validation Estimates Under Prior Probability Shift?

Assume our classifier accuracy measure is ("vanilla") accuracy (in symbols: Acc), i.e., the fraction of classification decisions that are correct, and assume we have an arbitrarily good, unbiased estimate $\widehat{\text{Acc}}$ of the classifier's accuracy obtained by means of cross-validation on training data L . Assume our labelled and unlabelled data are drawn from distributions P and Q , respectively, that are related via PPS: can we trust our $\widehat{\text{Acc}}$ estimate? This amounts to asking whether our estimator is unbiased under PPS, i.e., whether

$$\text{Bias}(\widehat{\text{Acc}}) \equiv \mathbb{E}[\widehat{\text{Acc}}] - Q(\hat{Y} = Y) = 0 \quad (1)$$

where $\mathbb{E}$ indicates expectation (taken over all possible sets L of labelled data-points) and $\hat{Y}$ is a random variable ranging on the predicted class labels. Since our training data is drawn from distribution P and our estimate is unbiased, asymptotically it holds that $\mathbb{E}[\widehat{\text{Acc}}] = P(\hat{Y} = Y)$. Note that

$$\begin{aligned} P(\hat{Y} = Y) &= \sum_{\omega_k \in \mathcal{Y}} P(\hat{Y} = \omega_k, Y = \omega_k) = \sum_{\omega_k \in \mathcal{Y}} P(\hat{Y} = \omega_k | Y = \omega_k) P(Y = \omega_k) \\ Q(\hat{Y} = Y) &= \sum_{\omega_k \in \mathcal{Y}} Q(\hat{Y} = \omega_k, Y = \omega_k) = \sum_{\omega_k \in \mathcal{Y}} Q(\hat{Y} = \omega_k | Y = \omega_k) Q(Y = \omega_k) \end{aligned}$$

The stationarity of the class-conditional classifier predictions under PPS (Section 3.1) implies that the class-conditional error rates are stationary as well. Let C denote the matrix with entries $C_{ij} = P(\hat{Y} = \omega_i | Y = \omega_j)$, and let π^P and π^Q be the training and test class prevalence values, respectively. We can now rewrite our above equations as

$$P(\hat{Y} = Y) = \text{diag}(C)^\top \pi^P \quad Q(\hat{Y} = Y) = \text{diag}(C)^\top \pi^Q$$

where $\text{diag}(M) \in \mathbb{R}^n$ denotes the vector of diagonal entries of $M \in \mathbb{R}^{n \times n}$. Letting $d = \text{diag}(C)$, it follows that

$$\text{Bias}(\widehat{\text{Acc}}) = d^\top \pi^P - d^\top \pi^Q = d^\top (\pi^P - \pi^Q) \quad (2)$$

PPS means that $\pi^P \neq \pi^Q$, which thus implies that $\text{Bias}(\widehat{\text{Acc}}) = 0$ holds only if $d = \mathbf{0}$, or if $d \perp (\pi^P - \pi^Q)$.

The first condition ($d = \mathbf{0}$) corresponds to a degenerate case in which the classifier has zero probability of making a correct prediction, and can thus be safely excluded. The second condition ($d \perp (\pi^P - \pi^Q)$) deserves further attention, though. Note that, as both π^P and π^Q lie on the probability simplex Δ^{K-1} , if this condition is verified the vector $(\pi^P - \pi^Q)$ lies in the linear subspace parallel to the simplex, namely $\{x \in [-1, 1]^K : \sum_{i=1}^K x_i = 0\}$. Since d contains the true positive rates of all classes, all its components are non-negative, and orthogonality between d and $(\pi^P - \pi^Q)$ is thus highly restrictive and unlikely to arise in realistic settings.

For example, one case in which $\text{Bias}(\widehat{\text{Acc}}) = 0$ irrespective of π^P and π^Q is when the true positive rates for all the classes are equal, i.e., when $d = \beta \mathbf{1}$ for some $\beta \in \mathbb{R}$. Unless the classifier is a perfect one, such a configuration becomes highly unlikely. However, since this condition does not depend on the class prevalence values (and more importantly, on the unknown π^Q), we can check its validity by inspecting C , i.e., using only training data. If the condition is approximately satisfied, we may safely use our cross-validated estimate of accuracy, even under suspected PPS.

In sum, the accuracy estimates obtained by cross-validation are generally unreliable under PPS.

4 Classifier accuracy prediction via surrogate quantifiers

The methods we propose in this section rely on the same underlying rationale. Since, as a result of Equation 2, accuracy estimates obtained via cross-validation are biased because of the prevalence shift $(\pi^P - \pi^Q)$, our goal is to estimate classifier performance using new sets drawn from the validation data, whose class prevalence values resemble those of the target test distribution. In order to achieve this, we explore new CAP methods that rely on quantification as a surrogate model, i.e., on methods specialised in estimating the class prevalence values π^U of sets U of unlabelled datapoints drawn from the target distribution Q .

A quantifier can be formalized as a function $q : 2^{\mathcal{X}} \rightarrow \Delta^{K-1}$, mapping a set σ of instances from $\mathcal{X}$ to a vector of class prevalence estimates $\hat{\pi}^Q = q(\sigma) = (\hat{\pi}_1^Q, \dots, \hat{\pi}_K^Q)$, where $\hat{\pi}_k^Q = \hat{Q}(Y = \omega_k)$ denotes the estimated prevalence of class ω_k in σ . The error we incur in estimating the true prevalence π^Q as $\hat{\pi}^Q$ is typically measured in terms of a divergence function $\mathcal{D}$.

As discussed in Section 3.1, under PPS it holds that $P(f(X)|Y) = Q(f(X)|Y)$, where function f acts as a *representation function*, typically instantiated as a probabilistic classifier. This allows quantifiers to model distributions over posterior probabilities rather than to directly model distributions of covariates, which is infeasible in high-dimensional spaces. This property underlies the broad class of *aggregative quantifiers* [§4.2 in 7], which includes those adopted in this paper.

In the methods we propose, we *do not* use the target classifier h as the classifier on which our aggregative quantifiers rely. Since the quantifier is used as an auxiliary instrument to estimate the accuracy of h , using h also as the surrogate classifier would make the measurement depend on the very classifier being measured, thus establishing a vicious circle. We thus keep the surrogate classifier used for quantification distinct from the classifier whose accuracy is to be predicted. As a consequence, the surrogate classifier must be trained from the validation data used to train the CAP method. Since in-sample posterior probability estimates would tend to be overconfident, we instead obtain out-of-sample estimates by applying cross-validation within the validation set.

In what follows, we present two new uncertainty-aware CAP methods that use quantification as a surrogate task.

4.1 SQS-H: Surrogate quantifier-based sampling with hard prevalence matching

Our first proposed CAP method based on quantification is *Surrogate Quantifier-based Sampling with Hard prevalence matching* (SQS-H). This method consists of these steps:

1. Train a quantifier q robust to PPS using V as the training data;
2. Compute the class prevalence value estimates $\hat{\pi}^Q = q(U)$ for the unlabelled set U ;

3. By drawing (with replacement) labelled datapoints uniformly at random from validation set V , generate m sets of datapoints $\{\sigma_i\}_{i=1}^m$ with class prevalence values $\pi^{(i)}$ (approximately) equal to the estimated class distribution $\hat{\pi}^Q$;²
4. Compute the accuracy of h on each of the m sets, as $\{A(h, \sigma_i)\}_{i=1}^m$;
5. Use these accuracy values to compute $100(1 - \alpha)\%$ empirical percentile intervals, and return the empirical mean as the point estimate, where α is the significance level.

The rationale of this algorithm is the following. Step 3 induces a new distribution P' over the selected validation sets³, with $P'(Y) \neq P(Y)$. However, Step 3 does not change the class-conditional distributions, since the selected sets are still made of instances drawn from the same conditional distributions as the original validation data, i.e., $P'(X|Y) = P(X|Y)$. Under PPS between validation and target data, it holds that $P(X|Y) = Q(X|Y)$, and thus it holds that $P'(X|Y) = Q(X|Y)$ as well. Since the selection is designed to make the class prior of the selected validation sets close to that of the target set, we also have $P'(Y) \approx Q(Y)$. If PPS is the only source of shift, it follows that $P'(X, Y) \approx Q(X, Y)$ (since $P'(X, Y)$ is factorized as $P'(X|Y)P'(Y)$ and since both factors are similar to the counterparts in Q), so the accuracy of h estimated on sets from P' is a good approximation of its accuracy on the target distribution Q .

4.2 SQS-S: Surrogate quantifier-based sampling with soft prevalence matching

A limitation of SQS-H is that, while we can observe the true prevalence values $\pi^{(i)}$ of our validation sets, we cannot observe the true prevalence π^Q of U , since we can observe only an estimate $\hat{\pi}^Q$, which is affected by quantification error.

The second quantification-based CAP method we propose is called *SQS-S* (standing for *Surrogate Quantifier-based Sampling with Soft prevalence matching*), and is aimed at mitigating this limitation. The key idea behind SQS-S is to inject the same type of estimation error before the selection of validation sets. The rationale is that, whenever the quantifier q returns similar *estimated* class prevalence values $\hat{\pi}^{(i)}$ and $\hat{\pi}^Q$ for sets σ_i and U , respectively, their true class prevalence values $\pi^{(i)}$ and π^Q are also likely to be similar.

A fundamental difference between SQS-S and SQS-H is that the quantifier q cannot be trained on the same validation instances from which sets are later drawn, since this would inevitably lead to overconfident prevalence estimates. Following this rationale, we reserve part of the validation data for quantifier training. A second important difference is that the validation sets employed to

² The fact that we draw with replacement implies that the σ_i 's are *bags*, and not sets. However, this is inessential, since no known quantification method depends on the fact that the datapoints in σ_i are all distinct, which means that quantifiers can indifferently be applied to sets or bags.

³ Note that P and P' are affected by a different type of dataset shift, namely *sampling selection bias*.

estimate accuracy cannot be selected directly, as in SQS-H, since the selection criterion now relies on similarity between *predicted* prevalence values. To address this issue, we first draw n initial sets whose prevalence values approximately cover the probability simplex uniformly, and then retain the fraction γ whose predicted prevalence values are closest to $\hat{\pi}^Q$. Note that this algorithm introduces two hyperparameters: the fraction $\gamma \in [0, 1]$ and the criterion of dissimilarity (divergence) $\mathcal{D}$.

SQS-S consists of the following steps:

1. Randomly partition the validation set V into V_1 and V_2 via stratification, with $|V_1| \approx |V_2|$;
2. Train a quantifier q robust to PPS using V_1 ;
3. Estimate the class prevalence of the unlabelled set $\hat{\pi}^Q = q(U)$;
4. Extract n random sets σ_i by drawing with replacement from V_2 , with each set characterised by different class prevalence values uniformly covering the probability simplex;
5. Estimate class prevalence values in $\hat{\pi}^{(i)} = q(\sigma_i)$ for all sets σ_i ;
6. Take the $m = \lceil \gamma n \rceil$ sets σ_i whose predicted prevalence is closest to $\hat{\pi}$ in terms of a divergence function $\mathcal{D}$. We here adopt the L1 distance, i.e., given $r_i = \|\hat{\pi}^Q - \hat{\pi}^{(i)}\|_1$ and $r_{(m)}$ the m -th closest such distance, we keep $I_\gamma = \{i : r_i \leq r_{(m)}\}$;
7. Compute the accuracy of h $\{A(h, \sigma_j)\}_{j \in I_\gamma}$ in each of the selected sets;
8. Use these accuracy values to compute $100(1 - \alpha)\%$ empirical percentile intervals, and return the empirical mean as the point estimate, where α is the significance level.

5 Bayesian posterior sampling of accuracy

Bayesian methods provide a natural framework for uncertainty-aware classifier accuracy prediction. In contrast to point-estimation approaches, they return a posterior distribution over the quantities of interest and allow prior information to be incorporated explicitly. In our setting, the quantity of interest is the performance of a fixed classifier h on an unlabelled target set U .

Rather than specifying a probabilistic model directly for a particular evaluation measure, we model the posterior distribution of the target contingency table. This choice makes the method metric-agnostic: once posterior sets of the contingency table are available, posterior sets of any contingency-table-based evaluation measure can be obtained by applying the corresponding functional to each sampled table. As from Section 3.1, we use C to denote the classifier prediction rates with (estimated) entries $C_{ij} = P(\hat{Y} = \omega_i \mid Y = \omega_j)$, where $\hat{Y} = h(X)$; this matrix is shared by distributions P and Q under PPS.

We adopt the BayesianCC model of [33], originally proposed as a quantification method, which performs Bayesian inference over the source prevalences, target prevalences, and the conditional prediction matrix. In particular, the posterior is given by

$$p(\pi^P, \pi^Q, C \mid V, U) \propto p(V, U \mid \pi^P, \pi^Q, C) p(\pi^P, \pi^Q, C)$$

Following [33], Dirichlet priors can be placed on the simplex-valued parameters π^P , π^Q , and on the columns of C . The model assumes the sampling process

$$\begin{aligned} (n_{Y=\omega_1}^s, \dots, n_{Y=\omega_K}^s) \mid \pi^P &\sim \text{Multinomial}(N^P, \pi^P) \\ (f_{1j}, \dots, f_{Kj}) \mid n_{Y=\omega_j}^s, C &\sim \text{Multinomial}(n_{Y=\omega_j}^s, C_{:j}) \quad j = 1, \dots, K \\ (n_{\hat{Y}=\omega_1}^t, \dots, n_{\hat{Y}=\omega_K}^t) \mid \pi^Q, C &\sim \text{Multinomial}(N^Q, C\pi^Q) \end{aligned}$$

where $N^P = |V|$ and $N^Q = |U|$. Here, $n_{Y=\omega_j}^s$ denotes the number of validation instances whose true class is ω_j ; f_{ij} denotes the number of validation instances with predicted class ω_i and true class ω_j ; and $n_{\hat{Y}=\omega_i}^t$ denotes the number of target instances predicted as class ω_i .

After drawing m posterior sets $\{(\pi_{(r)}^Q, C_{(r)})\}_{r=1}^m \sim p(\pi^Q, C \mid V, U)$, we construct, for each draw, the corresponding normalized target contingency table $T_{(r)} = C_{(r)} \text{diag}(\pi_{(r)}^Q)$, where entry (i, j) in $T_{(r)}$ represents one posterior draw of the joint probability $Q(\hat{Y} = \omega_i, Y = \omega_j)$. Given any evaluation measure A expressible as a function of a contingency table, we obtain posterior samples of the corresponding target performance as $\{A(T_{(r)})\}_{r=1}^m$. Thus, the posterior distribution over the evaluation measure is the *pushforward* of the posterior distribution over target contingency tables through the metric functional A .

We use the resulting empirical posterior distribution to compute $100(1 - \alpha)\%$ posterior credible intervals and report its mean as a point estimate.

6 Experiments

In this section, we evaluate the proposed uncertainty-aware CAP methods on binary and multiclass tabular datasets under PPS, considering both point-estimation accuracy and interval quality.

Datasets: We evaluate the CAP methods considered in this paper on tabular classification datasets from the UCI Machine Learning Repository [16]. We restrict our attention to those accessible through QuaPy [22], and select the 5 largest binary and 10 largest multiclass datasets. The number of instances ranges from 830 (*mammographic*) to 1,024,985 (*poker_hand*); the number of features ranges from 5 (*mammographic*) to 617 (*isolet*); the number of classes ranges from 2 (all binary datasets) to 26 (*isolet* and *letter*).

Classifier-Training Algorithms: In order to generate the classifiers whose accuracy we aim to predict, we consider five standard learning algorithms: Logistic Regression (LR), k -Nearest Neighbours (kNN), Support Vector Machines (SVM), MultiLayer Perceptrons (MLP), and Random Forests (RF). All classifiers are trained using the implementations provided by scikit-learn [23]. We leave all hyperparameters at their default values, except for SVM, for which we use the RBF kernel.

Baselines: As baselines against which we test the performance of our methods, we use DoC [11], QuAcc [29], LEAP [28], and L-CBPE [17].

For DoC, we use our own implementation, since no public implementation was released by the authors. For QuAcc, we use the QuAcc- $K \times K$ variant, which estimates the entries of the contingency table by reducing the problem to K quantification tasks over the K classes. For LEAP, we use O-LEAP, the overconstrained variant that takes into account all the equations governing the contingency-table entries. For QuAcc and LEAP, we use the original implementations available online.⁴ Since CBPE [17] assumes posterior probabilities to be calibrated for the target distribution, we first recalibrate them with LasCal [24], a calibration method designed for PPS, and refer to the resulting pipeline as L-CBPE. We use the original implementations released by the respective authors; since the available CBPE implementation only supports binary problems, we evaluate this baseline only on binary datasets.

We exclude several previously proposed CAP methods from our comparison; some [3, 15] rely on assumptions that are too restrictive for our setting, while others [4, 9, 20, 31] were not competitive in preliminary experiments. We nevertheless include L-CBPE, despite its lower competitiveness, because CBPE is, to the best of our knowledge, the only existing uncertainty-aware CAP method.

The code that reproduces all our experiments is available on GitHub.⁵

Evaluation Measures: We evaluate the competing CAP methods along two dimensions. First, we assess the quality of their point estimates; second, we assess the quality of their interval estimates.

For evaluating point estimates, we use absolute error (AE). For a generic classifier accuracy measure A , AE is defined as $|A(h, U_i) - \hat{A}(h, U_i)|$, where $A(h, U_i)$ is the true accuracy of classifier h on the i -th test set U_i , and $\hat{A}(h, U_i)$ is the corresponding value predicted by the CAP method. In our experiments, the target classifier accuracy measure A is vanilla accuracy (Acc), which is one of the most widely used measures of classifier effectiveness.

For uncertainty evaluation, we rely on coverage and amplitude of the interval estimates generated by each method. Let $[\ell_i, u_i]$ be the interval estimate generated by a given method at $\alpha = 0.05$ significance level for the i -th test set, and let $a_i = A(h, U_i)$ be the corresponding true accuracy. Coverage ($\mathcal{C}$) is defined as the fraction of the N test sets for which the true accuracy falls within the predicted interval, i.e., $\mathcal{C} = \frac{1}{N} \sum_{i=1}^N \mathbf{1}[\ell_i \leq a_i \leq u_i]$. Amplitude ($\mathcal{A}$) is defined as the average width of the predicted intervals, i.e., $\mathcal{A} = \frac{1}{N} \sum_{i=1}^N (u_i - \ell_i)$. A good interval estimate should thus exhibit high coverage and low amplitude. Since these two objectives are conflicting, we also report the mean Winkler score ($\mathcal{W}$), which combines them into a single measure. For a given interval $[\ell_i, u_i]$, the Winkler score [10] is defined as

$$\mathcal{W}_i = (u_i - \ell_i) + \frac{2}{\alpha} (\max\{0, \ell_i - a_i\} + \max\{0, a_i - u_i\})$$

⁴ <https://github.com/lorenzovolpi/cap-ml>.

⁵ <https://github.com/lorenzovolpi/Uncertainty-Aware-CAP>.

Lower values of AE, amplitude, and Winkler score indicate better performance, while higher values of coverage are preferable. Throughout all experiments, we use nominal level $(1 - \alpha) = 0.95$.

Evaluation Protocol: The experimental protocol we adopt is as follows. Given a labelled collection D of pairs (x_i, y_i) , we split it into a training set (70%) and a test set U (30%) via stratified sampling. We further split, with stratification, the training set into a set L (50%), used for training the classifier h , and a validation set V (50%), used for training the CAP methods. These partitions are consistent across all methods.

We then draw, uniformly at random, 100 prevalence vectors $\mathbf{v}_1, \dots, \mathbf{v}_{100}$ from the unit simplex Δ^{K-1} . For each prevalence vector $\mathbf{v}_i$, we randomly extract, with replacement, from U a test set U_i of size $|U_i| = 1000$ whose class prevalence values match $\mathbf{v}_i$. This protocol, known as the Artificial Prevalence Protocol (APP), is the standard protocol for simulating PPS in the literature [7, 8].

Since our evaluation is uncertainty-aware, all methods are required to return both a point estimate and an interval estimate for each test set. SQS-H, SQS-S, BayCAP, and L-CBPE provide such intervals natively: SQS-H and SQS-S return empirical percentile intervals, BayCAP returns posterior credible intervals, and L-CBPE derives confidence intervals from the Poisson-binomial distribution induced by the calibrated confidence scores. For methods that only return point estimates (namely LEAP, QuAcc, and DoC), we obtain confidence intervals by bootstrap resampling the test set and taking the empirical $\frac{\alpha}{2}$ and $(1 - \frac{\alpha}{2})$ quantiles of the resulting accuracy estimates; we return the mean of such samples as the point estimate.

In all experiments, we let each method generate 1,000 estimates. For surrogate-quantifier-based methods, we draw validation sets σ_j such that $|\sigma_j| = |U_i|$. For SQS-H, we generate $m = 1,000$ such sets. For SQS-S, we first draw $n = 10,000$ candidate sets and then retain the closest $m = \lceil \gamma n \rceil$ sets, with $\gamma = 0.1$, thus obtaining $m = 1,000$ estimates. Our implementation of BayCAP builds on the BayesianCC implementation of [33], available in QuaPy. This implementation uses NUTS [13] as its sampler; we run 500 warm-up iterations before collecting posterior samples. In all experiments, we use uniform Dirichlet priors, i.e., Dirichlet distributions with all concentration parameters equal to one. As the surrogate quantifier in our methods, we adopt PACC [1] as implemented on the QuaPy⁶ package [22], since PACC is an efficient algorithm that requires no hyperparameter tuning.

6.1 Results

Table 1 summarizes the experimental results in terms of average ranks. For each classifier and dataset, methods are ranked according to their mean AE and Winkler score ($\mathcal{W}$) across the APP test sets; the table then reports the average rank across the 15 datasets, separately for each classifier. Since raw AE and

⁶ <https://github.com/HLT-ISTI/QuaPy>

Winkler scores are not directly comparable across datasets, average ranks provide a more compact comparison of point-estimation performance and prediction-interval quality; lower ranks are better. Results for L-CBPE are not included in this summary because L-CBPE was evaluated only on the binary datasets (5 out of 15). The full table of results is reported in Appendix A.

Table 1. Evaluation of point estimation (AE) and predicted interval quality ($\mathcal{W}$) in terms of macro-averaged ranks; lower ranks are better. Boldface indicates the best result for a given row and measure; superscript $\dagger$ denotes results (if any) that are not statistically significantly different from the best one according to a Wilcoxon signed-rank test at significance level 0.05. Cells are colour coded to ease visual inspection, with darker green indicating better results.

	AE						$\mathcal{W}$					
	DoC	QuAcc	LEAP	SQS-H	SQS-S	BayCAP	DoC	QuAcc	LEAP	SQS-H	SQS-S	BayCAP
LR	3.27 [†]	4.80	2.53 [†]	2.33	3.47 [†]	4.60	4.27	4.80	3.93	2.33	2.60 [†]	3.07 [†]
kNN	3.00 [†]	5.00	2.67	2.67 [†]	4.00 [†]	3.67	4.20	5.00	4.00	1.87	3.13	2.80 [†]
SVM	3.93	4.93	2.07	2.53 [†]	3.80	3.73	4.87	4.53	3.53	2.00	2.80 [†]	3.27 [†]
MLP	2.80 [†]	5.73	2.93 [†]	2.20	3.93	3.40	3.80	5.07	4.00	2.13	3.00 [†]	3.00 [†]
RF	2.73 [†]	4.73	2.73 [†]	2.60	4.07	4.13	3.93	4.27	3.93	2.47	3.13 [†]	3.27 [†]

The results show that SQS-H is the most competitive of the proposed methods for point estimation. It obtains the best average rank, although the statistical significance tests indicate that DoC and, especially, LEAP also belong to the group of best-performing methods. The full results in Table 2 in Appendix A show that SQS-H is competitive (i.e., it either obtains the best result or is not statistically significantly different from the best) in 43 out of 75 classifier-dataset combinations. This places it ahead of LEAP and DoC, which obtain 38 and 29 such results, respectively.

SQS-H also obtains the strongest results in terms of interval quality, achieving the best average rank. The statistical significance tests indicate that SQS-S and, especially, BayCAP also belong to the group of best-performing methods. In terms of Winkler score, SQS-H is competitive (i.e., it either obtains the best result or is not statistically significantly different from the best) in 45 out of 80 classifier-dataset combinations, as shown in Table 2 in Appendix A. This substantially outperforms all competing methods. Although SQS-S and BayCAP are less competitive in terms of point estimation, their strong Winkler scores suggest that they can model uncertainty effectively, even when their central estimates are not always the most accurate. Overall, SQS-H achieves the best balance between point-estimation accuracy and interval quality.

It may seem surprising that SQS-S underperforms SQS-H, since SQS-S was designed to mitigate one of the limitations of SQS-H, namely that SQS-H compares true validation prevalence values with an estimated target prevalence. However, SQS-S does so by sacrificing part of the available labelled data. This creates a trade-off between a more sophisticated matching criterion and a less accurate quantifier, trained on less labelled information.

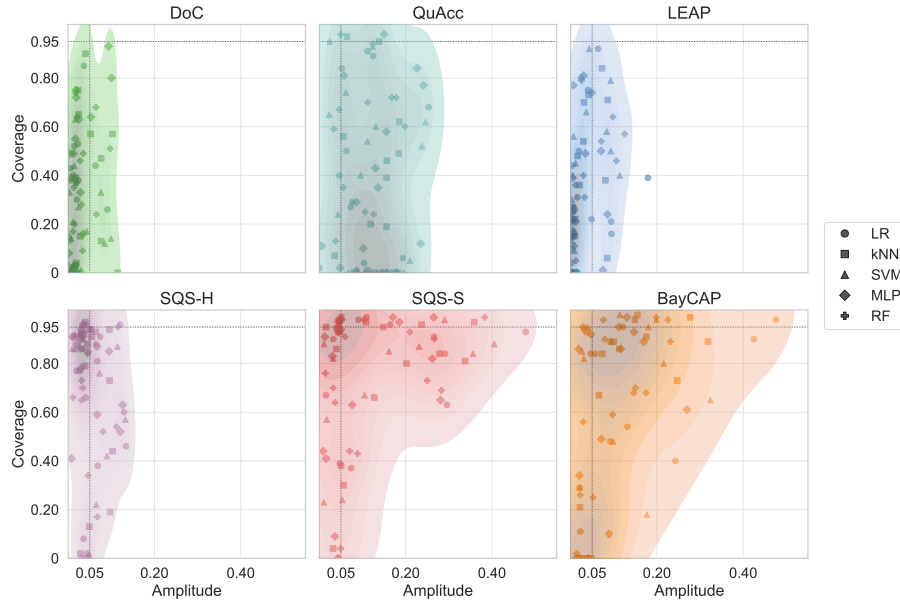


Fig. 1. Relationship between coverage-amplitude across different methods. Each point represents the average amplitude (x-axis) and coverage (y-axis) obtained by one method on one classifier-dataset combination across the 100 test sets generated via APP. Facets correspond to CAP methods, and markers distinguish the target classifier algorithms. The shaded regions show a two-dimensional kernel density estimate of the points in each facet. The horizontal dotted line marks the nominal coverage level 0.95, while the vertical dotted line marks a reference amplitude value established at 0.05. Points close to the intersection of both lines are desirable.

As explained in Section 6, L-CBPE is evaluated only on binary datasets. Although it obtains competitive point estimates in a few cases, its Winkler scores are poor. This behaviour is consistent with CBPE’s dependence on calibrated posterior probabilities for the target distribution, a condition that is difficult to guarantee under PPS even when using LasCal.

Figure 1 provides a complementary view of the quality of the interval estimates by relating their amplitude to their coverage.⁷ The plot clarifies an aspect not immediately visible from the Winkler scores alone. Among the baselines, DoC and LEAP tend to produce intervals with relatively small amplitude, often close to the reference value that we establish at 0.05, but their coverage is typically far below the nominal 0.95 level. QuAcc exhibits the same under-coverage issue,

⁷ L-CBPE is not reported in Figure 1. Its results are less informative in this visualization, since they are available only for binary datasets and, in most cases, the method returns intervals with very small amplitude and, as a consequence, very low coverage, often close to zero.

while also producing intervals with larger amplitudes. This indicates that, although some bootstrap-based baselines may obtain competitive Winkler scores, their intervals are often not well calibrated as 95% interval estimates.

The proposed uncertainty-aware methods display a different behaviour. SQS-S and BayCAP generally achieve high coverage, but often at the cost of wide intervals. SQS-H instead shows a more balanced trade-off, with most results concentrated in regions of high coverage and moderate amplitude. This confirms the trend observed in Table 2: among the proposed methods, SQS-H is the one that most consistently balances accurate point estimates with reliable and informative interval estimates.

As a final remark, BayCAP has two features that are not exploited in our experimental setting. First, among the methods considered here, it is the only one that naturally allows prior knowledge to be incorporated. In our experiments, we use a uniform prior; however, in applications where prior knowledge about plausible target prevalence values is available, more informative priors could provide an advantage. Second, the Bayesian formulation allows the spread of the posterior distribution to be adjusted through a temperature parameter. The current implementation uses temperature equal to one, but this parameter could be tuned on validation sets to improve the calibration of the resulting uncertainty estimates. We leave the investigation of both directions to future work.

7 Conclusion

In this paper we have addressed classifier accuracy prediction under prior probability shift from an uncertainty-aware perspective. We have proposed three methods (SQS-H, SQS-S, BayCAP) that natively provide both point estimates of classifier performance and interval estimates around them. Our empirical comparison with existing CAP methods shows that SQS-H is the most effective among the evaluated methods, both in terms of point-estimation accuracy and uncertainty assessment. Among the proposed methods, BayCAP is, to the best of our knowledge, the first Bayesian approach to CAP. Although our experiments do not show BayCAP to outperform the other methods considered, its Bayesian formulation offers additional flexibility that remains unexplored in this work, i.e., the possibility of incorporating informative priors and calibrating posterior uncertainty through tempering. In future work, we plan to investigate these directions, as well as to extend the analysis to other types of dataset shift and to evaluation measures beyond vanilla accuracy.

Bibliography

- [1] Bella, A., Ferri, C., Hernández-Orallo, J., Ramírez-Quintana, M.J.: Quantification via probability estimators. In: Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2010). pp. 737–742. Sydney, AU (2010).
- [2] Bhaskaruni, D., Moss, F.P., Lan, C.: Estimating prediction qualities without ground truth: A revisit of the reverse testing framework. In: Proceedings of the

- 24th International Conference on Pattern Recognition (ICPR 2018). pp. 49–54. Beijing, CN (2018).
- [3] Chen, J., Liu, F., Avci, B., Wu, X., Liang, Y., Jha, S.: Detecting errors and estimating accuracy on unlabeled data with self-training ensembles. In: Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021). pp. 14980–14992. Virtual Event (2021)
 - [4] Deng, W., Suh, Y., Gould, S., Zheng, L.: Confidence and dispersity speak: Characterizing prediction matrix for unsupervised accuracy estimation. In: Proceedings of the 40th International Conference on Machine Learning (ICML 2023). pp. 7658–7674. Honolulu, US (2023)
 - [5] Deng, W., Zheng, L.: AutoEval: Are labels always necessary for classifier accuracy evaluation? In: Proceedings of the 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2021). pp. 15069–15078. Virtual Event (2021).
 - [6] Elsahar, H., Gallé, M.: To annotate or not? Predicting performance drop under domain shift. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019). pp. 2163–2173. Hong Kong, CN (2019).
 - [7] Esuli, A., Fabris, A., Moreo, A., Sebastiani, F.: Learning to quantify. Springer Nature, Cham, CH (2023).
 - [8] Forman, G.: Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery* **17**(2), 164–206 (2008).
 - [9] Garg, S., Balakrishnan, S., Lipton, Z.C., Neyshabur, B., Sedghi, H.: Leveraging unlabeled data to predict out-of-distribution performance. In: Proceedings of the 10th International Conference on Learning Representations (ICLR 2022). Virtual Event (2022)
 - [10] Gneiting, T., Raftery, A.E.: Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association* **102**(477), 359–378 (2007).
 - [11] Guillory, D., Shankar, V., Ebrahimi, S., Darrell, T., Schmidt, L.: Predicting with confidence on unseen distributions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2021). pp. 1134–1144. Montreal, CA (2021).
 - [12] Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: Proceedings of the 5th International Conference on Learning Representations (ICLR 2017). Toulon, FR (2017)
 - [13] Hoffman, M.D., Gelman, A.: The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research* **15**(1), 1593–1623 (2014).
 - [14] Hofman, P., Sale, Y., Hüllermeier, E.: Uncertainty quantification for machine learning: One size does not fit all. In: Proceedings of the 40th AAAI Conference on Artificial Intelligence (AAAI 2026). pp. 21726–21734. Singapore, SN (2026).
 - [15] Jiang, Y., Nagarajan, V., Baek, C., Kolter, J.Z.: Assessing generalization of SGD via disagreement. In: Proceedings of the International Conference on Learning Representations (ICLR 2022). Virtual Event (2022)
 - [16] Kelly, M., Longjohn, R., Nottingham, K.: The UCI machine learning repository, <https://archive.ics.uci.edu>
 - [17] Kivimäki, J., Bialek, J., Kuberski, W., Nurminen, J.K.: Performance estimation in binary classification using calibrated confidence. *Machine Learning* **115**(3), 67 (2026).

- [18] Kivimäki, J., Nurminen, J.K., Białek, J., Kuberski, W.: Confidence-based estimators for predictive performance in model monitoring. *Journal of Artificial Intelligence Research* **82**, 209–240 (2025).
- [19] Lipton, Z.C., Wang, Y., Smola, A.J.: Detecting and correcting for label shift with black box predictors. In: *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*. pp. 3128–3136. Stockholm, SE (2018)
- [20] Lu, Y., Qin, Y., Zhai, R., Shen, A., Chen, K., Wang, Z., Kolouri, S., Stepputtis, S., Campbell, J., Sycara, K.P.: Characterizing out-of-distribution error via optimal transport. In: *Proceedings of the 36th Annual Conference on Neural Information Processing Systems (NeurIPS 2023)*. New Orleans, US (2023)
- [21] Moreno-Torres, J.G., Raeder, T., Alaíz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. *Pattern Recognition* **45**(1), 521–530 (2012).
- [22] Moreo, A., Esuli, A., Sebastiani, F.: QuaPy: A Python-based framework for quantification. In: *Proceedings of the 30th ACM International Conference on Knowledge Management (CIKM 2021)*. pp. 4534–4543. Gold Coast, AU (2021).
- [23] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011)
- [24] Popordanoska, T., Radevski, G., Tuytelaars, T., Blaschko, M.B.: Lascal: Label-shift calibration without target labels. In: *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*. pp. 65386–65414. Vancouver, CA (2024)
- [25] Quiñero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D. (eds.): *Dataset shift in machine learning*. The MIT Press, Cambridge, US (2009).
- [26] Redyuk, S., Schelter, S., Rukat, T., Markl, V., Bießmann, F.: Learning to validate the predictions of black box machine learning models on unseen data. In: *Proceedings of the Workshop on Human-In-the-Loop Data Analytics (HILDA@SIGMOD 2019)*. pp. 4:1–4:4. Amsterdam, NL (2019).
- [27] Tasche, D.: Comments on Friedman’s method for class distribution estimation. *arXiv:2405.16666 [cs.LG]* (2024)
- [28] Volpi, L., Moreo, A., Sebastiani, F.: LEAP: Linear equations for classifier accuracy prediction under prior probability shift. *Machine Learning* **114**(12), Article 293 (2025).
- [29] Volpi, L., Moreo, A., Sebastiani, F.: QuAcc: Using quantification to predict classifier accuracy under prior probability shift. *Intelligenza Artificiale* **19**(2), 141–157 (2025).
- [30] Wang, L., Wang, X., Liao, K.P., Cai, T.: Semisupervised transfer learning for evaluation of model classification performance. *Biometrics* **80**(1) (2024).
- [31] Xie, R., Wei, H., Feng, L., Cao, Y., An, B.: On the importance of feature separability in predicting out-of-distribution error. In: *Proceedings of the 36th Annual Conference on Neural Information Processing Systems (NeurIPS 2023)*. pp. 27783–27800. New Orleans, US (2023)
- [32] You, S.D., Liu, H., Liu, C.: Predicting classification accuracy of unlabeled datasets using multiple deep neural networks. *IEEE Access* **10**, 44627–44637 (2022).
- [33] Ziegler, A., Czyż, P.: Bayesian quantification with black-box estimators. *Transactions on Machine Learning Research* **2024** (2024)

A Full table of results

Table 2 reports the full set of experiments in terms of AE and Winkler score ($\mathcal{W}$); lower is better for both measures. Boldface indicates the best result for a given row and measure; superscript $\dagger$ denotes results that are not statistically significantly different from the best one according to a Wilcoxon signed-rank test at significance level 0.01. The cell colouring is meant to ease visual inspection. For each evaluation measure and each classifier-dataset combination, colours are assigned locally within the corresponding row. Specifically, we consider the standard deviation for the distribution of scores obtained by all methods across all test sets. The method with the best mean score in the row is coloured green. Methods whose mean score differs from the best mean score by more than one standard deviation are coloured red, while intermediate cases are coloured according to a green-to-red gradient, based on their distance from the best mean.

Table 2. Results of the CAP experiments in terms of AE and Winkler score ($\mathcal{W}$).

		AE							W						
		Baseline methods				Our methods			Baseline methods				Our methods		
		DoC	QuAcc	LEAP	L-CBPE	SQS-H	SQS-S	BayCAP	DoC	QuAcc	LEAP	L-CBPE	SQS-H	SQS-S	BayCAP
LR	spambase	0.014	0.031	0.006 [†]	0.030	0.006	0.008 [†]	0.007	0.299	0.438	0.206	36.623	0.036	0.054	0.038
	semeion	0.009	0.030	0.037	0.083	0.043	0.036	0.041	0.063	0.314	1.199	33.885	1.009	0.643	0.127
	german	0.040	0.176	0.055 [†]	0.079	0.057	0.050 [†]	0.078	0.674 [†]	3.411	1.351	25.152	1.233	0.779	0.579
	tictactoe	0.004	0.024	0.006	0.013	0.007	0.007	0.011	0.086 [†]	0.946	0.184	38.892	0.066	0.083 [†]	0.053
	mammographic	0.086	0.112	0.059	0.064	0.060	0.070	0.052	3.435	2.734	2.350	30.723	1.529	1.966	0.444
	poker-hand	0.081	0.089	0.146	-	0.087 [†]	0.073	0.246	3.211	2.153	3.185	-	1.445	0.608	0.864
	connect-4	0.079	0.028 [†]	0.071	-	0.026	0.128	0.107	1.730	0.156	1.082	-	0.142	1.677	0.497
	shuttle	0.149	0.083	0.012	-	0.022	0.127	0.127	5.762	2.101	0.078	-	0.118	0.600	0.687
	chess	0.027	0.029	0.015	-	0.021	0.028	0.027	0.768	0.181	0.137	-	0.156	0.180	0.168
	letter	0.013 [†]	0.131	0.013	-	0.010	0.017	0.065	0.228	3.570	0.273	-	0.065	0.102	1.965
	dry-bean	0.010	0.268	0.008	-	0.007	0.010	0.015	0.222	8.282	0.232	-	0.041	0.052	0.247
	nursery	0.021	0.016	0.011	-	0.018	0.019	0.021	0.465	0.120 [†]	0.152 [†]	-	0.197	0.110	0.095
	hand-digits	0.008 [†]	0.313	0.007 [†]	-	0.007	0.008 [†]	0.021	0.132	10.431	0.170	-	0.036	0.053	0.440
	isolat	0.006	0.454	0.013	-	0.009 [†]	0.007 [†]	0.178	0.050	15.473	0.345	-	0.071 [†]	0.047	6.387
	wine-quality	0.100	0.102	0.101	-	0.059	0.092	0.210	3.649	0.967 [†]	2.314	-	0.797 [†]	0.497	3.794
kNN	spambase	0.010	0.028	0.007 [†]	0.021	0.007 [†]	0.012	0.007	0.062	0.347	0.213	34.621	0.046	0.073	0.043
	semeion	0.040	0.310	0.013	0.037	0.013 [†]	0.041	0.029	1.179	8.536	0.276	34.514	0.095	0.677	0.153
	german	0.209	0.153	0.031	0.076	0.034 [†]	0.056	0.099	6.076	3.334	0.340 [†]	22.927	0.306	0.585	0.422
	tictactoe	0.026	0.032	0.030	0.020	0.034	0.035	0.033	0.693	0.577	0.980	35.705	0.820	0.783	0.173
	mammographic	0.030	0.127	0.040	0.108	0.041	0.033	0.034	0.986	3.417	1.587	29.739	0.714	0.502	0.118
	poker-hand	0.071	0.131	0.078	-	0.074	0.051	0.110	2.513	2.563	1.975	-	1.502	0.253	1.490
	connect-4	0.046	0.026 [†]	0.022	-	0.023 [†]	0.061	0.031	0.717	0.159	0.162	-	0.133	0.478	0.155
	shuttle	0.112	0.143	0.014	-	0.011	0.118	0.034	3.092	3.245	0.123 [†]	-	0.055	0.693	0.182
	chess	0.026	0.098	0.016 [†]	-	0.015	0.030	0.022	0.383 [†]	1.768	0.249 [†]	-	0.113	0.174	0.165
	letter	0.007	0.521	0.013	-	0.009 [†]	0.010	0.078	0.067 [†]	17.657	0.345	-	0.046	0.051	2.684
	dry-bean	0.013	0.273	0.010	-	0.008	0.010	0.017	0.229	8.591	0.295	-	0.055	0.065	0.329
	nursery	0.056	0.010	0.010 [†]	-	0.012	0.075	0.034	0.851	0.073	0.150 [†]	-	0.071	0.474	0.183
	hand-digits	0.003	0.655	0.004 [†]	-	0.003 [†]	0.003 [†]	0.022	0.068	23.002	0.101	-	0.018	0.019	0.631
	isolat	0.010	0.308	0.015	-	0.011 [†]	0.020	0.159	0.112 [†]	9.885	0.309	-	0.061	0.122	5.455
	wine-quality	0.071	0.088	0.121	-	0.108	0.095	0.053	2.337	1.171 [†]	3.073	-	2.603	0.381	0.281
SVM	spambase	0.018	0.038	0.008	0.031	0.008 [†]	0.009 [†]	0.008	0.616	0.748	0.254	36.661	0.044	0.071 [†]	0.058 [†]
	semeion	0.068	0.022	0.008	0.066	0.007 [†]	0.007	0.016	1.934	0.193	0.159	36.596	0.046	0.043	0.114
	german	0.091	0.254	0.042 [†]	0.164	0.066	0.039	0.159	2.106	6.394	0.593 [†]	22.955	1.262	0.413	2.456
	tictactoe	0.013	0.007	0.004	0.022	0.004 [†]	0.016	0.005	0.310	0.116	0.073	39.050	0.026	0.442	0.042
	mammographic	0.056	0.054 [†]	0.062	0.077	0.063	0.061 [†]	0.054	1.661	0.674	2.387	29.609	1.609	1.527	0.398
	poker-hand	0.079	0.132	0.052	-	0.082	0.053 [†]	0.079 [†]	2.934	1.920	0.748 [†]	-	2.070	0.293	1.126
	connect-4	0.055	0.056	0.033 [†]	-	0.031	0.063	0.040 [†]	1.005	0.527	0.236 [†]	-	0.184	0.432	0.261
	shuttle	0.119	0.074	0.012	-	0.013	0.101	0.031	2.981	1.375	0.094	-	0.086	0.524	0.200
	chess	0.035	0.025	0.016	-	0.024	0.041	0.022	0.998	0.144	0.185 [†]	-	0.223 [†]	0.294	0.143
	letter	0.008	0.443	0.014	-	0.008 [†]	0.009 [†]	0.083	0.097 [†]	15.119	0.406	-	0.050	0.052	2.880
	dry-bean	0.009	0.323	0.008	-	0.007	0.009	0.014	0.174	8.525	0.227	-	0.038	0.053	0.243
	nursery	0.008	0.005	0.003	-	0.005	0.010	0.005	0.256	0.030	0.039 [†]	-	0.035	0.136	0.035 [†]
	hand-digits	0.003	0.480	0.003 [†]	-	0.003 [†]	0.003 [†]	0.021	0.058	18.018	0.067	-	0.019	0.023	0.614
	isolat	0.013	0.435	0.013	-	0.008	0.010	0.178	0.235	15.143	0.343	-	0.055	0.055	6.404
	wine-quality	0.109	0.111	0.077	-	0.065	0.104	0.149	4.265	1.313 [†]	1.323	-	0.785 [†]	0.527	1.059 [†]
MLP	spambase	0.009	0.017	0.006 [†]	0.043	0.006 [†]	0.009	0.006	0.133	0.135	0.199	37.105	0.034	0.066 [†]	0.036
	semeion	0.013	0.067	0.035	0.079	0.042	0.055	0.036	0.168 [†]	1.656	1.141	34.740	1.072	1.503	0.121
	german	0.021 [†]	0.067	0.017	0.056	0.029	0.035	0.042	0.115	0.416	0.160 [†]	25.580	0.314 [†]	0.342 [†]	0.163
	tictactoe	0.005 [†]	0.028	0.004	0.017	0.005 [†]	0.012	0.007	0.116	0.968	0.104	38.707	0.031	0.195	0.049
	mammographic	0.085	0.085	0.076	0.075	0.078	0.080	0.071	3.009	1.467	2.929	30.551	2.278	2.337	1.012
	poker-hand	0.062	0.088	0.099	-	0.052 [†]	0.050 [†]	0.041	1.875	0.902	2.247	-	0.665 [†]	0.268	0.270
	connect-4	0.033 [†]	0.029 [†]	0.035 [†]	-	0.027	0.037	0.028 [†]	0.247	0.161	0.359 [†]	-	0.179	0.198	0.181
	shuttle	0.024	0.144	0.009	-	0.014	0.092	0.017	0.253	5.075	0.074	-	0.088	0.543	0.131
	chess	0.022 [†]	0.082	0.020 [†]	-	0.018	0.026 [†]	0.034	0.471	1.072	0.351 [†]	-	0.128	0.175	0.380
	letter	0.006	0.651	0.011	-	0.007	0.008	0.080	0.057	22.997	0.297	-	0.046	0.050	2.782
	dry-bean	0.007	0.137	0.008 [†]	-	0.008 [†]	0.011	0.014	0.121	3.739	0.242	-	0.048	0.061	0.231
	nursery	0.009	0.105	0.002	-	0.008	0.009	0.003	0.320	2.062	0.016	-	0.176	0.180	0.026
	hand-digits	0.003 [†]	0.506	0.003	-	0.002	0.003 [†]	0.022	0.046	17.027	0.081	-	0.015	0.015	0.651
	isolat	0.007	0.449	0.014	-	0.009 [†]	0.009 [†]	0.177	0.058	15.271	0.361	-	0.085	0.052	6.332
	wine-quality	0.112	0.200	0.169	-	0.072	0.125	0.139	4.323	3.706	5.246	-	1.226 [†]	1.018	1.674
RF	spambase	0.012	0.018	0.012 [†]	0.016 [†]	0.012 [†]	0.013 [†]	0.014 [†]	0.439	0.150	0.420	37.277	0.087	0.094	0.132
	semeion	0.020	0.031	0.008	0.031	0.011	0.054	0.020	0.420	0.293	0.071	31.303	0.077	1.074	0.198
	german	0.028	0.058	0.037	0.093	0.044	0.052	0.093	0.373	0.462	0.524	24.530	0.676	0.715	0.502
	tictactoe	0.008	0.018	0.009 [†]	0.017	0.012	0.014	0.009 [†]	0.102 [†]	0.282 [†]	0.176	36.179	0.136 [†]	0.154 [†]	0.077
	mammographic	0.049 [†]	0.046	0.055	0.090	0.055	0.048 [†]	0.048 [†]	1.365	0.357 [†]	2.182	29.691	1.286	0.986	0.246
	poker-hand	0.078 [†]	0.099 [†]	0.063	-	0.095	0.067 [†]	0.084 [†]	2.851	1.185 [†]	0.817 [†]	-	2.508	0.360	1.629 [†]
	connect-4	0.073	0.028 [†]	0.042	-	0.023	0.087	0.042	1.857	0.146	0.346 [†]	-	0.143	0.815	0.247
	shuttle	0.020	0.048	0.031	-	0.037	0.125	0.058	0.286 [†]	1.205	0.703	-	0.684	1.062	0.228
	chess	0.016	0.106	0.021 [†]	-	0.017 [†]	0.018 [†]	0.035	0.347	2.076	0.402	-	0.102	0.115	0.557
	letter	0.010	0.574	0.014	-	0.008	0.008 [†]	0.084	0.197	19.554	0.422	-	0.039	0.048	2.931
	dry-bean	0.013	0.127	0.009	-	0.008	0.011	0.016	0.259	2.946	0.260	-	0.052	0.066	0.289
	nursery	0.034	0.007 [†]	0.006	-	0.009	0.047	0.016	0.846	0.050	0.057	-	0.047	0.260	0.095
	hand-digits	0.006	0.563	0.003	-	0.005	0.006	0.018	0.181	20.899	0.069 [†]	-	0.067	0.052	4.059
	isolat	0.016	0.524	0.021	-	0.013	0.014 [†]	0.180	0.431	18.006	0.641	-	0.119 [†]	0.076	6.432
	wine-quality														

Uncertainty-aware Finite-sample Quantification

Clemens Damke^{1,2} (✉) and Eyke Hüllermeier^{1,2,3}

¹ LMU Munich, Germany {clemens.damke, eyke}@ifi.lmu.de

² Munich Center for Machine Learning (MCML)

³ German Centre for Artificial Intelligence (DFKI, DSA)

Abstract A *quantification learning* (QL) method estimates the label distribution of unlabeled data. Existing methods typically target the population label distribution, even though many applications and common evaluation protocols ask for the label distribution realized in a particular finite sample. This distinction matters for small target samples, where the finite-sample label distribution can differ substantially from the population label distribution. We formalize this estimand as finite-sample quantification and propose FS-UP-ACC, a Bayesian hard-classifier-based quantifier that assigns posterior mass directly to feasible finite target counts while propagating calibration uncertainty. The method retains the correction logic of *Adjusted Classify & Count* (ACC), admits a closed-form binary update, and avoids the continuous posterior over population label distributions defined by existing Bayesian black-box-estimator-based quantifiers. Synthetic and real-data experiments show that FS-UP-ACC improves point estimates and uncertainty intervals, especially for small target samples.

Keywords: Quantification learning · Bayesian estimation

1 Introduction

Quantification learning (QL) is the task of estimating the label distribution in an unlabeled sample $\mathcal{X}_U$, given a set of labeled training data $\mathcal{D}_L$. The term “quantification” was coined by Forman [6, 7, 8], who introduced it as a *machine learning* problem distinct from standard *classification* or *regression*. Estimating the label distribution in an unlabeled sample of feature vectors can be interpreted as a dataset-level prediction problem, as opposed to the instance-level predictions in classification. Practical applications of QL include, for instance, estimating disease prevalence in a population, sentiment distributions in collections of reviews, or topic distributions in collections of documents. Esuli et al. [4] provide a comprehensive overview of the field.

A naive approach to QL is to reduce it to classification by applying a classifier to the unlabeled data and counting the predicted labels, an approach known as *Classify & Count* (CC). Given a perfect classifier, this approach is plausible, because it would yield the true label distribution. However, in practice, classifiers are rarely perfect and their errors can lead to biased estimates of the label distribution. The bias of CC is ultimately induced by the fundamentally distinct

objectives of classification and quantification — while (binary) classification aims to minimize the total number of misclassified instances, i.e., $(\text{FP} + \text{FN})$, quantification aims to minimize the distance between the true and estimated label distributions, i.e., $|\text{FP} - \text{FN}|$. As a consequence, even poor classifiers can yield good or even perfect quantification performance if false positives and false negatives are balanced, while a good (but not perfect) classifier may fail to provide accurate quantification estimates.

To make the QL problem tractable, some assumptions about the relationship between the source measure P and the target measure Q are necessary. Without such assumptions, the QL problem is ill-posed, as the training data $\mathcal{D}_L$ would be uninformative about the label distribution of the target data $\mathcal{X}_U$. A common assumption in QL is the so-called *prior probability shift* (PPS) assumption, which states that the class-conditional distributions $P(X | Y)$ and $Q(X | Y)$ are identical, while the class priors $P(Y)$ and $Q(Y)$ may differ.

A second common assumption in the QL literature is that the target quantity of interest is the population label distribution $Q(Y)$, as opposed to the label distribution realized in the finite target sample. This is a natural estimand whenever the unlabeled sample is viewed as a finite observation of a larger target population: for example, when estimating disease prevalence in a region, the long-run sentiment distribution among reviews submitted to a platform, or the distribution of topics in articles published by a news outlet. This population-level question has been studied extensively in quantification learning and label-shift estimation. Under the PPS assumption, many Fisher-consistent estimators of the target-population label distribution $Q(Y)$ given $P(X, Y)$ and $Q(X)$ have been proposed [14, 19, 22], including the *Adjusted Classify & Count* (ACC) estimator [2, 6, 8, 14], *Probabilistic Adjusted Classify & Count* (PACC) [1], the *Expectation-Maximization* method [18] and, more recently, *Bayesian Classify & Count* (BCC) [24]. These methods differ in how they estimate $Q(Y)$ and in how they represent uncertainty, but their inferential target remains the population label distribution.

Apart from estimating $Q(Y)$, there is a less well-studied but equally natural second quantification problem. In many applications, the unlabeled sample itself is the object of interest: for example, when estimating the sentiment distribution in a given set of product reviews, the topic distribution in a specific document collection, or the prevalence of a condition in a particular tested cohort. Then the relevant quantity is not the population label distribution $Q(Y)$, but the label distribution realized in that particular finite target sample. Writing $Y_1^U, \dots, Y_m^U$ for the latent labels of the unlabeled sample, this finite-sample target is the empirical label distribution

$$S_y = \frac{1}{m} \sum_{j=1}^m \mathbf{1}[Y_j^U = y] \quad \forall y \in \mathcal{Y}.$$

Even under i.i.d. sampling from Q , S_y need not equal the population probability $q_Y(y) = Q(Y = y)$ for finite samples due to sampling variation and quantization effects.

The finite-sample problem has received comparatively little direct attention, despite being implicit in common experimental protocols. For instance, the widely used *artificial prevalence protocol* (APP) evaluates quantifiers by repeatedly drawing finite target samples and comparing estimates against the realized label distribution in each sample, not against the population prevalence from which the samples were drawn. This creates a mismatch between the estimand targeted by much of the theory and the quantity used in empirical evaluation, especially when target samples are small. In this paper, we focus on this second estimand and refer to the corresponding task as *finite-sample quantification*.

2 Quantification Learning

2.1 Problem Setting

Let $\mathcal{X}$ denote the feature space and let $\mathcal{Y} = \{1, \dots, C\}$ denote a finite label space. We consider two probability measures P and Q on $\mathcal{X} \times \mathcal{Y}$. The measure P describes the labeled source population, while Q describes the target population. Let X and Y denote the canonical random variables that project from $\mathcal{X} \times \mathcal{Y}$ to the feature and label spaces, respectively. We write $P(X)$, $P(Y)$, $Q(X)$, and $Q(Y)$ for the corresponding pushforward measures and denote their densities or *probability mass functions* (PMFs) by p_X , p_Y , q_X , and q_Y , respectively.

The observed labeled data are a realized sample $\mathcal{D}_L$, modeled as an outcome of the random vector

$$\mathbf{D}_L = ((X_1^L, Y_1^L), \dots, (X_n^L, Y_n^L)), \quad (X_i^L, Y_i^L) \stackrel{\text{iid}}{\sim} P.$$

Likewise, the target data are modeled by the complete random vector

$$\mathbf{D}_U = ((X_1^U, Y_1^U), \dots, (X_m^U, Y_m^U)), \quad (X_j^U, Y_j^U) \stackrel{\text{iid}}{\sim} Q.$$

Only the feature vector is observed,

$$\mathbf{X}_U = (X_1^U, \dots, X_m^U), \quad \mathbf{Y}_U = (Y_1^U, \dots, Y_m^U),$$

where $\mathbf{Y}_U$ is latent. The realized value of $\mathbf{X}_U$ is denoted by $\mathcal{X}_U = (x_1^U, \dots, x_m^U)$.

Population quantification targets the target-population label distribution $Q(Y)$, or equivalently its PMF $q_Y \in \Delta_C$, where Δ_C is the probability simplex over C classes. **Finite-sample quantification** instead targets the label distribution realized in the target sample. We denote this random vector by $\mathbf{S} = (S_y)_{y \in \mathcal{Y}}$, where

$$S_y = \frac{1}{m} \sum_{j=1}^m \mathbf{1}[Y_j^U = y] \quad \forall y \in \mathcal{Y}. \quad (1)$$

Thus, $\mathbf{S}$ takes values in the finite simplex lattice

$$\Delta_{m,C} = \left\{ s \in [0, 1]^C : m \cdot s \in (\mathbb{N}_0)^C, \sum_{y \in \mathcal{Y}} s_y = 1 \right\}.$$

Under i.i.d. sampling from Q , it holds that $\mathbb{E}[S_y] = q_Y(y)$, but it is generally not true that $S_y = q_Y(y)$ for finite m . The two estimands therefore differ even when the sampling model is correctly specified.

In the following, we use uppercase symbols for random quantities and lowercase symbols for their realizations. Thus, $\mathbf{S}$ denotes the random finite-sample label distribution induced by $\mathbf{D}_U$, while $\mathbf{s}$ denotes its realized value for a particular target sample. Likewise, $\hat{\mathbf{S}}$ denotes the random output of a **finite-sample quantifier** when applied to random samples $\mathbf{D}_L$ and $\mathbf{X}_U$, whereas $\hat{\mathbf{s}}$ denotes the estimate obtained for observed samples $\mathcal{D}_L$ and $\mathcal{X}_U$. A **population quantifier** instead returns an estimate $\hat{q}_Y$ of q_Y . **Aggregative quantifiers** additionally have access to a hypothesis $h : \mathcal{X} \rightarrow \mathcal{Y}$ trained on a separate labeled sample.

2.2 Population Label Distribution Estimation

Most quantification methods are formulated as estimators of the population label distribution $Q(Y)$. We recall the two population-level constructions that are most relevant for the method developed in Section 3. Let $h : \mathcal{X} \rightarrow \mathcal{Y}$ be a fixed classifier and write $\hat{Y} = h(X)$ for the induced predicted-label random variable. For the observed target feature vectors $\mathcal{X}_U$, the realized predicted-label counts are

$$m_y = \sum_{j=1}^m \mathbb{1}[h(x_j^U) = y] \quad \forall y \in \mathcal{Y}. \quad (2)$$

The corresponding (potentially biased) *Classify & Count* (CC) estimate of the target-population probability of class y is $\hat{q}_{Y,CC}(y) = m_y/m$.

Adjusted Classify & Count (ACC). ACC corrects CC by accounting for the classifier’s class-conditional error rates [6, 8]. It relies on the *prior probability shift* (PPS) assumption (also commonly referred to as *label shift*), which states that the source and target class-conditional feature distributions agree,

$$P(X \mid Y = y) = Q(X \mid Y = y) \quad \forall y \in \mathcal{Y}, \quad (3)$$

while the label marginals $P(Y)$ and $Q(Y)$ may differ [14, 19, 22]. For a fixed binary classifier h , Eq. (3) implies equality of the class-conditional probabilities

$$\begin{aligned} a &= P(\hat{Y} = 1 \mid Y = 1) = Q(\hat{Y} = 1 \mid Y = 1), \\ b &= P(\hat{Y} = 1 \mid Y = 0) = Q(\hat{Y} = 1 \mid Y = 0). \end{aligned} \quad (4)$$

Here, a is the true-positive rate and b is the false-positive rate. Consequently, the population predicted-positive probability under Q satisfies

$$Q(\hat{Y} = 1) = a \cdot q_Y(1) + b \cdot q_Y(0) = b + q_Y(1)(a - b).$$

Replacing $Q(\hat{Y} = 1)$ by the observed fraction m_1/m and a, b by calibration estimates $\hat{a}, \hat{b}$ yields the binary ACC estimator

$$\hat{q}_{Y, \text{ACC}}(1) = \frac{m_1/m - \hat{b}}{\hat{a} - \hat{b}}. \quad (5)$$

Thus, binary ACC is a plug-in population-prevalence estimator: its target is $q_Y(1)$, not the realized finite-sample prevalence S_1 . Multiclass variants of ACC were proposed by Vucetic and Obradovic [23], Firat [5] and Bunse [2].

Bayesian Classify & Count (BCC). Ziegler and Czyż [24] propose a Bayesian formulation of quantification with black-box estimators. Akin to ACC, BCC avoids modeling the generally high-dimensional distribution of X directly and instead uses the finite classifier output $\hat{Y} = h(X)$. Under PPS, the class-conditional behavior of the classifier can be represented by the shared conditional PMFs

$$p_{\hat{Y}|Y=y}(\hat{y}) = P(\hat{Y} = \hat{y} \mid Y = y) = Q(\hat{Y} = \hat{y} \mid Y = y) \quad \forall y, \hat{y} \in \mathcal{Y}. \quad (6)$$

This model can be viewed as replacing the intractable posterior over target label distributions based on raw features by an approximate posterior based on classifier outputs. For a candidate target label distribution q_Y , the induced target distribution of predicted labels is

$$q_{\hat{Y}}(\hat{y}) = \sum_{y \in \mathcal{Y}} q_Y(y) p_{\hat{Y}|Y=y}(\hat{y}) \quad \forall \hat{y} \in \mathcal{Y}. \quad (7)$$

The relevant data can be summarized by finite counts. For $\mathcal{D}_L$, define the source label counts and joint label-prediction (i.e., confusion matrix) counts

$$n_y = \sum_{(x, y') \in \mathcal{D}_L} \mathbb{1}[y' = y], \quad n_{y\hat{y}} = \sum_{(x, y') \in \mathcal{D}_L} \mathbb{1}[y' = y \wedge h(x) = \hat{y}]. \quad (8)$$

For the unlabeled target sample, the predicted-label counts $m_{\hat{y}}$ are the counts from Eq. (2), with $\hat{y}$ used to emphasize predicted labels. Here, $\text{Mult}(\cdot; N, \rho)$ denotes the multinomial PMF evaluated at the observed count vector. The sufficient-statistic likelihood contributions can then be written as

$$\ell_Y(p_Y) = \text{Mult}((n_y)_{y \in \mathcal{Y}}; n, p_Y), \quad (9)$$

$$\ell_h(p_{\hat{Y}|Y}) = \prod_{y \in \mathcal{Y}} \text{Mult}((n_{y\hat{y}})_{\hat{y} \in \mathcal{Y}}; n_y, p_{\hat{Y}|Y=y}), \quad (10)$$

$$\ell_U(q_Y, p_{\hat{Y}|Y}) = \text{Mult}((m_{\hat{y}})_{\hat{y} \in \mathcal{Y}}; m, q_{\hat{Y}}). \quad (11)$$

Let $\pi_p(p_Y)$, $\pi_q(q_Y)$, and $\pi_h(p_{\hat{Y}|Y})$ denote the prior densities for the source label distribution, target label distribution, and calibration probabilities, respectively. Then the BCC posterior is

$$\begin{aligned} & \pi_{\text{BCC}}(p_Y, q_Y, p_{\hat{Y}|Y} \mid \mathcal{D}_L, \mathcal{X}_U) \\ & \propto \pi_p(p_Y) \pi_q(q_Y) \pi_h(p_{\hat{Y}|Y}) \ell_Y(p_Y) \ell_h(p_{\hat{Y}|Y}) \ell_U(q_Y, p_{\hat{Y}|Y}). \end{aligned} \quad (12)$$

The prior densities π_p , π_q , and π_h are free parameters in BCC. Since the factor $\pi_p(p_Y)\ell_Y(p_Y)$ is separate from the target-distribution and calibration terms, p_Y is a nuisance quantity for posterior inference on q_Y under this factorization and can be omitted. The reference implementation in QuaPy [16] does this and uses uniform priors over q_Y and over the rows of $p_{\hat{Y}|Y}$. Inference in Eq. (12) yields a posterior over the target-population label distribution q_Y . Thus, BCC represents uncertainty about the target-population label distribution; note that the realized finite-sample distribution $\mathbf{S}$ from Eq. (1) is *not* the direct posterior target. In practice, posterior samples can be obtained using *Hamiltonian Monte Carlo* (HMC) algorithms such as the *No-U-Turn sampler* (NUTS) [10].

2.3 Prior Work on Finite-sample Quantification

Although most work in QL and label-shift estimation targets $q_Y = Q(Y)$, the distinction between population and finite-sample targets is visible in parts of the literature. Hopkins and King [11] emphasize aggregate proportions in collections of documents rather than individual document labels, although their estimand is naturally interpreted as a population or collection-level quantity. Tasche [20] explicitly distinguishes confidence intervals for population class prevalences from prediction intervals for finite-sample relative frequencies under PPS. This distinction is close to the one made here: population quantification concerns $Q(Y)$, whereas finite-sample quantification concerns the random vector $\mathbf{S}$ from Eq. (1).

The closest finite-sample approaches are uncertainty-aware prevalence estimators that define the target as the class proportion in a specific group or target sample. Keith and O’Connor [13] model document-level prevalence for a fixed group and construct uncertainty intervals for the group prevalence. Igarane et al. [12] similarly distinguish a prior prevalence parameter from the target-sample prevalence and perform Bayesian inference for prevalence with uncertainty. These works show that finite-sample prevalence is a meaningful inferential target in its own right. However, they do not develop the particular finite-sample formulation pursued in this paper, namely an estimator that assigns probability mass directly to the realizable values of $\mathbf{S}$ while using the PPS-based correction logic of ACC.

3 A Bayesian Finite-sample Quantification Method

We now develop a finite-sample analog of the Bayesian population model from Section 2.2. The guiding idea is to keep the ACC correction logic, but to change the posterior target from the population label distribution q_Y to the realized finite-sample label distribution $\mathbf{S}$ from Eq. (1). The construction proceeds in three steps, successively modifying Bayesian CC. First, finite-sample BCC replaces the population target q_Y in Fig. 1(a) by the finite-sample target $\mathbf{S}$ in Fig. 1(b), represented through the count vector $\mathbf{J}$ (Section 3.1). Second, *finite-sample uncertainty-propagating ACC* (FS-UP-ACC) keeps this finite-sample target but makes the calibration posterior analytically tractable by cutting feedback

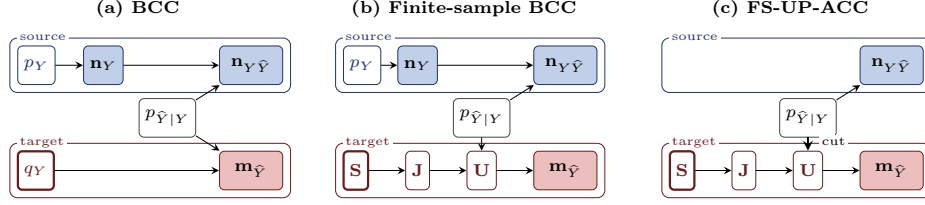


Figure 1. Dependency graphs for the Bayesian quantification models discussed in Section 3. Filled nodes denote observed count vectors: source label counts $\mathbf{n}_Y$, source joint predicted-label counts $\mathbf{n}_{Y\hat{Y}}$, and target predicted-label counts $\mathbf{m}_{\hat{Y}}$. The node $p_{\hat{Y}|Y}$ abbreviates the class-conditional predicted-label probabilities. In panels (b) and (c), $\mathbf{S}$ is represented by the finite count vector $\mathbf{J}$, which carries a prior directly. In the FS-UP-ACC model, only variables used by this cut posterior are shown. The arrow marked “cut” indicates that target counts do not update the calibration posterior for $p_{\hat{Y}|Y}$.

from the unlabeled target predictions to the calibration posterior, as shown in Fig. 1(c) (Section 3.2). Third, we describe the computationally tractable binary variant of FS-UP-ACC (Section 3.3). Figure 1 summarizes these dependency structures.

3.1 A Direct Finite-sample Bayesian Target

A direct approach to a finite-sample BCC model is to exchange the target variable. As shown in Fig. 1(a), the target label distribution q_Y in BCC induces the multinomial likelihood for the observed predicted-label counts. In finite-sample BCC (see Fig. 1(b)), this population target is replaced by the finite-sample label-distribution vector $\mathbf{S}$. Since $\mathbf{S}$ is fully determined by target-sample label counts, we work with the count vector

$$J_y = \sum_{j=1}^m \mathbb{1}[Y_j^U = y] \quad \forall y \in \mathcal{Y}, \quad (13)$$

and write $\mathbf{J} = (J_y)_{y \in \mathcal{Y}}$ for the random vector of target-sample label counts. Then $\mathbf{S} = \mathbf{J}/m$, with $\mathbf{J}$ taking values in the finite count simplex

$$\mathcal{J}_{m,C} = \left\{ \mathbf{j} \in \mathbb{N}_0^C : \sum_{y \in \mathcal{Y}} j_y = m \right\}. \quad (14)$$

A direct finite-sample Bayesian formulation therefore assigns posterior mass to the grid points $\mathbf{j}/m$, rather than a posterior density over population simplex points. The corresponding prior $\pi_{\mathbf{J}}$ is a PMF on the finite count simplex $\mathcal{J}_{m,C}$.

The source side remains unchanged. It consists of the source label likelihood from Eq. (9), the calibration likelihood from Eq. (10), and the class-conditional prediction probabilities from Eq. (6). Finite-sample BCC replaces the population

target likelihood $\ell_U(q_Y, p_{\hat{Y}|Y})$ from Eq. (11) by a likelihood conditional on the finite count vector $\mathbf{J}$. Let $\mathbf{M} = (M_{\hat{y}})_{\hat{y} \in \mathcal{Y}}$ denote the random target predicted-label count vector, with realization $(m_{\hat{y}})_{\hat{y} \in \mathcal{Y}}$ as defined in Eq. (2). For a candidate finite-sample count vector $\mathbf{J} = \mathbf{j}$, introduce latent table rows $\mathbf{U}_y = (U_{y\hat{y}})_{\hat{y} \in \mathcal{Y}}$ that allocate the j_y examples of true class y to predicted labels $\hat{y}$. A direct model contains the conditional table likelihoods

$$\begin{aligned} \mathbb{P}(\mathbf{U}_y = \mathbf{u}_y \mid J_y = j_y, p_{\hat{Y}|Y=y}) &= \text{Mult}(\mathbf{u}_y; j_y, p_{\hat{Y}|Y=y}) \quad \forall y \in \mathcal{Y}, \\ M_{\hat{y}} &= \sum_{y \in \mathcal{Y}} U_{y\hat{y}} \quad \forall \hat{y} \in \mathcal{Y}. \end{aligned} \quad (15)$$

Let $\mathcal{U}(\mathbf{j}, \mathbf{m})$ denote the set of nonnegative contingency tables with row margins $\mathbf{j}$ and column margins $\mathbf{m}$,

$$\mathcal{U}(\mathbf{j}, \mathbf{m}) = \left\{ \mathbf{u} \in \mathbb{N}_0^{C \times C} : \begin{array}{ll} \sum_{\hat{y} \in \mathcal{Y}} u_{y\hat{y}} = j_y & \forall y \in \mathcal{Y}, \\ \sum_{y \in \mathcal{Y}} u_{y\hat{y}} = m_{\hat{y}} & \forall \hat{y} \in \mathcal{Y} \end{array} \right\}. \quad (16)$$

Consequently, the target likelihood contribution for a candidate $\mathbf{j}$ is

$$\begin{aligned} \ell_{\text{FS}}(\mathbf{j}, p_{\hat{Y}|Y}) &= \mathbb{P}(\mathbf{M} = \mathbf{m} \mid \mathbf{J} = \mathbf{j}, p_{\hat{Y}|Y}) \\ &= \sum_{\mathbf{u} \in \mathcal{U}(\mathbf{j}, \mathbf{m})} \prod_{y \in \mathcal{Y}} \text{Mult}(\mathbf{u}_y; j_y, p_{\hat{Y}|Y=y}). \end{aligned} \quad (17)$$

Thus, finite-sample BCC is obtained from Eq. (12) by replacing the prior factor $\pi_q(q_Y)$ with the finite-count prior $\pi_{\mathbf{J}}(\mathbf{j})$, and by replacing the target likelihood $\ell_U(q_Y, p_{\hat{Y}|Y})$ with $\ell_{\text{FS}}(\mathbf{j}, p_{\hat{Y}|Y})$. For a candidate count vector $\mathbf{j}$, its posterior is

$$\begin{aligned} \pi_{\text{FS-BCC}}(p_Y, p_{\hat{Y}|Y}, \mathbf{J} = \mathbf{j} \mid \mathcal{D}_L, \mathcal{X}_U) \\ \propto \pi_p(p_Y) \pi_h(p_{\hat{Y}|Y}) \pi_{\mathbf{J}}(\mathbf{j}) \ell_Y(p_Y) \ell_h(p_{\hat{Y}|Y}) \ell_{\text{FS}}(\mathbf{j}, p_{\hat{Y}|Y}). \end{aligned} \quad (18)$$

This direct model makes clear why the finite-sample formulation is computationally expensive. The posterior is mixed discrete-continuous: HMC algorithms such as the NUTS sampler used for continuous Bayesian quantification [10] cannot directly update $\mathbf{J}$ as a differentiable parameter. One must either sample the finite count vector with separate discrete moves or marginalize it out. Both options require evaluating probabilities for the observed predicted-label counts, and those probabilities involve sums over latent contingency tables as in Eq. (17).

3.2 Finite-sample Uncertainty-propagating ACC

Next, we address the computational challenges of direct finite-sample BCC by introducing *finite-sample uncertainty-propagating ACC* (FS-UP-ACC), shown in Fig. 1(c). The underlying calibration assumption is the same as in ACC and BCC: for a fixed classifier h , predicted labels are generated from stable class-conditional confusion rates. That is, conditional on $Y = y$, classifier outputs are

modeled as conditionally independent draws from a categorical distribution with probability vector $p_{\hat{Y}|Y=y}$ from Eq. (6). For the labeled source data, this gives the multinomial calibration likelihood ℓ_h in Eq. (10); for the target data, the same probabilities govern the latent table rows $\mathbf{U}_y$ conditional on J_y in Eq. (15). Under PPS, the source-calibrated probabilities can therefore be used for the target sample. ACC estimates these row probabilities from labeled source data and plugs them into a correction formula. BCC and finite-sample BCC instead assign priors to these probabilities; in the fully joint finite-sample model, the unlabeled target predictions can also update them through $\ell_{\text{FS}}(\mathbf{j}, p_{\hat{Y}|Y})$.

FS-UP-ACC keeps the same multinomial confusion-rate model, but imposes the ACC-style modularization: source labels calibrate the classifier, while target predicted counts are used to infer the finite-sample count vector $\mathbf{J}$. Graphically, this is the cut edge in Fig. 1(c): target predictions depend on calibration uncertainty, but they do not update the calibration posterior. The nuisance source-label-distribution block $\pi_p(p_Y)\ell_Y(p_Y)$ from Eq. (18) is also dropped, since it is separate from the finite-sample target and calibration terms. In terms of the direct posterior in Eq. (18), the cut removes the finite-sample target likelihood $\ell_{\text{FS}}(\mathbf{j}, p_{\hat{Y}|Y})$ from the calibration update and retains only the BCC calibration block,

$$\pi_{\text{cut}}(p_{\hat{Y}|Y} \mid \mathcal{D}_L) \propto \pi_h(p_{\hat{Y}|Y}) \ell_h(p_{\hat{Y}|Y}). \quad (19)$$

Thus, relative to finite-sample BCC, FS-UP-ACC removes feedback from the unlabeled target batch into the classifier-error model while keeping the same finite-sample target chain $\mathbf{S} \rightarrow \mathbf{J} \rightarrow \mathbf{U} \rightarrow \mathbf{M}$.

To propagate calibration uncertainty under this cut, we instantiate the prior π_h over the class-conditional confusion rates. Since each row $p_{\hat{Y}|Y=y}$ parameterizes a categorical model for classifier outputs and the calibration likelihood ℓ_h is multinomial row by row, we use independent symmetric Dirichlet row priors $p_{\hat{Y}|Y=y} \sim \text{Dir}((\lambda)_{\hat{y} \in \mathcal{Y}})$ with $\lambda > 0$, where Dir denotes the Dirichlet distribution. Under the cut, these priors are conjugate to the calibration likelihood; this yields a closed-form representation of calibration uncertainty. Combining Eq. (19) with the source joint counts $n_{y\hat{y}}$ from Eq. (8) gives the row-wise posterior

$$p_{\hat{Y}|Y=y} \mid \mathcal{D}_L \sim \text{Dir}((n_{y\hat{y}} + \lambda)_{\hat{y} \in \mathcal{Y}}) \quad \forall y \in \mathcal{Y}. \quad (20)$$

For target inference, note that, conditional on $J_y = j_y$ and $p_{\hat{Y}|Y=y}$, each row $\mathbf{U}_y$ is multinomial. After integrating the Dirichlet posterior in Eq. (20), its posterior predictive distribution is Dirichlet-multinomial:

$$\mathbf{U}_y \mid J_y = j_y, \mathcal{D}_L \sim \text{DirichletMultinomial}(j_y, (n_{y\hat{y}} + \lambda)_{\hat{y} \in \mathcal{Y}}). \quad (21)$$

The observed predicted-label counts are the column sums of the latent table, so the feasible latent tables are the fixed-margin tables $\mathcal{U}(\mathbf{j}, \mathbf{m})$ from Eq. (16). Writing $\text{DM}(\cdot; n, \boldsymbol{\alpha})$ for the Dirichlet-multinomial PMF, the finite-sample predictive

likelihood is

$$\mathbb{P}(\mathbf{M} = \mathbf{m} \mid \mathbf{J} = \mathbf{j}, \mathcal{D}_L) = \sum_{\mathbf{u} \in \mathcal{U}(\mathbf{j}, \mathbf{m})} \prod_{y \in \mathcal{Y}} \text{DM}(\mathbf{u}_y; j_y, (n_{y\hat{y}} + \lambda)_{\hat{y} \in \mathcal{Y}}). \quad (22)$$

Combining this likelihood with the finite-count prior $\pi_{\mathbf{J}}$ gives

$$\mathbb{P}(\mathbf{J} = \mathbf{j} \mid \mathbf{M} = \mathbf{m}, \mathcal{D}_L) = \frac{\pi_{\mathbf{J}}(\mathbf{j}) \mathbb{P}(\mathbf{M} = \mathbf{m} \mid \mathbf{J} = \mathbf{j}, \mathcal{D}_L)}{\sum_{\mathbf{j}' \in \mathcal{J}_{m,C}} \pi_{\mathbf{J}}(\mathbf{j}') \mathbb{P}(\mathbf{M} = \mathbf{m} \mid \mathbf{J} = \mathbf{j}', \mathcal{D}_L)}. \quad (23)$$

The posterior mass assigned to the finite-sample label-distribution vector $\mathbf{j}/m$ is this same quantity.

The closed-form posterior in Eq. (23) is the FS-UP-ACC update. Its support is the finite grid of realizable target-sample label distributions. However, exact computation is expensive. The number of candidate count vectors is $\binom{m+C-1}{C-1}$, and each likelihood evaluation in Eq. (22) requires summing over constrained nonnegative integer tables. The closed-form expression is therefore useful conceptually, but not practical as a multiclass algorithm for large m or C .

3.3 A Tractable Binary Special Case

The multiclass update in Eq. (23) has a closed form, but is not yet a practical general-purpose multiclass quantifier: exact evaluation still ranges over all candidate count vectors and all compatible fixed-margin tables. For binary crisp classification, this same construction simplifies substantially. There is only one free target count, and each compatible target table is determined by one latent count. Thus, the closed-form FS-UP-ACC update from Section 3.2 becomes an efficient uncertainty-aware binary quantifier.

Let $\mathcal{Y} = \{0, 1\}$, with class 1 denoting the positive class, and write

$$J = J_1, \quad K = M_1, \quad k = m_1.$$

Here, J is the random number of positives in the target sample, $S = J/m$, K is the random number of predicted positives, and k is its realization. The finite target grid is therefore $j \in \{0, \dots, m\}$. Let $\text{tp} = n_{11}$, $\text{fn} = n_{10}$, $\text{fp} = n_{01}$, and $\text{tn} = n_{00}$ denote the realized binary source confusion counts. The row-wise Dirichlet posteriors from Eq. (20) reduce to beta posteriors for the true-positive and false-positive rates from Eq. (4):

$$a \mid \mathcal{D}_L \sim \text{Beta}(\text{tp} + \lambda, \text{fn} + \lambda), \quad b \mid \mathcal{D}_L \sim \text{Beta}(\text{fp} + \lambda, \text{tn} + \lambda). \quad (24)$$

For a candidate finite-sample count $J = j$, let $U = U_{11}$ be the number of positive target examples predicted as positive, and let $V = U_{01}$ be the number of negative target examples predicted as positive. Then $K = U + V$. Integrating the beta calibration posteriors gives the binary analogue of Eq. (21), namely the beta-binomial posterior predictive distributions

$$\begin{aligned} U \mid J = j, \mathcal{D}_L &\sim \text{BetaBinomial}(j, \text{tp} + \lambda, \text{fn} + \lambda), \\ V \mid J = j, \mathcal{D}_L &\sim \text{BetaBinomial}(m - j, \text{fp} + \lambda, \text{tn} + \lambda). \end{aligned} \quad (25)$$

Consequently, the likelihood of the observed predicted-positive count is the one-dimensional convolution

$$\mathbb{P}(K = k \mid J = j, \mathcal{D}_L) = \sum_{u=u_{\min}}^{u_{\max}} \text{BB}(u; j, \text{tp} + \lambda, \text{fn} + \lambda) \cdot \text{BB}(k - u; m - j, \text{fp} + \lambda, \text{tn} + \lambda), \quad (26)$$

where $\text{BB}(\cdot; n, \alpha, \beta)$ is the beta-binomial PMF and

$$u_{\min} = \max(0, k - (m - j)), \quad u_{\max} = \min(j, k).$$

Combining this likelihood with a prior $\pi_j = \mathbb{P}(J = j)$ on the binary finite count grid gives

$$\mathbb{P}(J = j \mid K = k, \mathcal{D}_L) = \frac{\pi_j \mathbb{P}(K = k \mid J = j, \mathcal{D}_L)}{\sum_{j'=0}^m \pi_{j'} \mathbb{P}(K = k \mid J = j', \mathcal{D}_L)}. \quad (27)$$

The posterior over the finite-sample prevalence is trivially obtained by rescaling the count grid:

$$S \in \left\{0, \frac{1}{m}, \dots, 1\right\}, \quad \mathbb{P}\left(S = \frac{j}{m} \mid K = k, \mathcal{D}_L\right) = \mathbb{P}(J = j \mid K = k, \mathcal{D}_L). \quad (28)$$

This binary specialization is cheap to compute because exact inference requires only $m + 1$ candidate values of J and, for each candidate, one finite sum over $u_{\min}, \dots, u_{\max}$. There is no continuous posterior sampling and no enumeration of high-dimensional fixed-margin tables. The multiclass formula in Eq. (23) shows the same finite-sample principle, but a practical multiclass variant would require additional algorithmic optimizations or approximations such as dynamic programming, sparse enumeration, or (discrete) Monte Carlo methods.

4 Evaluation

To evaluate the proposed FS-UP-ACC, we focus on the binary variant described in Section 3.3 and compare it against ACC and BCC. We do not compare against quantification methods that have access to richer distributional information, e.g., in the form of a probabilistic classifier or stronger shift assumptions; this includes PACC [1], as well as *distribution matching* methods, such as the *Mixture Model* approach [6], HDy [9], DyS [15], or KDEy [17]. While the additional information available to these methods can improve quantification performance in practice, our goal is not to demonstrate that the proposed FS-UP-ACC method is generally superior. Instead, we aim to show the following:

1. Finite-sample and population-level quantification are two distinct problems and require distinct approaches.
2. Given a hard classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$, FS-UP-ACC is a more appropriate method for finite-sample quantification than existing aggregative quantifiers using the same hard classifier h .

Since the multiclass methods described in Sections 3.1 and 3.2 require additional algorithmic optimizations or approximations to make them computationally feasible, we only evaluate the binary case and leave the development of such methods to subsequent research.

4.1 Synthetic Evaluation

We first isolate finite-sample quantification in a controlled count-level experiment. Instead of generating feature vectors, we simulate the output of a fixed binary classifier directly. For each requested prevalence p , we draw a random finite target count $J \sim \text{Binomial}(m, p)$ and evaluate estimates against the realized finite-sample prevalence J/m . Thus, the population prevalence p is only a data-generating parameter; it is not the evaluation target. This deliberately focuses the synthetic evaluation on the estimand addressed by FS-UP-ACC.

We model the synthetic binary classifier using a true-positive rate a and a false-positive rate b . Given a target sample of size m with j positive examples, the observed number of predicted positives is generated as

$$K = U + V, \quad \text{where } U \sim \text{Binomial}(j, a) \text{ and } V \sim \text{Binomial}(m - j, b).$$

The calibration data are generated analogously from the same classifier channel: for each class, we draw $n_{\text{cal}} = 1000$ calibration examples and record the corresponding classifier-output counts. This setup removes feature-modeling effects and directly tests whether a method is aligned with the finite-sample target. We vary the classifier TPR a and FPR b , the target sample size m , and the population sampling prevalence p . For each configuration of the resulting grid, calibration and target samples are generated and evaluated 30 times. To summarize the posterior distributions over prevalences by a single point estimate, we use the posterior mean for both methods. For uncertainty summaries, both methods use central 95% posterior credible intervals. The goal of this overall setup is to compare the differences between BCC and FS-UP-ACC.

Figure 2 summarizes the comparison. Each panel reports FS-UP-ACC minus BCC; negative values therefore favor FS-UP-ACC. The point-error panels in the top row show that the small-sample cells are consistently negative for both *absolute error* (AE) and *Kullback-Leibler divergence* (KLD), with the largest advantages at $m = 10$ and $m = 25$. This is exactly the regime in which the realized prevalence J/m can differ substantially from the population prevalence p . A population-target method such as BCC is therefore systematically mismatched to the finite-sample loss, while FS-UP-ACC places its posterior mass directly on the finite count J . As m grows, the sampling variation in J/m decreases and the differences move toward zero or become mixed. The figure therefore supports both evaluation claims: finite-sample quantification is a distinct target, and it is best addressed by a finite-sample estimator, especially for small target samples.

The panels in the bottom row of Fig. 2 compare FS-UP-ACC and BCC from the perspective of coverage and interval size. In the small-sample regime, FS-UP-ACC has smaller absolute coverage gaps than BCC, meaning that its nominal

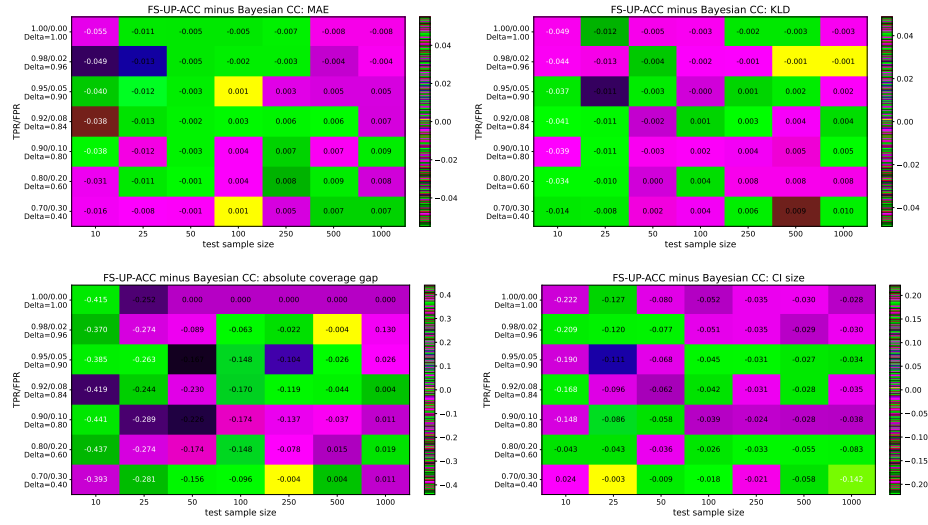


Figure 2. Synthetic comparison on the random finite-sample target. Each cell shows FS-UP-ACC minus BCC; negative values favor FS-UP-ACC. Top row: mean AE and mean KLD. Bottom row: absolute coverage gap from nominal 95% central credible-interval coverage and average central credible-interval size. Rows within each panel vary the classifier TPR and FPR.

95% intervals are closer to their intended coverage. At the same time, the intervals are almost always shorter. The uncertainty results therefore reinforce the point-estimate results: when the target is the realized finite-sample prevalence, modeling the finite count is most valuable precisely when the target sample is small.

4.2 Real-data Evaluation

We next ask whether the same small-sample pattern appears with real classifiers. The benchmark contains 29 binary datasets from the QuaPy UCI collection [16]. For the UCI datasets, we use a train/validation/test-pool split with proportions 0.50/0.25/0.25. The classifier is a logistic-regression pipeline with median imputation and standardization for numeric features, and most-frequent imputation plus one-hot encoding for categorical features. The validation split is used to estimate the class-conditional classifier-output probabilities required by ACC, BCC, and FS-UP-ACC.

Target samples are generated using the *artificial prevalence protocol* (APP): for requested prevalences 0.00, 0.04, $\dots$, 1.00, we draw small target samples of size $m = 25$ from the test pool, with 25 repeats per prevalence. The experiment is repeated for three split seeds; in all cases the evaluation target is the realized sample prevalence. We report *absolute error* (AE) and *Kullback-Leibler divergence* (KLD), as well as 95% central posterior credible-interval width and

Table 1. Per-dataset small-sample real-data point-estimate performance. Entries are means $\pm$ standard errors over repeated APP runs. Lower values are better; bold values mark the minimum-mean method and methods whose mean is not significantly higher than that minimum with one-sided 95% normal-approximation confidence. The final row reports the number of datasets won or tied by each method under the same criterion. FUA denotes FS-UP-ACC.

Dataset	AE				KLD			
	CC	ACC	BCC	FUA	CC	ACC	BCC	FUA
balance.1	0.039 $\pm$ 0.001	0.035 $\pm$ 0.001	0.039 $\pm$ 0.001	0.035 $\pm$ 0.001	0.014 $\pm$ 0.001	0.042 $\pm$ 0.005	0.011 $\pm$ 0.000	0.008 $\pm$ 0.000
balance.3	0.041 $\pm$ 0.001	0.036 $\pm$ 0.001	0.043 $\pm$ 0.001	0.037 $\pm$ 0.001	0.014 $\pm$ 0.001	0.020 $\pm$ 0.003	0.012 $\pm$ 0.000	0.009 $\pm$ 0.000
breast-cancer	0.020 $\pm$ 0.001	0.025 $\pm$ 0.000	0.024 $\pm$ 0.000	0.025 $\pm$ 0.000	0.006 $\pm$ 0.001	0.029 $\pm$ 0.004	0.007 $\pm$ 0.000	0.005 $\pm$ 0.000
cmc.1	0.223 $\pm$ 0.004	0.215 $\pm$ 0.004	0.176 $\pm$ 0.003	0.173 $\pm$ 0.003	0.182 $\pm$ 0.005	2.066 $\pm$ 0.081	0.121 $\pm$ 0.003	0.119 $\pm$ 0.003
cmc.2	0.418 $\pm$ 0.006	0.317 $\pm$ 0.006	0.224 $\pm$ 0.003	0.223 $\pm$ 0.003	1.481 $\pm$ 0.059	5.482 $\pm$ 0.163	0.178 $\pm$ 0.005	0.178 $\pm$ 0.005
cmc.3	0.347 $\pm$ 0.006	0.336 $\pm$ 0.006	0.235 $\pm$ 0.003	0.235 $\pm$ 0.003	0.538 $\pm$ 0.025	5.918 $\pm$ 0.166	0.188 $\pm$ 0.005	0.189 $\pm$ 0.005
ctg.1	0.115 $\pm$ 0.002	0.072 $\pm$ 0.001	0.072 $\pm$ 0.001	0.069 $\pm$ 0.001	0.064 $\pm$ 0.002	0.092 $\pm$ 0.008	0.029 $\pm$ 0.001	0.027 $\pm$ 0.001
ctg.2	0.207 $\pm$ 0.003	0.101 $\pm$ 0.002	0.104 $\pm$ 0.002	0.100 $\pm$ 0.002	0.168 $\pm$ 0.005	0.231 $\pm$ 0.018	0.051 $\pm$ 0.002	0.048 $\pm$ 0.001
ctg.3	0.157 $\pm$ 0.003	0.081 $\pm$ 0.002	0.080 $\pm$ 0.001	0.075 $\pm$ 0.001	0.118 $\pm$ 0.005	0.058 $\pm$ 0.005	0.035 $\pm$ 0.001	0.032 $\pm$ 0.001
german	0.243 $\pm$ 0.004	0.168 $\pm$ 0.003	0.152 $\pm$ 0.002	0.149 $\pm$ 0.002	0.222 $\pm$ 0.006	1.285 $\pm$ 0.062	0.095 $\pm$ 0.003	0.093 $\pm$ 0.003
haberman	0.392 $\pm$ 0.006	0.261 $\pm$ 0.005	0.236 $\pm$ 0.003	0.235 $\pm$ 0.003	0.863 $\pm$ 0.036	3.867 $\pm$ 0.127	0.185 $\pm$ 0.004	0.184 $\pm$ 0.004
ionosphere	0.167 $\pm$ 0.003	0.110 $\pm$ 0.002	0.109 $\pm$ 0.002	0.105 $\pm$ 0.002	0.109 $\pm$ 0.003	0.069 $\pm$ 0.004	0.054 $\pm$ 0.002	0.051 $\pm$ 0.001
iris.2	0.214 $\pm$ 0.004	0.195 $\pm$ 0.004	0.181 $\pm$ 0.003	0.177 $\pm$ 0.003	0.177 $\pm$ 0.005	0.935 $\pm$ 0.048	0.129 $\pm$ 0.004	0.126 $\pm$ 0.004
iris.3	0.049 $\pm$ 0.001	0.044 $\pm$ 0.001	0.053 $\pm$ 0.001	0.047 $\pm$ 0.001	0.021 $\pm$ 0.002	0.027 $\pm$ 0.003	0.017 $\pm$ 0.000	0.014 $\pm$ 0.000
mammographic	0.089 $\pm$ 0.001	0.097 $\pm$ 0.002	0.090 $\pm$ 0.001	0.089 $\pm$ 0.002	0.041 $\pm$ 0.001	0.442 $\pm$ 0.028	0.039 $\pm$ 0.001	0.040 $\pm$ 0.001
pageblocks.5	0.380 $\pm$ 0.005	0.197 $\pm$ 0.004	0.173 $\pm$ 0.003	0.172 $\pm$ 0.003	0.869 $\pm$ 0.028	0.515 $\pm$ 0.030	0.121 $\pm$ 0.003	0.123 $\pm$ 0.004
semeion	0.092 $\pm$ 0.002	0.050 $\pm$ 0.001	0.059 $\pm$ 0.001	0.054 $\pm$ 0.001	0.045 $\pm$ 0.002	0.114 $\pm$ 0.012	0.020 $\pm$ 0.001	0.016 $\pm$ 0.001
sonar	0.129 $\pm$ 0.002	0.110 $\pm$ 0.002	0.119 $\pm$ 0.002	0.116 $\pm$ 0.002	0.070 $\pm$ 0.002	0.409 $\pm$ 0.028	0.059 $\pm$ 0.001	0.057 $\pm$ 0.001
spambase	0.061 $\pm$ 0.001	0.049 $\pm$ 0.001	0.051 $\pm$ 0.001	0.048 $\pm$ 0.001	0.024 $\pm$ 0.001	0.062 $\pm$ 0.006	0.017 $\pm$ 0.001	0.015 $\pm$ 0.001
spectf	0.291 $\pm$ 0.005	0.255 $\pm$ 0.005	0.213 $\pm$ 0.003	0.210 $\pm$ 0.003	0.341 $\pm$ 0.013	3.543 $\pm$ 0.134	0.162 $\pm$ 0.004	0.160 $\pm$ 0.004
tictactoe	0.052 $\pm$ 0.001	0.035 $\pm$ 0.001	0.041 $\pm$ 0.001	0.035 $\pm$ 0.001	0.020 $\pm$ 0.001	0.027 $\pm$ 0.004	0.012 $\pm$ 0.001	0.009 $\pm$ 0.000
transfusion	0.435 $\pm$ 0.006	0.296 $\pm$ 0.005	0.223 $\pm$ 0.003	0.222 $\pm$ 0.003	2.093 $\pm$ 0.085	5.327 $\pm$ 0.150	0.169 $\pm$ 0.004	0.168 $\pm$ 0.004
wdbc	0.033 $\pm$ 0.001	0.040 $\pm$ 0.001	0.042 $\pm$ 0.001	0.040 $\pm$ 0.001	0.012 $\pm$ 0.001	0.018 $\pm$ 0.002	0.012 $\pm$ 0.000	0.010 $\pm$ 0.000
wine-q-red	0.152 $\pm$ 0.002	0.129 $\pm$ 0.002	0.126 $\pm$ 0.002	0.123 $\pm$ 0.002	0.092 $\pm$ 0.002	0.412 $\pm$ 0.028	0.071 $\pm$ 0.002	0.069 $\pm$ 0.002
wine-q-white	0.226 $\pm$ 0.004	0.143 $\pm$ 0.003	0.134 $\pm$ 0.002	0.131 $\pm$ 0.002	0.187 $\pm$ 0.005	0.705 $\pm$ 0.039	0.079 $\pm$ 0.002	0.077 $\pm$ 0.002
wine.1	0.000 $\pm$ 0.000	0.022 $\pm$ 0.001	0.020 $\pm$ 0.000	0.021 $\pm$ 0.000	0.000 $\pm$ 0.000	0.029 $\pm$ 0.003	0.005 $\pm$ 0.000	0.003 $\pm$ 0.000
wine.2	0.010 $\pm$ 0.001	0.022 $\pm$ 0.001	0.023 $\pm$ 0.001	0.024 $\pm$ 0.001	0.004 $\pm$ 0.001	0.007 $\pm$ 0.001	0.006 $\pm$ 0.000	0.004 $\pm$ 0.000
wine.3	0.019 $\pm$ 0.001	0.031 $\pm$ 0.001	0.040 $\pm$ 0.001	0.036 $\pm$ 0.001	0.008 $\pm$ 0.001	0.059 $\pm$ 0.006	0.011 $\pm$ 0.000	0.008 $\pm$ 0.000
yeast	0.309 $\pm$ 0.005	0.184 $\pm$ 0.004	0.162 $\pm$ 0.003	0.159 $\pm$ 0.003	0.392 $\pm$ 0.015	1.124 $\pm$ 0.056	0.107 $\pm$ 0.003	0.104 $\pm$ 0.003
# wins or ties	6/29	9/29	13/29	21/29	4/29	0/29	15/29	28/29

coverage for the Bayesian methods. For both BCC and FS-UP-ACC, we use the posterior mean as the point estimate; for FS-UP-ACC, this mean is taken over the finite-sample posterior. The tables below first average over samples and split seeds within each dataset; the aggregate row shows the number of times each method performs best or is tied with the best-performing method.

Table 1 shows the per-dataset point-estimate results. The wins-or-ties row clearly shows that FS-UP-ACC is competitive on substantially more datasets than BCC: 21/29 versus 13/29 under AE, and 28/29 versus 15/29 under KLD. This mirrors the synthetic result: when target samples are small and the evaluation target is the realized sample prevalence, the finite-sample posterior is the better match. The individual rows also show that the advantage is not uniform, while FS-UP-ACC does not dominate every task, it is the most reliable hard-classifier method in the finite-sample regime.

The results in Table 2 show the same uncertainty pattern as the synthetic plots. Compared with BCC, FS-UP-ACC produces tighter intervals on average while also moving coverage closer to the nominal 95% level. Thus, the real-data evidence supports the same conclusion as the synthetic study: finite-sample quantification is best handled by a method whose posterior target is the sample prevalence, and the advantage is most visible when the target sample is small.

Table 2. Small-sample real-data central posterior credible-interval performance. Entries are equal-dataset means $\pm$ standard errors. Coverage is empirical coverage of nominal 95% central credible intervals.

Method	Credible interval width	Coverage
BCC	0.575 ± 0.041	0.912 ± 0.003
FS-UP-ACC	0.509 ± 0.055	0.980 ± 0.003

5 Conclusion

We argue that quantification has two related but distinct targets: the population label distribution and the label distribution realized in a particular finite target sample. The latter is the relevant estimand in many applications and in common finite-sample evaluation protocols, but it is not the posterior target of standard population-level methods such as ACC or BCC. To address this mismatch, we introduced finite-sample quantification and developed FS-UP-ACC, a Bayesian hard-classifier quantifier that places posterior mass directly on feasible finite target counts while propagating calibration uncertainty. We found that, across controlled synthetic experiments and small-sample real-data benchmarks, binary FS-UP-ACC generally improves over BCC when evaluated against the realized finite-sample target. This work focuses on the special case of hard binary classifier outputs and is based on the commonly made PPS assumption. For future work, we envision a more general framework that extends FS-UP-ACC efficiently to the multiclass setting, to probabilistic classifier outputs, and other types of distribution shift (e.g., covariate [21], structural [3], or concept shift).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bella, A., Ferri, C., Hernández-Orallo, J., Ramírez-Quintana, M.J.: Quantification via Probability Estimators. In: IEEE International Conference on Data Mining, pp. 737–742 (2010), ISSN 2374-8486, <https://doi.org/10.1109/ICDM.2010.75>
2. Bunse, M.: On multi-class extensions of adjusted classify and count. In: The 2nd International Workshop on Learning to Quantify (LQ 2022), pp. 43–50 (2022)
3. Damke, C., Hüllermeier, E.: Adjusted Count Quantification Learning on Graphs. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)
4. Esuli, A., Fabris, A., Moreo, A., Sebastiani, F.: Learning to Quantify, The Information Retrieval Series, vol. 1. Springer, Cham (2023), ISBN 978-3-031-20467-8
5. Firat, A.: Unified Framework for Quantification (2016), <https://doi.org/10.48550/arXiv.1606.00868>
6. Forman, G.: Counting positives accurately despite inaccurate classification. In: ECML, pp. 564–575, ECML’05, Springer-Verlag, Berlin, Heidelberg (2005), ISBN 978-3-540-29243-2, https://doi.org/10.1007/11564096_55

7. Forman, G.: Quantifying trends accurately despite classifier error and class imbalance. In: KDD, pp. 157–166, KDD '06, Association for Computing Machinery, New York, NY, USA (2006), ISBN 978-1-59593-339-3
8. Forman, G.: Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery* **17**(2), 164–206 (2008), ISSN 1573-756X, <https://doi.org/10.1007/s10618-008-0097-y>
9. González-Castro, V., Alaiz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the Hellinger distance. *Information Sciences* **218**, 146–164 (2013), ISSN 0020-0255, <https://doi.org/10.1016/j.ins.2012.05.028>
10. Hoffman, M.D., Gelman, A.: The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.* **15**(47), 1593–1623 (2014), ISSN 1533-7928
11. Hopkins, D.J., King, G.: A Method of Automated Nonparametric Content Analysis for Social Science. *American Journal of Political Science* **54**(1), 229–247 (2010), ISSN 1540-5907, <https://doi.org/10.1111/j.1540-5907.2009.00428.x>
12. Igraneza, A.B., Fraser, C., Hinch, R.: Estimating prevalence with precision and accuracy (2025), <https://doi.org/10.48550/arXiv.2507.06061>
13. Keith, K., O'Connor, B.: Uncertainty-aware generative models for inferring document class prevalence. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4575–4585 (2018)
14. Lipton, Z., Wang, Y.X., Smola, A.: Detecting and Correcting for Label Shift with Black Box Predictors. In: *ICML*, pp. 3122–3130, PMLR (2018), ISSN 2640-3498
15. Maletzke, A., dos Reis, D., Cherman, E., Batista, G.: DyS: A Framework for Mixture Models in Quantification. *Proceedings of the AAAI Conference on Artificial Intelligence* **33**(01), 4552–4560 (2019), ISSN 2374-3468
16. Moreo, A., Esuli, A., Sebastiani, F.: QuaPy: A Python-Based Framework for Quantification. In: *CIKM*, pp. 4534–4543, *CIKM '21*, Association for Computing Machinery, New York, NY, USA (2021), ISBN 978-1-4503-8446-9, <https://doi.org/10.1145/3459637.3482015>
17. Moreo, A., González, P., del Coz, J.J.: Kernel density estimation for multiclass quantification. *Machine Learning* **114**(4), 1–38 (2025), ISSN 1573-0565, <https://doi.org/10.1007/s10994-024-06726-5>
18. Saerens, M., Latinne, P., Decaestecker, C.: Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation* **14**(1), 21–41 (2002), ISSN 0899-7667, <https://doi.org/10.1162/089976602753284446>
19. Tasche, D.: Fisher consistency for prior probability shift. *J. Mach. Learn. Res.* **18**(1), 3338–3369 (2017), ISSN 1532-4435
20. Tasche, D.: Confidence Intervals for Class Prevalences under Prior Probability Shift. *Machine Learning and Knowledge Extraction* **1**(3), 805–831 (2019), ISSN 2504-4990, <https://doi.org/10.3390/make1030047>
21. Tasche, D.: Class Prior Estimation under Covariate Shift: No Problem? (2022), <https://doi.org/10.48550/arXiv.2206.02449>
22. Vaz, A., Izbicki, R., Stern, R.B.: Prior Shift Using the Ratio Estimator. In: Polpo, A., Stern, J., Louzada, F., Izbicki, R., Takada, H. (eds.) *Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, pp. 25–35, Springer International Publishing, Cham (2018), ISBN 978-3-319-91143-4
23. Vucetic, S., Obradovic, Z.: Classification on Data with Biased Class Distribution. In: De Raedt, L., Flach, P. (eds.) *ECML*, pp. 527–538, Springer, Berlin, Heidelberg (2001), ISBN 978-3-540-44795-5, https://doi.org/10.1007/3-540-44795-4_45
24. Ziegler, A., Czyż, P.: Bayesian Quantification with Black-Box Estimators. *Transactions on Machine Learning Research* (2024), ISSN 2835-8856

Decision Tree Leaf Binning with Expectation Maximization for Quantification

David Kaftan^{1,2}, George Corliss¹, Anna Bershteyn², and Richard Povinelli¹

¹ Marquette University, Milwaukee WI, USA

{david.kaftan, richard.povinelli}@marquette.edu

² NYU Grossman School of Medicine, New York NY, USA

Abstract. Recent large-scale evaluations have reported poor empirical performance of decision trees for quantification, the task of estimating class prevalence in an unlabelled bag of data. We hypothesize that this failure stems from the choice of aggregation method rather than from the tree structure itself. Drawing on unfolding work in experimental physics, we propose Leaf EM, a quantification method that uses decision-tree leaves as a discrete binning of the feature space and applies Iterative Bayesian Unfolding to estimate class prevalence directly from leaf-count distributions, without requiring assumptions about classifier calibration. We evaluate Leaf EM against seven aggregative quantifiers across three tree splitting criteria (including Gini and quantification criteria Classification Error Balancing and Classification-Quantification Balancing) on 40 binary and 17 multiclass datasets. On binary tasks, the Saerens-Latinne-Decaestecker algorithm applied to a Classification-Quantification Balancing tree achieved the best overall mean rank (4.47 of 21 tree-quantifier combinations), while Leaf EM attained the best within-tree mean rank on Gini trees (2.35 out of 7 quantifiers) and was significantly better than Adjusted Classify and Count, Probabilistic Adjusted Classify and Count, and the Saerens-Latinne-Decaestecker algorithm with temperature scaling on the same tree. Adjusted Classify and Count, previously evaluated on decision trees, performed among the worst quantifiers across tree types. These results indicate that with an appropriate aggregator, decision trees can be greatly improved for quantification.

Keywords: Quantification · Decision Trees · Expectation-Maximization · Maximum Likelihood Estimation · Prior Probability Shift · Prevalence Estimation

1 Introduction

Quantification is the supervised learning task of estimating the prevalence of each class within an unlabelled data sample, rather than predicting the class of individual instances. Quantification arises wherever population-level estimates matter more than individual predictions: tracking political sentiment on social

media [24], estimating disease prevalence from imperfect screening tests [25], monitoring e-commerce catalog defect rates [32], estimating species distributions in ecology [20], and assessing classifier fairness across demographic subpopulations [16].

Quantification typically occurs in two stages. First, a base classifier reports scores for an unlabelled bag. Second, these scores are aggregated into estimates of prevalence for each class. Most aggregative methods can, in principle, be applied to any base classifier. However, the characteristics of the base classifier can influence which aggregative quantifier is optimal. For example, the expectation maximization algorithm proposed by Saerens et al. is highly sensitive to the calibration of the base classifier [13]. It is therefore important to explore aggregative methods that perform well for specific base classifiers.

Classification trees are a simple but powerful classifier. Their ability to model non-linear relationships and their interpretability make classifier trees a relevant machine learning technique decades after they were first popularized. Unfortunately, aggregative quantification methods have shown poor empirical performance when applied to decision trees [26, 34]. Additionally, decision trees are commonly miscalibrated [37], which leads to poor performance of certain aggregative quantifiers [13]. Despite this, empirical evaluations of quantification methods on decision trees have been limited.

Our work builds on two prior lines of research. First, Börner et al. showed that decision-tree leaves can provide an informative discretization of the feature space for unfolding problems [4]. Second, Bunse showed that unfolding and quantification are closely related problems, and that Iterative Bayesian Unfolding (IBU), an expectation maximization algorithm, can be interpreted within a quantification framework [7]. We propose Leaf EM, a quantification method for decision trees that uses tree leaves as discrete bins and infers class prevalence directly from leaf-count distributions. We also provide a large-scale empirical evaluation showing that the weak quantification performance previously reported for decision trees is attributable in part to the aggregation method, and that with a more appropriate aggregator, decision trees can be substantially improved for quantification.

2 Background

2.1 Quantification under Prior Probability Shift

Quantification, or class prevalence estimation, is the task of estimating the distribution of class labels in an unlabelled data sample [12]. Given a feature space $\mathcal{X}$ and a label set $Y \in \mathcal{Y} = \{Y_1, \dots, Y_{n_Y}\}$, the goal is to estimate the prevalence vector $\mathbf{p} = (p_1, \dots, p_{n_Y})$ where $p_i = P(Y = Y_i)$ for an unlabelled test sample. When quantification is performed as a post-process to calibration, the feature space $\mathcal{X}$ typically represents the output of the classifier. The training data consists of labeled instances drawn from a distribution $P_{train}(X, Y)$, while the test data is drawn from a different distribution $P_{test}(X, Y)$. Under

the assumption of prior probability shift, we have $P_{train}(X|Y) = P_{test}(X|Y)$ but $P_{train}(Y) \neq P_{test}(Y)$ [12]. This means that while the relationship between features and labels remains stable, the overall class prevalences differ between training and test sets. This also implies that, under prior probability shift, simply counting the predicted labels from a classifier will yield biased prevalence estimates on the test bag. There are several methods for quantification that exploit the assumption of prior probability shift, including the method proposed in this manuscript.

One such method optimizes $\mathbf{p}$ to maximize the likelihood of observing an unlabelled bag of data (Eq. 1).

$$L(X) = \prod_{i=1}^N \sum_{j=1}^{n_Y} P(X_i|Y_j) \mathbf{p}_j \quad (1)$$

Saerens, Latinne, and Decaestecker developed an EM algorithm (SLD), which iteratively updates prevalence estimates using Bayes' theorem, converging to the maximum likelihood estimate of prevalence [33]. SLD avoids the difficult task of estimating $P(X|Y)$ by instead relying on the posterior probabilities $P(Y|X)$ reported by a base classifier, using the substitution in Eq. 2. This makes SLD highly sensitive to classifier calibration [13]. Despite this, SLD often ranks among the state of the art for quantification tasks [14, 15, 34], and has been shown to be Fisher consistent [36].

$$P(X_i | Y_j) = \frac{P^{(0)}(Y_j | X_i) P^{(0)}(X_i)}{P^{(0)}(Y_j)} \quad (2)$$

In addition to SLD, several other methods have been proposed and evaluated for quantification under prior probability shift. These methods have been broadly grouped into three categories: classify, count, and correct; machine learners adapted for quantification; and distribution matching methods [19]. Many of these methods can be applied as a post-process to decision trees, though evaluation and direct adaptation to decision trees have received relatively little attention. In the next sections, we review the few methods that have been directly proposed for quantification with decision trees, as well as the unfolding methods from physics that we adapt in this work.

2.2 Decision Trees for Quantification

Decision trees are one such machine learner adapted for quantification. Milli et al. (2013) propose quantification trees, a type of decision tree optimized not for individual classification accuracy, but directly for the quantification task of estimating class prevalences [26]. Milli proposes two splitting criteria that provide the best balance between false positives and false negatives across classes. These two criteria are Classification Error Balancing (CEB), where for each class i the quantification error is $E_i = |FP_i - FN_i|$, and Classification-Quantification Balancing (CQB), where $E_i = |FP_i^2 - FN_i^2|$, which additionally penalizes total misclassification. These per-class errors form a single vector QE_r , and the

goodness of a split is measured by the gain $\Delta = \|QE_r^{parent}\|_2 - \|QE_r^{child}\|_2$. However, in a recent large-scale evaluation by Schumacher et al. (2025), both quantification forests and their ACC-corrected variant performed poorly [34].

2.3 Unfolding Methods for Quantification

Unfolding is a family of techniques developed in experimental physics for the problem of recovering a “true” distribution of a physical quantity from observations that have been distorted by detector resolution, measurement noise, and limited efficiency [10]. Such distortions are ubiquitous in high-energy and astroparticle physics, where the quantity of interest can only be inferred indirectly through correlated, measurable observables [7]. The problem is typically discretized by binning the underlying quantity into a set of cause bins $\{C_i\}_{i=1}^{n_C}$ and the observable into a set of effect bins $\{E_j\}_{j=1}^{n_E}$, so that the measurement process is characterized by the conditional probabilities $P(E_j | C_i)$ of observing effect E_j given cause C_i . The goal of unfolding is to recover the distribution over causes from the observed distribution over effects, for which the field has developed several approaches, including Iterative Bayesian Unfolding (IBU) [9, 10], described in the forthcoming Methods section; Singular Value Decomposition-based unfolding [23]; and Blobel’s Regularized Unfolding [2, 3]. At its core, unfolding addresses the same inferential problem as quantification: recovering a distribution over classes (or bins) from imperfect observations [7].

There are several methods for defining the bins E_j . One such method is to use the leaf nodes of a decision tree as bins, as proposed by Börner et al. (2020) [4]. In this approach, a decision tree is trained on the data, and each leaf node corresponds to a bin. Börner did not perform unfolding, but demonstrated that decision tree leaf binning can partition the feature space into informative effect bins. Combining this approach for estimating $P(E_j | C_i)$, and Bunse’s unification of IBU into a quantification framework [7], we propose a method for quantification with decision trees that uses the tree leaves as bins and IBU as the solver.

3 Methods

3.1 Overview: Tree-Leaf Binning + EM Quantification

We adapt IBU to the context of decision tree-based quantification. The method consists of two stages: first, a decision tree is trained on the labeled training data, and the leaf nodes are used to both classify instances and define a discrete partition of the feature space. The class conditional distribution $P(E | C)$ characterizes how instances from each true cause are distributed across the effects. $P(E | C)$ in unfolding can be translated to $P(X | Y)$ in quantification. Second, given an unlabelled test sample, we count the number of instances that fall into each leaf node, yielding an observed distribution $\mathbf{n}$. We then apply IBU to solve for the prevalence vector $\mathbf{p}$ that best explains the observed leaf counts $\mathbf{n}$ under the class conditional distribution.

3.2 Tree Construction

Trees are constructed using standard classification tree algorithms (e.g., CART [5]) with various splitting criteria (Gini, CEB, CQB). Trees using Gini Impurity were implemented using scikit-learn v1.8.0 [30], while trees constructed using CEB and CQB splitting criteria were reimplemented in Python according to Milli et al. [26]. The tree is trained on a subset of the labeled training data, and the remaining labeled data is used as a held-out estimation set to estimate the class conditional distributions. Details of split proportions are given in Section 4. Trees are grown without a maximum depth limit, or a minimum number of samples per leaf constraint during fitting. Instead, the tree is pruned post-hoc using a minimum of 5 samples per leaf from the held-out estimation set to both reduce overfitting and ensure a stable estimate of the class conditional distributions.

3.3 EM Solver for Prevalence Estimation

Iterative Bayesian Unfolding (IBU), originally proposed by D’Agostini [9], is an expectation-maximization approach to solving the unfolding problem. Starting from a prior estimate $\mathbf{p}^{(0)}$ (e.g., a uniform distribution $\mathbf{p}^{(0)} = \frac{1}{n_Y}$), IBU iteratively updates the prevalence estimate according to Bayes’ theorem, converging to the maximum likelihood estimate of prevalence under the observed leaf counts $\mathbf{n}$ and the class conditional distribution $P(\text{Leaf} | Y)$ (see Algorithm 1). The IBU algorithm, as applied to quantification by Bunse et al., is nearly identical to the SLD algorithm. The only difference is that SLD estimates the class conditional probabilities $P(X|Y)$ using the posterior probabilities $P(Y|X)$ output by a base classifier according to Eq. 2, while IBU assumes that the class conditional probabilities are known, typically through a partitioning of X . It is for this reason that SLD is highly sensitive to classifier calibration. Decision trees are known to produce poorly calibrated probabilities [31, 35], motivating the use of IBU in this context.

Decision trees map input data into leaves. We therefore redefine class conditional distributions $P(X | Y) = P(\text{Leaf} | Y)$ [4]. We choose to model $P(\text{Leaf} | Y_i)$ as a Dirichlet distribution with Laplace smoothing and updated by counting the number of instances from each true class that fall into each leaf node on the held-out estimation set. We use the expected value as a well-defined, regularized set of class conditional distributions that can be used in the IBU algorithm without requiring any assumptions about classifier calibration. Because trees are pruned based on the held-out estimation data, we avoid using leaves that provide little information.

4 Experimental methodology

To evaluate the performance of our proposed quantification approach, we follow the experimental protocol of Schumacher et al. [34]. In total, we evaluate 40 binary datasets and 17 multiclass datasets across the remaining combinations of train-test splits, training class prevalences, and test bag prevalences. Across

Algorithm 1 Iterative Bayesian Unfolding (IBU) adapted for Leaf EM

Require: $n(Leaf_j)$, $P(Leaf_j | Y_i)$
Ensure: Estimated prevalence vector $\hat{\mathbf{p}} = (\hat{p}_1, \dots, \hat{p}_{n_Y})$

```

1:  $k \leftarrow 0$ 
2:  $P^{(0)}(Y_i) \leftarrow \frac{1}{n_Y}$  for  $i = 1, \dots, n_Y$ 
3: repeat
4:   for  $j = 1$  to  $n_{Leaf}$  do
5:     for  $i = 1$  to  $n_Y$  do
6:       
$$P^{(k)}(Y_i | Leaf_j) \leftarrow \frac{P(Leaf_j | Y_i) P^{(k)}(Y_i)}{\sum_{i'=1}^{n_Y} P(Leaf_j | Y_{i'}) P^{(k)}(Y_{i'})}$$

7:     end for
8:   end for
9:   for  $i = 1$  to  $n_Y$  do
10:    
$$\hat{n}^{(k+1)}(Y_i) \leftarrow \sum_{j=1}^{n_{Leaf}} n(Leaf_j) P^{(k)}(Y_i | Leaf_j)$$

11:    
$$P^{(k+1)}(Y_i) \leftarrow \frac{\hat{n}^{(k+1)}(Y_i)}{\sum_{i'=1}^{n_Y} \hat{n}^{(k+1)}(Y_{i'})}$$

12:   end for
13:    $k \leftarrow k + 1$ 
14: until convergence
15:  $\hat{p}_i \leftarrow P^{(k)}(Y_i)$  for  $i = 1, \dots, n_Y$ 
16: return  $\hat{\mathbf{p}} = (\hat{p}_1, \dots, \hat{p}_{n_Y})$ 

```

all variables, we run 12,540 experiments, each repeated ten times with different random seeds, for a total of 125,400 experiments.

Deviating from the original setup, we do not employ stratified 10-fold cross-validation when estimating the misclassification rates, classifier scores, or other quantities typically determined through cross-validation. Instead, we estimate these quantities from a single held-out estimation set (25% of training data). This demonstrates the performance of a single, interpretable tree, rather than an ensemble. While ensemble classifiers have been shown to improve performance [26, 29, 34], we consider ensemble methods to be orthogonal to the contributions of this work, and leave the question of how best to ensemble our proposed method to future work.

Details on training and testing splits, class prevalences, datasets, and seeding can be found in [34]. Quantification methods and evaluation metrics are described in the following.

4.1 Quantification Methods

We compare the following quantification methods.

Leaf EM: Our proposed method, which uses the leaf nodes of a decision tree as bins, and applies Iterative Bayesian Unfolding to estimate class prevalences. The held-out estimation set is used to estimate $P(Leaf | Y)$.

ACC: Adjusted Classify and Count, which corrects the prevalence estimates of a base classifier by adjusting for its true and false positive rates [17, 18]. ACC

is used in this experiment as it is the quantification technique applied in the original quantification tree paper by Milli et al. (2013) [26], and in Schumacher et al. (2025) [34]. The multiclass extension of ACC is implemented according to the unconstrained least squares method proposed by Bunse [6] to minimize the loss function proposed by Hopkins and King [24]. The held-out estimation set is used to estimate the true and false positive rates for each class.

PACC: Probabilistic Adjusted Classify and Count, which extends ACC to probabilistic classifiers by using the predicted probabilities instead of hard classifications [1]. The multiclass version of this method is implemented using the same methods as ACC. The held-out estimation set is used to estimate the average predicted probability for each class.

KDEyML: Kernel Density Estimation with Maximum Likelihood Estimation, which estimates the class conditional distributions of classifier scores using kernel density estimation, and then applies maximum likelihood estimation to recover the prevalence vector [28]. While this method is designed for continuous classifier scores, we apply it to the discrete scores output by decision trees because it bears a similarity to the unfolding approach we propose. Specifically, as the bandwidth of the kernel approaches zero, KDEyML approaches the same likelihood function as our proposed method (as long as no leaves produce identical classifier scores). By using KDEyML, we can compare the performance of our proposed method to a similar method that does not take advantage of the discrete nature of decision tree outputs. For this experiment, a bandwidth of 0.05 is used. The held-out estimation set is used to fit the kernel density estimates for each class.

SLD: The EM algorithm described in Section 2.1. Because calibration is important for this method, it is evaluated with and without temperature scaling, denoted SLD (TS). The held-out estimation set is used to perform temperature scaling [22].

HDy: The Hellinger Distance method proposed by Gonzalez-Castro et al. [21], which estimates class prevalences by minimizing the Hellinger distance between the predicted and true distributions of classifier scores. It performs well as a binary quantification method, and is included as a baseline for comparison [8, 34]. It was not implemented for multiclass quantification as it is not inherently designed for multiclass problems and has been shown to perform poorly [34].

For each of these methods, we evaluated three base classifiers: a standard decision tree using the Gini impurity criterion, a quantification tree using splitting criteria designed for quantification: the CEB criterion and the CQB criterion [26]. All comparative methods were implemented using the QuaPy library in Python 3.13 [27].

4.2 Evaluation Measures

Following [34], we evaluate quantification performance using Absolute Error (AE), summed across all classes (Eq. 3). This metric is used, rather than mean absolute error across classes, as it allows for consistent comparisons across datasets with different class cardinality, as described in [34].

$$e_{\text{AE}}(p, \hat{p}) = \sum_{i=1}^L |p_i - \hat{p}_i|. \quad (3)$$

Most methods run by Schumacher et al. [34] used logistic regression base classifiers, which are able to provide informative confidence scores when trained on few data. They evaluated several settings where the base classifier was trained on little data with high class imbalance. Under these settings, the CEB and CQB splitting criteria frequently reached their stopping conditions before making a single split, yielding degenerate single-leaf trees. Additionally, the pruning policy of requiring a minimum number of samples in each leaf also contributed to the formation of degenerate trees. While degenerate trees are clearly undesirable, their relative impact on performance across quantification methods is arbitrary as the trees will provide no information about the underlying class distribution. For this reason, we exclude all trials where at least one tree was degenerate from calculation of absolute error. We do, however, report the percent of trials by tree type that were degenerate to evaluate the relative impact of degenerate trees across tree types.

For each (algorithm, dataset) pair, we average errors across all sampled scenarios. To assess statistical significance of performance differences across algorithms, we follow the two-step procedure of Demšar [11] as adopted in [34]: a Friedman test on the null hypothesis of equal performance, followed by a Nemenyi post-hoc test at the 5% significance level when the Friedman test is rejected. These tests are conducted on average AE across datasets, yielding 40 samples for binary tests and 17 samples for multiclass tests.

5 Results

5.1 Binary Quantification: Overall Performance

The CQB tree with SLD outperformed all other methods for binary quantification, achieving the best (smallest) mean rank (4.47 out of 21 methods) and lowest absolute error in 16 out of 40 datasets (Table 1, Figure 2). The next best performing tree-quantifier combination was Leaf EM applied to the Gini tree (mean rank: 4.70), followed by SLD applied to the CEB tree (mean rank: 4.72). The top performing quantifier varied by tree type, with SLD performing the best for the CQB and CEB trees and Leaf EM for the Gini tree. The performance of ACC was moderate, with a mean rank of 6.10–10.60 across all trees. Temperature Scaling (TS) did not improve SLD performance, indicating temperature scaling is not an effective calibration method for decision trees on our binary quantification tasks.

Both overall and within-tree Friedman tests indicated statistically significant differences in performance across quantification methods ($p < 0.01$). Within each tree type, the Nemenyi critical difference was 1.4245, and across all tree types, the critical difference was 4.9518. Nemenyi post-hoc tests indicated that the Gini tree with Leaf EM was significantly better than the Gini tree with SLD (TS),

PACC, and ACC. SLD when applied to the CQB and CEB trees was significantly better than any quantifier applied to those same trees. In all cases, ACC was significantly worse than the best performing quantifier for that tree type.

Most tree-quantifier pairs exhibited little bias (Figure 1) when averaged across datasets. SLD exhibited a weak bias towards a uniform distribution, while most other methods exhibited weak biases towards the extremes of the prevalence range.

Leaf EM performed best on the Gini tree (mean overall rank: 3.45), followed by a sharp dropoff by the CQB tree (mean rank: 13.66), and then the CEB tree (mean rank: 14.76). Across all quantifiers, the Gini tree tallied the most datasets with the lowest AE, indicating the Gini splitting criterion yields trees that are particularly well-suited for quantification. That is not to say that the CEB and CQB trees do not have their strengths; they each had the best performance for multiple datasets.

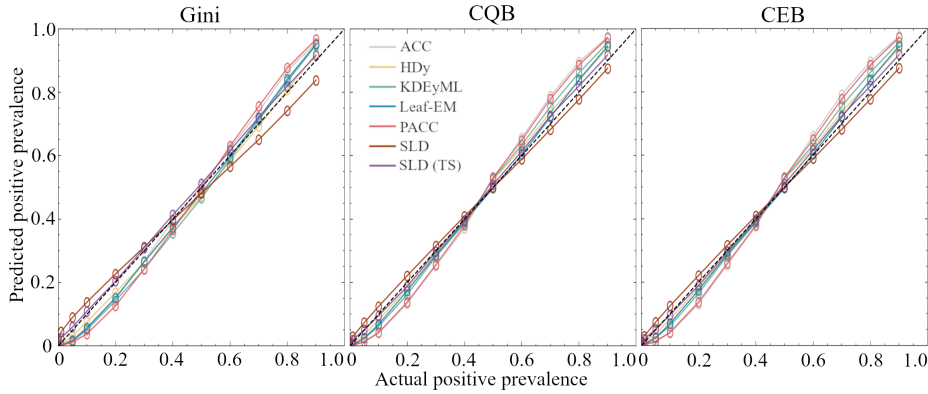


Fig. 1. Model Performance on Binary Quantification Tasks.

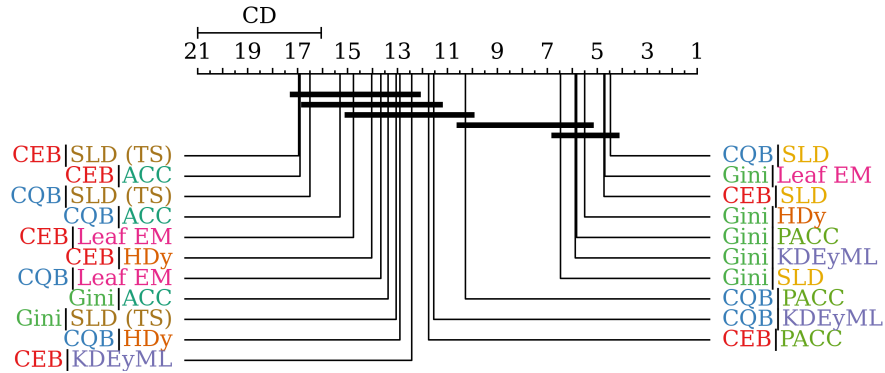


Fig. 2. Critical Difference Diagram for Binary Quantification Tasks.

Table 1. Absolute Error by Binary Dataset, Tree Type, and Algorithm. **Bold** indicates the lowest value in each row overall. *Italic* indicates the lowest value within each tree type per row. A Friedman test indicated statistical significance ($p < 0.05$) of performance differences. Summary rows report mean rank across datasets (Nemenyi critical difference: 4.9518) and mean rank within each tree type (Nemenyi critical difference: 1.4245).

Dataset	Gini					CQB					CEB							
	ACC	HDy	KDEyML	Leaf	EM	PACC	SLD	SLD	(TS)	ACC	HDy	KDEyML	Leaf	EM	PACC	SLD	SLD	(TS)
adult	0.058	0.033	0.038	0.030	0.045	0.031	0.041	0.075	0.074	0.074	0.076	0.072	0.047	0.080	0.077	0.074	0.074	0.079
avila	0.093	0.067	0.066	0.064	0.072	0.090	0.089	0.124	0.132	0.119	0.160	0.115	0.095	0.168	0.125	0.132	0.120	0.161
bike	0.104	0.074	0.069	0.070	0.076	0.071	0.088	0.112	0.104	0.101	0.111	0.103	0.080	0.125	0.112	0.106	0.102	0.115
blog	0.061	0.035	0.038	0.035	0.043	0.056	0.057	0.067	0.060	0.058	0.079	0.057	0.038	0.067	0.067	0.061	0.058	0.080
bc-cat	0.133	0.133	0.133	0.114	0.133	0.097	0.138	0.180	0.180	0.179	0.152	0.178	0.100	0.163	0.180	0.180	0.179	0.152
bc-cont	0.115	0.111	0.112	0.102	0.113	0.073	0.129	0.118	0.117	0.117	0.103	0.117	0.070	0.107	0.120	0.119	0.118	0.103
cars	0.211	0.121	0.124	0.149	0.137	0.087	0.128	0.195	0.178	0.179	0.181	0.187	0.115	0.192	0.222	0.204	0.205	0.202
conc	0.278	0.235	0.231	0.202	0.224	0.183	0.297	0.289	0.275	0.273	0.267	0.268	0.169	0.301	0.295	0.278	0.275	0.271
contra	0.429	0.383	0.379	0.361	0.368	0.318	0.458	0.368	0.369	0.359	0.381	0.353	0.271	0.405	0.380	0.374	0.370	0.389
cappl	0.204	0.134	0.136	0.134	0.145	0.101	0.182	0.287	0.269	0.267	0.255	0.263	0.169	0.300	0.302	0.280	0.277	0.267
ccard	0.214	0.090	0.091	0.089	0.095	0.141	0.204	0.223	0.238	0.222	0.283	0.206	0.194	0.293	0.225	0.239	0.222	0.286
diam	0.050	0.021	0.022	0.023	0.024	0.029	0.035	0.025	0.020	0.020	0.022	0.021	0.013	0.020	0.025	0.021	0.020	0.022
dota	0.527	0.363	0.408	0.489	0.249	0.492	0.573	0.399	0.430	0.441	0.500	0.320	0.456	0.365	0.412	0.422	0.438	0.502
drugs	0.244	0.146	0.147	0.125	0.139	0.110	0.183	0.222	0.220	0.213	0.216	0.214	0.154	0.253	0.240	0.235	0.225	0.230
ener	0.172	0.100	0.100	0.100	0.099	0.149	0.196	0.191	0.214	0.189	0.265	0.181	0.170	0.248	0.191	0.213	0.188	0.266
fifa	0.090	0.046	0.047	0.044	0.055	0.047	0.070	0.079	0.069	0.066	0.072	0.066	0.045	0.072	0.080	0.069	0.065	0.072
flare	0.473	0.430	0.424	0.414	0.417	0.316	0.487	0.428	0.422	0.419	0.438	0.421	0.281	0.474	0.428	0.418	0.416	0.433
grid	0.172	0.133	0.120	0.115	0.119	0.203	0.202	0.184	0.203	0.178	0.217	0.169	0.157	0.261	0.184	0.203	0.178	0.217
ads	0.147	0.132	0.133	0.135	0.133	0.088	0.155	0.177	0.173	0.172	0.162	0.172	0.111	0.182	0.177	0.174	0.172	0.162
magic	0.129	0.080	0.076	0.070	0.081	0.117	0.107	0.113	0.123	0.113	0.129	0.106	0.084	0.129	0.113	0.123	0.113	0.129
boone	0.022	0.015	0.022	0.016	0.019	0.050	0.035	0.029	0.026	0.027	0.037	0.026	0.020	0.029	0.029	0.026	0.027	0.037
mush	0.041	0.034	0.036	0.035	0.038	0.022	0.035	0.045	0.039	0.041	0.037	0.043	0.023	0.038	0.045	0.040	0.041	0.037
music	0.536	0.516	0.516	0.503	0.520	0.461	0.701	0.647	0.573	0.576	0.576	0.591	0.449	0.723	0.647	0.573	0.576	0.576
musk	0.164	0.150	0.148	0.143	0.149	0.152	0.187	0.199	0.197	0.185	0.193	0.183	0.152	0.204	0.203	0.201	0.188	0.196
news	0.295	0.175	0.180	0.196	0.139	0.313	0.468	0.248	0.282	0.277	0.395	0.208	0.286	0.450	0.248	0.283	0.279	0.395
nurse	0.000	0.000	0.001	0.013	0.000	0.006	0.000	0.000	0.000	0.001	0.013	0.000	0.006	0.000	0.000	0.000	0.000	0.013
occup	0.012	0.009	0.010	0.015	0.012	0.007	0.010	0.016	0.014	0.015	0.015	0.015	0.009	0.013	0.016	0.014	0.015	0.015
phish	0.056	0.032	0.033	0.046	0.040	0.025	0.034	0.063	0.062	0.062	0.060	0.062	0.043	0.065	0.064	0.063	0.063	0.061
craft	0.219	0.156	0.152	0.137	0.157	0.161	0.212	0.183	0.169	0.163	0.160	0.161	0.104	0.186	0.184	0.169	0.163	0.160
spam	0.107	0.094	0.095	0.087	0.097	0.083	0.119	0.119	0.112	0.108	0.101	0.108	0.070	0.120	0.121	0.114	0.109	0.103
alco	0.467	0.459	0.452	0.443	0.449	0.372	0.565	0.547	0.508	0.516	0.524	0.513	0.363	0.634	0.562	0.517	0.524	0.531
study	0.211	0.165	0.164	0.134	0.167	0.119	0.164	0.194	0.189	0.189	0.171	0.188	0.121	0.203	0.192	0.187	0.187	0.170
cond	0.050	0.036	0.037	0.031	0.038	0.045	0.041	0.076	0.072	0.066	0.070	0.068	0.049	0.079	0.076	0.072	0.066	0.070
telco	0.217	0.143	0.133	0.133	0.108	0.164	0.208	0.152	0.150	0.143	0.149	0.140	0.103	0.167	0.154	0.150	0.144	0.150
thrm	0.423	0.405	0.405	0.382	0.383	0.348	0.537	0.503	0.449	0.459	0.457	0.452	0.320	0.524	0.504	0.449	0.458	0.456
turk	0.507	0.408	0.418	0.391	0.383	0.398	0.535	0.379	0.378	0.350	0.370	0.350	0.301	0.344	0.399	0.383	0.360	0.370
vgame	0.252	0.158	0.154	0.154	0.155	0.181	0.245	0.204	0.205	0.195	0.214	0.187	0.146	0.258	0.206	0.202	0.196	0.215
voice	0.055	0.046	0.047	0.044	0.051	0.029	0.046	0.092	0.082	0.081	0.071	0.084	0.045	0.077	0.092	0.082	0.081	0.071
wine	0.255	0.177	0.175	0.160	0.172	0.201	0.279	0.232	0.218	0.205	0.240	0.201	0.173	0.257	0.236	0.223	0.208	0.245
yeast	0.382	0.307	0.308	0.298	0.307	0.257	0.408	0.445	0.404	0.406	0.412	0.409	0.315	0.481	0.454	0.414	0.418	0.418
Mean Rank	13.38	5.50	5.88	4.70	5.83	6.47	13.05	15.30	12.90	11.55	13.66	10.28	4.47	16.50	16.90	14.03	12.43	14.76
Mean Rank (per Tree)	6.30	3.20	3.38	2.35	3.80	3.12	5.85	5.53	4.22	3.55	4.50	3.08	1.35	5.78	5.55	4.20	3.45	3.27
																		1.35
																		5.75

5.2 Multiclass Quantification

Performance on multiclass quantification was quite different than on binary quantification (Table 2, Figure 3). The best performing method was SLD applied to the Gini tree, which achieved the lowest absolute error in 12 out of 17 datasets, and the lowest overall mean rank across all tree types. The best performing tree-quantifier combination was SLD applied to the Gini tree (mean rank: 1.65), followed by SLD applied to the CQB tree (mean rank: 2.88), and then SLD applied to the CEB tree (mean rank: 4.65). Across all trees, the ordinal ranking of the top three quantifiers was consistent: SLD performed best, followed by KDEyML, then PACC. The performance of Leaf EM was particularly poor, with a mean rank of 8.29–13.47 across all trees.

A Friedman test indicated statistically significant differences in performance across algorithms, both within each tree type ($p < 0.01$) and across all tree types ($p < 0.01$). The Nemenyi critical difference was 1.8288 within each tree type, and 6.3887 across all tree types. Nemenyi post-hoc tests indicated that SLD applied to the Gini tree was significantly better than Leaf EM, ACC, PACC, and SLD (TS) applied to the Gini tree.

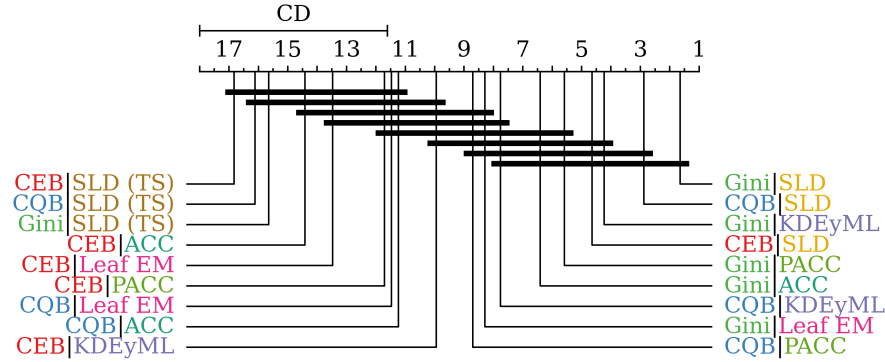


Fig. 3. Critical Difference Diagram for Multiclass Quantification Tasks.

5.3 Degenerate Trees

Overall, 30.4% of trials had at least one degenerate tree, with the Gini tree having the lowest percent of degenerate trials (14.1%) and the CQB and CEB trees having the highest (27.5% and 27.4%, respectively). A trial was considered degenerate if a tree had only a single node, which provides no information to the quantification methods. If these trials are included in the analysis, the ranks of Gini trees improve, with Leaf EM applied to Gini trees achieving the best overall mean rank (Table 3).

6 Discussion

The Gini tree with Leaf EM achieved the best mean rank across all quantifiers applied to a Gini tree for binary quantification (Table 1). The Nemenyi post-hoc

Table 2. Absolute Error by Multiclass Dataset, Tree Type, and Algorithm. **Bold** indicates the lowest value in each row overall. *Italic* indicates the lowest value within each tree type per row. A Friedman test indicated statistical significance ($p < 0.05$) of performance differences. Summary rows report mean rank across datasets (Nemenyi critical difference: 6.3887) and mean rank within each tree type (Nemenyi critical difference: 1.8288).

Dataset	Gini					CQB					CEB										
	ACC	KDEyML	Leaf	EM	PACC	SLD	SLD (TS)	ACC	KDEyML	Leaf	EM	PACC	SLD	SLD (TS)	ACC	KDEyML	Leaf	EM	PACC	SLD	SLD (TS)
bike	0.326	0.270	0.337	0.282	0.209	0.433	0.429	0.340	0.393	0.376	<i>0.275</i>	0.591	0.466	0.372	0.419	0.417	0.285	0.602			
blog	0.309	0.169	0.168	0.217	0.142	0.387	0.356	0.232	0.296	0.316	<i>0.195</i>	0.498	0.370	0.255	0.304	0.312	<i>0.204</i>	0.523			
conc	0.412	0.417	0.476	0.418	0.395	0.572	0.493	0.478	0.549	0.463	<i>0.410</i>	0.708	0.558	0.528	0.612	0.546	<i>0.448</i>	0.723			
contra	0.554	0.598	0.600	0.575	<i>0.475</i>	0.816	0.574	0.537	0.539	0.558	0.428	0.757	0.619	0.562	0.573	0.579	<i>0.442</i>	0.801			
diam	0.053	0.044	0.055	0.048	<i>0.037</i>	0.092	0.056	0.052	0.065	0.052	0.037	0.063	0.066	0.059	0.078	0.057	<i>0.048</i>	0.067			
drugs	0.323	0.329	0.324	0.319	0.267	0.447	0.366	0.364	0.390	0.347	<i>0.289</i>	0.455	0.461	0.444	0.482	0.432	<i>0.366</i>	0.620			
ener	0.292	0.266	0.339	0.243	0.311	0.597	0.320	0.341	0.472	0.281	0.333	0.632	0.344	0.352	0.518	<i>0.313</i>	0.314	0.625			
fifa	0.265	0.161	0.187	0.215	0.128	0.339	0.298	0.211	0.251	0.274	<i>0.158</i>	0.396	0.390	0.265	0.283	0.340	<i>0.191</i>	0.507			
news	0.593	0.426	0.438	0.514	0.383	1.018	0.644	0.548	0.585	0.579	<i>0.447</i>	0.995	0.692	0.605	0.625	0.618	<i>0.467</i>	0.959			
nurse	0.041	0.022	0.160	0.031	0.019	0.020	0.091	0.079	0.105	0.085	<i>0.054</i>	0.078	0.099	0.085	0.111	0.092	<i>0.063</i>	0.086			
craft	0.374	0.354	0.386	0.351	<i>0.341</i>	0.607	0.366	0.357	0.460	0.345	<i>0.305</i>	0.538	0.390	0.362	0.458	0.344	0.297	0.520			
cond	0.155	0.119	0.167	0.133	0.106	0.287	0.204	0.164	0.215	0.181	<i>0.133</i>	0.384	0.262	0.201	0.227	0.209	<i>0.140</i>	0.393			
thrm	0.702	0.713	0.778	0.707	0.590	1.007	0.779	0.751	0.852	0.770	<i>0.611</i>	0.974	0.788	0.762	0.858	0.778	<i>0.623</i>	0.930			
turk	0.621	0.576	0.597	0.582	<i>0.478</i>	0.980	0.690	0.639	0.667	0.658	0.453	1.158	0.702	0.636	0.664	0.667	<i>0.471</i>	1.173			
vgame	0.574	0.566	0.577	0.596	0.437	0.994	0.606	0.586	0.578	0.613	<i>0.465</i>	0.974	0.607	0.566	0.570	0.617	<i>0.468</i>	0.930			
wine	0.639	0.676	0.749	0.708	0.518	1.077	0.712	0.694	0.727	0.727	<i>0.545</i>	1.051	0.771	0.732	0.783	0.772	<i>0.622</i>	1.047			
yeast	0.443	0.439	0.462	0.464	0.278	0.707	0.525	0.553	0.588	0.558	<i>0.412</i>	0.811	0.599	0.589	0.628	0.599	<i>0.471</i>	0.836			
Mean Rank	6.41	4.24	8.29	5.59	1.65	15.65	11.24	7.76	11.47	8.71	<i>2.88</i>	16.12	14.41	9.94	13.47	11.71	<i>4.65</i>	16.82			
Mean Rank (per Tree)	3.59	2.71	4.41	3.35	1.18	5.76	4.12	2.53	4.41	3.12	<i>1.12</i>	5.71	4.35	2.29	4.35	3.18	1.06	5.76			

Table 3. Overall mean rank for binary (Nemenyi critical difference: 4.95) and multi-class (Nemenyi critical difference: 6.39) settings. Bold indicates the best overall method within a setting (comparing all tree-method combinations in that setting); italic indicates the best within each tree type.

Setting	Tree	ACC	HDy	KDEyML	Leaf EM	PACC	SLD	SLD (TS)
Binary	Gini	9.9	4.7	5.3	4.0	5.8	4.3	10.7
	CQB	16.2	13.5	12.8	14.2	11.5	<i>6.3</i>	17.1
	CEB	17.1	14.1	12.9	15.0	12.2	<i>6.2</i>	17.3
Multi	Gini	6.3	–	4.4	8.5	5.8	1.6	15.5
	CQB	11.1	–	7.7	11.5	8.7	<i>3.0</i>	16.1
	CEB	14.3	–	9.9	13.5	11.6	<i>4.7</i>	16.9

test indicated that the Gini–Leaf EM combination was significantly better than ACC, PACC, and SLD (TS) applied to the Gini tree. Within the Gini tree, Leaf EM was not significantly better than KDEyML, SLD, or HDy. SLD applied to CQB performed the best among all quantifier-tree combinations on binary quantification, but CQB and CEB trees had a higher percentage of degenerate trials than Gini trees, indicating it might not be an ideal choice for small, imbalanced datasets. Consistent with how certain quantifiers are best for certain base classifiers, certain splitting criteria are best for certain quantifiers. Leaf EM performed poorly in the multiclass setting. Ultimately, the best tree-quantifier combination will depend on the dataset and the quantification task.

SLD applied to Gini trees performed relatively poorly on binary quantification. This aligns with the expectation that SLD’s performance is dependent on the calibration of the base classifier. SLD maximizes the same likelihood function as Leaf EM, but SLD depends on the calibration of $P(Y | X)$ and is therefore subject to the miscalibration of the base classifier. Decision trees are known to produce poorly calibrated probabilities, and Leaf EM avoids this dependency. Poor calibration is often attributed to the purity driven splitting used to fit a tree [37]. CEB and CQB are splitting criteria that are not directly tied to increasing the purity of the resulting leaves. It follows that CEB and CQB trees are likely to produce better calibrated probabilities than Gini trees, and therefore produce better SLD estimates of prevalence.

The Gini tree outperformed the CEB or CQB trees when used as a base classifier for Leaf EM. The CEB and CQB splitting criteria were designed for the quantification task, while Gini was designed for classification. However, Gini impurity can be directly tied to the Leaf EM approach. Gini impurity at a $Leaf_i$ can be written in terms of class conditionals as

$$G(Leaf_i) = 1 - \sum_{i=1}^{n_Y} \left(\frac{P(Leaf_i | Y_i) P(Y_i)}{\sum_{i'=1}^{n_Y} P(Leaf_i | Y_{i'}) P(Y_{i'})} \right)^2, \quad (4)$$

which is minimized when one $P(Leaf_i | Y_i)$ dominates the others. Gini’s split criterion is therefore used to maximize the purity of the same class conditionals $P(Leaf | Y)$ that Leaf EM consumes.

Leaf EM and KDEyML perform similarly, particularly when applied to the Gini tree. KDEyML approaches the same likelihood function as Leaf EM as

the kernel bandwidth approaches zero, provided no two leaves produce identical classifier scores. The two methods were applied here without hyperparameter tuning: KDEyML used a fixed bandwidth of 0.05. Had the KDEyML bandwidth been tuned, it is plausible that the optimal value would have been close to zero, in which case KDEyML would have approximated Leaf EM directly. The competitive performance of KDEyML on the Gini tree indicates KDEyML may be learning a similar representation of class conditional distributions, but recovering it through a different mechanism.

Schumacher et al. reported that quantification forests and their ACC-corrected variant performed poorly relative to other quantification methods [34]. The results presented here suggest that this failure is attributable in part to the choice of aggregation method rather than to a fundamental limitation of quantification trees. When CEB and CQB trees are paired with most quantifiers rather than ACC, performance improves substantially.

This study was subject to several limitations. Neither Leaf EM nor any of the comparator methods were subjected to hyperparameter tuning. KDEyML used a fixed bandwidth of 0.05, and trees were pruned with a fixed minimum of 5 samples per leaf on the held-out estimation set. Moreo et al. found that tuning the KDEyML bandwidth can substantially improve performance [28]. Additionally, temperature scaling is used to calibrate SLD. While temperature scaling is commonly used in quantification, it is typically evaluated on continuous scores (e.g., from a neural network) rather than discrete tree-leaf predictions. Future work could explore alternative calibration methods that are designed for decision trees, such as kernel density smoothing proposed by Smyth et al. [35]. Finally, removing trials with degenerate trees from the analysis may have biased the results in favor of quantifiers that perform particularly poorly when fit to small, imbalanced datasets.

7 Conclusion

We introduced Leaf EM, a quantification method that pairs decision-tree leaves with Iterative Bayesian Unfolding and requires no assumption of classifier calibration. On binary tasks, SLD applied to a CQB tree achieved the best overall mean rank, while the Gini–Leaf EM combination attained the second best overall mean rank and the best within-tree mean rank on Gini trees. On multiclass tasks, SLD applied to a Gini tree was the strongest tree-quantifier combination. Together, these findings demonstrate that recent reports of poor decision-tree performance on quantification reflect the choice of aggregator, and not necessarily a limitation of decision trees themselves.

Acknowledgments. This study was funded by The National Institute of Allergy and Infectious Diseases (grant number R01AI174932).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bella, A., Ferri, C., Hernández-Orallo, J., Ramirez-Quintana, M.J.: Quantification via probability estimators. In: 2010 IEEE International conference on data mining. pp. 737–742. IEEE (2010)
2. Blobel, V.: Unfolding methods in high-energy physics experiments. Tech. rep., CERN (1984)
3. Blobel, V.: An unfolding method for high energy physics experiments. arXiv preprint hep-ex/0208022 (2002)
4. Börner, M., Hoinka, T., Meier, M., Menne, T., Rhode, W., Morik, K.: Measurement/simulation mismatches and multivariate data discretization in the machine learning era. *Astronomical Data Analysis Software and Systems XXVII* **522**, 431 (2020)
5. Breiman, L., Friedman, J., Olshen, R.A., Stone, C.J.: Classification and regression trees. Chapman and Hall/CRC (1984)
6. Bunse, M.: On multi-class extensions of adjusted classify and count. In: Proceedings of the 2nd International Workshop on Learning to Quantify (LQ 2022). pp. 43–50 (2022)
7. Bunse, M.: Unification of algorithms for quantification and unfolding. In: INFORMATIK 2022. pp. 459–468. Gesellschaft für Informatik, Bonn (2022)
8. Castaño, A., Alonso, J., González, P., del Coz, J.J.: An equivalence analysis of binary quantification methods. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 6944–6952 (2023)
9. D’Agostini, G.: A multidimensional unfolding method based on Bayes’ theorem. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **362**(2-3), 487–498 (1995)
10. D’Agostini, G.: Improved iterative Bayesian unfolding. arXiv preprint arXiv:1010.0632 (2010)
11. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* **7**(Jan), 1–30 (2006)
12. Esuli, A., Fabris, A., Moreo, A., Sebastiani, F., et al.: Learning to quantify. *The information retrieval series* **47** (2023)
13. Esuli, A., Molinari, A., Sebastiani, F.: A critical reassessment of the saeren-slatinne-decaestecker algorithm for posterior probability adjustment. *ACM Transactions on Information Systems (TOIS)* **39**(2), 1–34 (2020)
14. Esuli, A., Moreo, A., Sebastiani, F., Sperduti, G.: A concise overview of LeQua@CLEF 2022: learning to quantify. In: International Conference of the Cross-Language Evaluation Forum for European Languages. pp. 362–381. Springer (2022)
15. Esuli, A., Moreo, A., Sebastiani, F., Sperduti, G.: An overview of LeQua 2024, the 2nd International Data Challenge on Learning to Quantify. In: 4th International Workshop on Learning to Quantify (LQ 2024). p. 51 (2024)
16. Fabris, A., Esuli, A., Moreo, A., Sebastiani, F.: Measuring fairness under unawareness of sensitive attributes: A quantification-based approach. *Journal of Artificial Intelligence Research* **76**, 1117–1180 (2023)
17. Forman, G.: Counting positives accurately despite inaccurate classification. In: European conference on machine learning. pp. 564–575. Springer (2005)
18. Gart, J.J., Buck, A.A.: Comparison of a screening test and a reference test in epidemiologic studies. II. A probabilistic model for the comparison of diagnostic tests. *American Journal of Epidemiology* **83**(3), 593–602 (May 1966). <https://doi.org/10.1093/oxfordjournals.aje.a120610>

19. González, P., Castaño, A., Chawla, N.V., Coz, J.J.D.: A review on quantification learning. *ACM Computing Surveys (CSUR)* **50**(5), 1–40 (2017)
20. González, P., Castaño, A., Peacock, E.E., Díez, J., Del Coz, J.J., Sosik, H.M.: Automatic plankton quantification using deep features. *Journal of Plankton Research* **41**(4), 449–463 (2019)
21. González-Castro, V., Alaiz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the hellinger distance. *Information Sciences* **218**, 146–164 (2013)
22. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: *International conference on machine learning*. pp. 1321–1330. PMLR (2017)
23. Hoecker, A., Kartvelishvili, V.: SVD approach to data unfolding. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **372**(3), 469–481 (1996)
24. Hopkins, D.J., King, G.: A method of automated nonparametric content analysis for social science. *American Journal of Political Science* **54**(1), 229–247 (2010)
25. King, G., Lu, Y.: Verbal autopsy methods with multiple causes of death (2008)
26. Milli, L., Monreale, A., Rossetti, G., Giannotti, F., Pedreschi, D., Sebastiani, F.: Quantification trees. In: *2013 IEEE 13th International Conference on Data Mining*. pp. 528–536. IEEE (2013)
27. Moreo, A., Esuli, A., Sebastiani, F.: QuaPy: a python-based framework for quantification. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. pp. 4534–4543 (2021)
28. Moreo, A., González, P., del Coz, J.J.: Kernel density estimation for multiclass quantification. *Machine Learning* **114**(4), 92 (2025)
29. Moreo, A., Salvati, N.: An efficient method for deriving confidence intervals in aggregative quantification. In: *5th International Workshop on Learning to Quantify (LQ 2025)*. p. 12 (2025)
30. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011)
31. Provost, F.J., Fawcett, T., Kohavi, R., et al.: The case against accuracy estimation for comparing induction algorithms. In: *ICML*. vol. 98, pp. 445–453 (1998)
32. Sadinle, M., Bouyarmane, K., Galloway, G., Sam, S., Yuan, C., Tutar, I.: Semi-automated estimation of weighted rates for e-commerce catalog quality monitoring (2022), <https://www.amazon.science/publications/semi-automated-estimation-of-weighted-rates-for-e-commerce-catalog-quality-monitoring>
33. Saerens, M., Latinne, P., Decaestecker, C.: Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation* **14**(1), 21–41 (2002)
34. Schumacher, T., Strohmaier, M., Lemmerich, F.: A comparative evaluation of quantification methods. *Journal of Machine Learning Research* **26**(55), 1–54 (2025)
35. Smyth, P., Gray, A., Fayyad, U.M.: Retrofitting decision tree classifiers using kernel density estimation. In: *Machine Learning Proceedings 1995*, pp. 506–514. Elsevier (1995)
36. Tasche, D.: Fisher consistency for prior probability shift. *Journal of Machine Learning Research* **18**(95), 1–32 (2017)
37. Zadrozny, B., Elkan, C.: Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In: *Proceedings of the Eighteenth International Conference on Machine Learning*. p. 609–616. ICML '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001)

Out-of-Known-Safes Detection as a Benchmark for Runtime Monitors

Robbe De Groeve[✉], Laurens Devos[✉], and Mathias Verbeke[✉]

KU Leuven, Belgium

`{robbe.degroeve,laurens.devos,mathias.verbeke}@kuleuven.be`

Abstract. Runtime monitors safeguard ML models during deployment by deciding whether to pass a prediction to a downstream system or trigger a fallback, providing a defense against, e.g., dataset shift. Selecting the best monitor for an application requires setting up a benchmark. This paper identifies a bias in existing benchmarking frameworks rooted in how they handle covariate-shifted samples. These samples span a continuum from close to the training data to severe distortions that cause model failures. Existing benchmarks force monitors to either accept or reject covariate-shifted samples, incentivizing monitors to expand or contract their acceptance region, biasing monitor selection toward either higher false-accept rates, where dangerous predictions slip through, or excessive rejection of valid inputs. We introduce the Out-Of-Known-Safes (OOKS) benchmarking framework, a monitor evaluation methodology which excludes correctly classified covariate-shifted samples from evaluation rather than forcing an arbitrary accept or reject label on ambiguous cases. Experiments on CIFAR-10, CIFAR-100, and ImageNet across nine monitoring techniques demonstrate that benchmark design meaningfully changes monitor rankings.

Keywords: Model runtime monitoring · Benchmarking · Safety.

1 Introduction

Deep learning models are increasingly deployed in real-world systems where prediction errors carry operational, safety, or regulatory consequences. This challenge is particularly acute in safety-critical applications such as autonomous driving [40,4], medical diagnosis [37], and manufacturing [23], where models must continue operating safely even when deployment conditions differ from those seen during development [33,11,3,43].

A **runtime monitor** addresses this challenge by deciding for each prediction whether it should be used or withheld in favor of a fallback [31,11]. From a safety perspective, predictions should be accepted only when the model has been evaluated on that type of input and shown to behave safely [5]; predictions outside this known-safe operating region should instead be rejected.

Runtime monitors encounter three broad categories of inputs. *In-distribution (ID)* samples originate from the training distribution and split into correctly and incorrectly classified examples. *Semantically shifted (S-OOD)* samples lie outside the model’s intended task, while *covariate-shifted (C-OOD)* samples preserve task semantics but differ through changes such as blur, noise, or domain shift. The central challenge of runtime monitoring is deciding which of these samples should be accepted or rejected.

Historically, these categories have been studied separately [36]. Conventional *out-of-distribution* (OOD) detection focuses on distinguishing in-distribution data from out-of-distribution inputs, primarily targeting semantic shifts [21,24,43]. In contrast, *classification with a reject option (CRO)* [17] evaluates a classifier’s ability to abstain from uncertain predictions on in-distribution data [9,14,34].

Recent work has begun combining these perspectives. Examples include *full-spectrum OOD (FS-OOD)* [44], *G-ODIN* [24], *model-specific OOD (MS-OOD)* [2], and *selective classification under OOD data (SCOD)* [42,12]. While these frameworks all evaluate monitors across mixtures of ID, S-OOD, and C-OOD samples, they fundamentally disagree on how correctly classified covariate-shifted samples should be treated (Figure 1).

Covariate shifts form a continuum ranging from benign variations to severe corruptions, making it unrealistic for a runtime monitor to determine precisely where expected model generalization ends and unsafe behavior begins. Existing benchmarking frameworks nevertheless require these correctly classified C-OOD samples to be either accepted (MS-OOD) or rejected (G-ODIN). Both choices introduce optimization bias: rewarding acceptance encourages more permissive monitors with higher false-accept rates, while rewarding rejection favors overly conservative monitors with unnecessary false rejections.

This paper argues that correctly classified covariate-shifted samples should not influence runtime monitor evaluation. We introduce the Out-Of-Known-Safes (OOKS) benchmarking framework, which deliberately excludes these ambiguous samples from scoring (Figure 1, rightmost panel (g)). Instead of rewarding monitor behavior in this gray area, OOKS encourages deployment-

	(a) OOD	(b) CRO	(c) FS-OOD	(d) G-ODIN	(e) SCOD	(f) MS-OOD	(g) OOKS (ours)
Accept	ID ✓ ID ✗	ID ✓	ID ✓ ID C-OOD ✓ C-OOD ✗	ID ✓ ID ✗	ID ✓	ID ✓ C-OOD ✓	ID ✓ C-OOD ✓
Reject	S-OOD	ID ✗	S-OOD	C-OOD ✓ C-OOD ✗ S-OOD	ID ✗ S-OOD	ID ✗ C-OOD ✗ S-OOD	C-OOD ✓ ID ✗ C-OOD ✗ S-OOD

Fig. 1: Comparison of the accept/reject behavior across different runtime monitoring frameworks: conventional OOD, CRO [9], FS-OOD [44], G-ODIN [24], SCOD [42,12], MS-OOD [2] and OOKS. ID, S-OOD, and C-OOD denote in-distribution, semantically shifted, and covariate-shifted samples; ✓ and ✗ indicate correctly and incorrectly classified samples. The primary divergence lies in how frameworks label correctly and incorrectly classified covariate-shifted samples, C-OOD ✓ and C-OOD ✗ respectively. With our proposed OOKS framework, we argue that C-OOD ✓ should be excluded from monitor evaluation to prevent optimization biases that yield dangerous *false accepts*.

specific covariate variations to be incorporated directly into model development whenever they are expected during operation.

We make the following contributions:

1. We introduce the OOKS benchmarking framework to evaluate, tune, and compare runtime monitors.
2. We empirically show that benchmark design meaningfully affects monitor selection, and trace this bias to differing assumptions about correctly classified covariate-shifted samples.

2 Background & related work

This paper focuses on runtime monitors for image classification.¹ We consider classifiers f that map inputs $x \in X$ to labels $Y = \{1, \dots, C\}$, and runtime monitors g that score inputs based on f 's activations and outputs (e.g., softmax probabilities [21] or feature embeddings [28]). Inputs with $g(x, f) \leq \tau$ are *accepted* and passed through; inputs with $g(x, f) > \tau$ are *rejected* and routed to a fallback. A higher threshold τ is therefore more permissive.

Monitor evaluation arises when candidate monitors must be ranked and selected, whether for hyperparameter tuning or for choosing among entirely different monitoring strategies. This evaluation is performed on a dataset covering the range of inputs the monitor will encounter at deployment. Given a fixed

¹ The underlying ideas are not limited to classification and can be extended to other tasks such as regression or object detection and segmentation.

evaluation dataset, the final monitor ranking is entirely determined by what the benchmarking framework rewards; a benchmark that rewards the wrong behavior will consistently select a sub-optimal monitor.

2.1 Benchmarking monitors: choosing ground-truth monitor labels

A runtime monitor can be viewed as a binary classifier: it either accepts or rejects an incoming example. To evaluate a monitor, a benchmarking framework assigns ground-truth *monitor labels*. For each input x_i in an evaluation dataset, a *monitor label*² $z_i \in \{\text{accept}, \text{reject}\}$ represents the desired monitor action. Given a threshold τ , the monitor predicts $\hat{z}_i = \mathbb{1}[g(x, f) > \tau]$ (where $\hat{z}_i = 1$ denotes *reject*), and performance is measured by comparing $\hat{z}_i$ to z_i across the evaluation set using a chosen performance metric. The resulting ranking is clearly determined by the specific ground-truth rules a benchmarking framework enforces. Figure 1 gives an overview of how each framework labels the different categories of examples.

2.2 The safety perspective: maintaining known-safe operation

The goal of a runtime monitor is to decide whether a classifier’s prediction should be passed through to the downstream system or filtered out. From a safety-critical perspective, runtime monitoring can be understood as maintaining a *known-safe operating zone* around the monitored classifier.

Under this view, a prediction should be passed through only when it belongs to a scenario for which the model is known to behave safely. We call such a prediction *known safe*: the model’s performance on that type of scenario has been thoroughly evaluated, and the observed performance satisfies task-specific safety criteria. An ideal monitor g accepts known-safe cases and otherwise triggers:

$$g(x) = \begin{cases} 0 & \text{if } x \text{ belongs to a known-safe scenario,} \\ 1 & \text{otherwise.} \end{cases}$$

The corresponding trigger conditions cover both *known unsafe* scenarios (where the model is known to fail) and *unevaluated* scenarios for which no sufficient evidence about model behavior is available. Samples without such evidence fall outside the known-safe operating zone, even if the model might happen to guess correctly on some of them.

2.3 Relation to existing benchmarking frameworks

The growing number of runtime monitoring techniques has created a need for benchmarks that evaluate monitors across realistic deployment scenarios. Existing frameworks differ mainly in how they categorize deployment sample scenarios and how they expect the monitor to behave within those scenarios.

² This is called the model-specific acceptance/rejection (MS-A/R) in MS-OOD [2].

Deployment sample scenarios are commonly organized into in-distribution (ID) data, semantically-shifted out-of-distribution (S-OOD), and covariate-shifted out-of-distribution (C-OOD) samples [29,32]. In-distribution data matches the development layout, semantic shifts contain content outside the intended task scope, and covariate shifts preserve core task semantics while introducing feature corruptions like noise, blur, or illumination changes.

Conventional and Full-Spectrum OOD detection. Conventional out-of-distribution detection separates clean in-distribution data from out-of-distribution data. While well-suited to capturing semantic shifts, it does not directly capture a classifier’s mistakes on clean data, nor does it effectively handle the continuum of covariate corruptions [15]. Several works argue that runtime monitors should not reject covariate shifts because a robust model is expected to generalize to benign variations [44,1,22]. While the full-spectrum (FS-OOD) benchmarking framework [44] incorporates covariate-shifted data, its metrics remain primarily distribution-centric: they evaluate where a sample originates rather than whether the model is safe on it, creating a fundamental disconnect from the actual deployment objective.

For example, OpenOOD v1.5 [45] is a popular evaluation toolkit in this space. To maintain consistency with these established evaluation setups, our concrete OOKS evaluation setup uses the standard dataset selections and image corruption variants popularized by benchmarks like OpenOOD v1.5. However, the core novelty of OOKS lies entirely in its distinct evaluation methodology.

Classification with a reject option. Classification with a reject option, also known as selective classification, rejects predictions estimated to exceed a risk threshold while maximizing coverage [14,13]. However, classical formulations usually assume that the deployment data perfectly matches the clean development distribution. Extensions evaluating selective classification under distribution shifts exist [26], but they do not explicitly address the safety implications of diverse, unvetted covariate-shifted scenarios.

2.4 Resolving the covariate shift dilemma: MS-OOD and G-ODIN

Recent literature suggests that the ongoing dilemma regarding how to handle covariate-shifted examples can be resolved by evaluating correctness and distribution shifts simultaneously [2]. Specifically, combining out-of-distribution detection and correctness prediction ensures that misclassified covariate shifts (C-OOD ✗) are cleanly categorized as unsafe and targeted for rejection. However, there is no agreement on how to treat correctly-classified covariate shift (C-OOD ✓). There are two prominent options in existing literature:

- **MS-OOD (Model-Specific OOD)** argues that because the underlying model classifies C-OOD ✓ samples correctly, the monitor should be rewarded for accepting them [2]. The target monitor label for these samples is thus *accept*.
- **G-ODIN** conservatively rejects all out-of-distribution samples regardless of model accuracy; the target monitor label for C-OOD ✓ samples is thus *reject*.

3 Out-of-Known-Safes (OOKS) Benchmarking Framework

Depending on what a benchmarking framework rewards, different monitors will be selected as the best-performing option during hyperparameter tuning or architectural selection. This paper contends that correctly classified covariate-shifted samples reflect observed generalization rather than verified, systemic safety. Because forcing a hard accept or reject label on this gray zone introduces arbitrary optimization preferences, our proposed Out-Of-Known-Safes (OOKS) framework explicitly excludes these benign covariate shifts from scored evaluation. This design choice keeps monitor selection tightly focused on the operating boundaries where safety signals are unambiguous.

3.1 Why labeling correctly classified covariate shifts as *accept* biases monitor selection

Assigning a ground-truth label of $z_i = \text{accept}$ to correctly classified covariate-shifted samples, as the model-specific out-of-distribution (MS-OOD) framework does, creates a systematic bias toward monitors with higher false-accept rates. This occurs because the decision threshold τ is artificially driven toward more lenient values. For any runtime monitor g and threshold τ , the rates at which correctly classified covariate shifted (C-OOD ✓) samples and incorrectly classified covariate shifted (C-OOD ✗) samples are accepted are fundamentally driven by the same underlying score distribution. Incentivizing a high acceptance rate for C-OOD ✓ inevitably forces τ upward, dragging along the unintended acceptance of dangerous C-OOD ✗ failures. Consequently, a benchmarking framework that rewards high acceptance of C-OOD ✓ therefore systematically favors monitors that are more permissive on covariate-shifted data as a whole, including on the incorrectly classified samples that should be rejected.

OOKS operates on a foundational safety principle: *a monitor should accept a prediction only when the model has been rigorously evaluated on that specific type of input and its behavior has been verified as safe*. This translates into the acceptance of ID ✓ samples while ID ✗, C-OOD ✗ and S-OOD samples would need to be rejected.

Samples belonging to C-OOD ✓ occupy an ambiguous middle ground. Although the base model happens to classify them correctly, they originate from shifted environmental conditions whose safety has not necessarily been established. They represent opportunistic generalization rather than verified safety. Some may be mild, benign variations close to the in-distribution design domain, while others might sit perilously close to catastrophic model breakdowns or semantic anomalies. Because benchmarking frameworks provide no oracle boundary to separate safe generalization from impending failure, imposing a hard ground-truth label introduces severe trade-off biases:

- Labeling C-OOD ✓ as *accept* (the MS-OOD paradigm) is too lenient from a safety perspective. These inputs sit outside the verified operating zone

of the model. Rewarding a monitor for expanding its acceptance boundary over unvetted domains increases the probability of false accepts on highly corrupted, misclassified samples nearby.

- Conversely, labeling C-OOD ✓ as *reject* (the G-ODIN paradigm) is overly conservative. A model that successfully navigates mild environmental variations is executing exactly what robust generalization is intended to achieve. Treating these successes as failures penalizes system availability and favors overly restrictive monitors.

OOKS resolves these opposing tendencies by leaving C-OOD ✓ samples entirely unscored. While passing such an instance during deployment is not a failure in isolation (since the model output is correct), the benchmark should not actively penalize a monitor for filtering it out if a stricter stance is required to guarantee the rejection of nearby safety violations.

We illustrate this ambiguity and the resulting threshold bias with a Gaussian-based monitoring technique [28] fitted on CIFAR-10 [27]. We compare dataset-wide monitor score distributions over the full, clean in-distribution test set versus covariate-shifted test sets corrupted by Gaussian noise at two distinct severities. Figure 2 shows representative single-image examples of these visual corruptions, while Figure 3 tracks the macro-level dataset score distributions.



Fig. 2: CIFAR-10 examples under increasing Gaussian-noise corruption strength.

For each dataset score distribution, we can construct the mathematically optimal accept/reject threshold corresponding to the three different evaluation methodologies for handling correctly classified covariate-shifted samples: MS-OOD, G-ODIN, or our proposed OOKS.

As shown in Figure 3 (left), when corruptions are mild, optimizing for a distribution-centric framework like G-ODIN establishes a highly conservative threshold (τ_1) that rejects viable samples. In this regime, the optimal thresholds for MS-OOD and OOKS naturally align. However, under heavy corruptions (right), the MS-OOD metric skews the threshold drastically toward leniency (τ_2) to capture the heavily shifted C-OOD ✓ population, resulting in a dangerous surge of false accepts on failed model predictions. By omitting these ambiguous samples from the objective function, OOKS maintains a stable safety boundary that adapts to corruption without swinging toward dangerous permissiveness or crippling over-conservatism.

These opposing effects demonstrate why forcing a hard accept or reject label for correctly classified covariate-shifted samples introduces systematic bias.

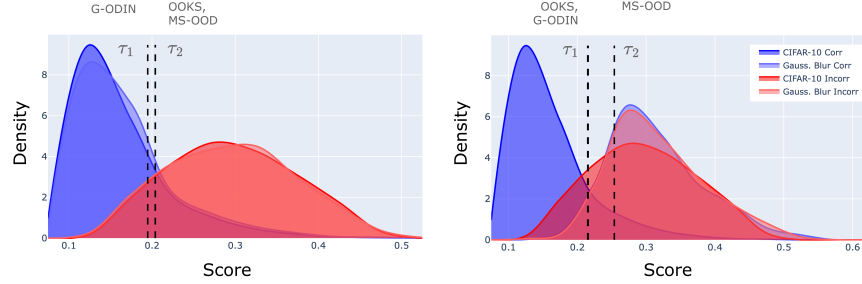


Fig. 3: Score distribution for ID ✓ (dark blue), ID ✗ (dark red), C-ODD ✓ (light blue), C-ODD ✗ (light red). Under mild corruption (left), G-ODIN yields a stricter threshold (τ_1) than MS-ODD and OOKS, whose optimal thresholds coincide. Under stronger corruption (right), the MS-ODD setting pulls the threshold in a lenient direction (τ_2), increasing false accepts relative to the two other settings. Threshold τ_1 is preferred by OOKS, while MS-ODD ranks τ_2 better, despite its higher false-accept rate and potential safety violations.

OOKS ignores these samples altogether, allowing the benchmark to avoid being too conservative or too permissive.

3.2 A ROC argument for excluding correctly classified covariate shifts

The bias can be made precise using a simple probabilistic argument. For any threshold, define the true accept rate as the fraction of correctly-classified samples that pass through, and the false accept rate as the fraction of incorrectly-classified ones that do. The curve tracing these two rates against each other as the threshold varies is a ROC curve, and its area has a natural interpretation: AUROC is the probability that the monitor assigns a more in-distribution score to a correctly-classified sample than to an incorrectly-classified one. Unless $\text{AUROC} = 1$, the score distributions overlap, meaning any threshold that accepts some correctly-classified C-ODD samples will inevitably also accept some incorrectly-classified ones.

This creates a structural problem for MS-ODD. By including C-ODD ✓ with an *accept* label, its objective rewards higher true accept rates in a way the OOKS objective does not. Since both the true and false accept rates are non-decreasing in the threshold, this pushes the MS-ODD-optimal threshold upward, and with it the false accept rate on unsafe C-ODD samples. The same logic applies to monitor selection: MS-ODD systematically favors monitors with higher true accept rates on C-ODD ✓, which by the C-ODD ROC structure implies higher false accept rates on C-ODD ✗. The magnitude of this bias scales directly with score overlap: it disappears only when $\text{AUROC} = 1$, and is largest when monitor is not predictive for C-ODD correctness.

4 Experiments

This paper answers the following experimental questions:

- Q1. Do the benchmarking frameworks rank different runtime monitors differently, resulting in a different selected monitor?
- Q2. Can a structural bias in failure and unavailability rates for the selected monitors be observed between MS-OOD, G-ODIN and OOKS?

4.1 Experimental setup

Datasets. We evaluate on three widely used vision datasets: CIFAR-10 [27], CIFAR-100 [27], and ImageNet1k [8]. For each benchmark setup, the original training and validation data define the in-distribution (ID) setting. We use the following additional datasets for covariate-shifted out-of-distribution (C-OOD) scenarios:

CIFAR-10 & CIFAR-100: The covariate-shifted scenarios use six corruption families: Gaussian & impulse noise, Gaussian & motion blur, and brightness & contrast shift. We follow the standard corruption-severity protocol from Hendrycks and Dietterich [20] to test monitor behavior under varying covariate-shift strengths.

ImageNet1k: The covariate-shifted scenarios are built from ImageNet-C, ImageNet-R [19], and ImageNet-V2 [35]. ImageNet-C uses the same corruptions as CIFAR, but combines them to test general corruption detection. ImageNet-R captures rendition and style shifts, while ImageNet-V2 captures distribution refresh and re-sampling effects.

Monitored classifiers. We monitor three classifiers, one per dataset. We base all three models on the same ResNet-50 [16] backbone. The model weights are obtained from Hugging Face [10]. The model cards report test accuracies of 94.65% on CIFAR-10 [6], 80.93% on CIFAR-100 [7], and ImageNet top-1/top-5 accuracies of 80.38%/94.6% at 224px evaluation resolution [39].

Runtime monitoring techniques. We evaluate post-hoc monitoring techniques that use confidence, logits, features, and gradients. Confidence-based monitors include entropy (Ent) and maximum softmax probability (Soft) [21]. Logit-based monitors include maximum logit (Logit) [18] and energy scoring (Energy) [30]. Feature-space monitors include KNN [38], Gaussian discriminant scoring (Gauss) [28], Virtual-logit Matching (ViM) [41], and GradNorm (Grad) [25].

All monitors are fit using only in-distribution training data. We split the original training set into ID_{Fit} and ID_{Tune} : ID_{Fit} is used to estimate monitor-specific quantities such as feature statistics or nearest-neighbor reference sets, while ID_{Tune} is used for threshold calibration. Thresholds are set to achieve $TNR(ID_{Tune} \checkmark) = 95\%$, ensuring a shared operating point across all monitors.

Evaluation metrics. For OOKS, the monitor labels are accept for ID ✓ and reject for ID ✗, C-OOD ✗, and S-OOD samples, while C-OOD ✓ samples are excluded from the metric. For the baseline benchmarking frameworks (MS-OOD and G-ODIN), we use the framework-specific accept/reject labels shown in Figure 1; this mirrors the model-specific accept/reject labeling idea used by MS-OOD [2], but changes the label mapping according to the framework under evaluation.

Let $A_\tau(S) = \{x \in S \mid g(x, f) \leq \tau\}$ denote accepted samples and $R_\tau(S) = \{x \in S \mid g(x, f) > \tau\}$ denote rejected samples for a dataset S . We treat rejection as the positive monitor decision and acceptance as the negative monitor decision. At the tuned threshold τ , we report:

$$\begin{aligned} \text{TNR}(\text{ID } \checkmark) &= \frac{|A_\tau(\text{ID } \checkmark)|}{|\text{ID } \checkmark|}, & \text{TPR}(\text{ID } \times) &= \frac{|R_\tau(\text{ID } \times)|}{|\text{ID } \times|}, \\ \text{TPR}(\text{C-ODD } \times) &= \frac{|R_\tau(\text{C-ODD } \times)|}{|\text{C-ODD } \times|}, & \text{TPR}(\text{S-ODD}) &= \frac{|R_\tau(\text{S-ODD})|}{|\text{S-ODD}|}. \end{aligned}$$

These quantities correspond respectively to true-negative rate on known-safe in-distribution samples and true-positive rates on known-unsafe or unevaluated samples. Equivalently, $1 - \text{TNR}(\text{ID } \checkmark)$ is the false-reject rate on known-safe samples, while $1 - \text{TPR}(\text{ID } \times)$, $1 - \text{TPR}(\text{C-ODD } \times)$, and $1 - \text{TPR}(\text{S-ODD})$ are false-accept rates on samples that should be filtered out under OOKS.

In the covariate-shift-focused experiments below, we do not separately evaluate monitoring performance on semantically shifted samples, because the benchmarking frameworks considered here differ only in how they treat covariate-shifted scenarios. A comprehensive monitor evaluation should nevertheless also include semantic shifts.

4.2 Results

Q1: Monitor ranking and technique selection. To answer Q1, we evaluate whether the choice of benchmarking framework (MS-OOD, G-ODIN, or OOKS), specifically how correctly classified covariate-shifted samples are handled, leads to different rankings between runtime monitors and alters which technique is ultimately selected.

The empirical results confirm that benchmarking framework choice affects monitor selection. As shown in Table 1, OOKS, MS-OOD, and G-ODIN disagree on the best monitor in 7 of the 15 covariate-shift cases.

Compared to OOKS, MS-OOD selects a different monitor in 5 covariate-shifted cases across CIFAR-10, CIFAR-100, and ImageNet1k, while G-ODIN diverges in 2. This demonstrates that a benchmarking framework’s monitor labels for covariate shifts materially affect monitor selection.

Q2: Differences in failure and unavailability rates between MS-OOD, G-ODIN, and OOKS. To address Q2, we analyze whether these different

Table 1: Runtime monitors selected by MS-OD, G-ODIN, and OOKS (*ours*) for different covariate-shift cases. Cases highlighted with * or † have different selected runtime monitors for MS-OD and G-ODIN compared to OOKS, respectively.

CIFAR-10				
C-OD dataset	MS-OD	G-ODIN	OOKS	
Brightness	Ent	Ent	Ent	
Contrast	Soft	Logit	Soft	†
Gaussian blur	VIM	VIM	VIM	
Gaussian noise	VIM	Gauss	Gauss	*
Impulse noise	VIM	Gauss	Gauss	*
Motion blur	KNN	VIM	VIM	*

CIFAR-100				
C-OD dataset	MS-OD	G-ODIN	OOKS	
Brightness	KNN	KNN	KNN	
Contrast	KNN	VIM	KNN	*
Gaussian blur	VIM	VIM	VIM	
Gaussian noise	VIM	VIM	VIM	
Impulse noise	VIM	VIM	VIM	
Motion blur	Gauss	VIM	VIM	*

ImageNet1k				
C-OD dataset	MS-OD	G-ODIN	OOKS	
ImageNet-C	Ent	Ent	Ent	
ImageNet-R	VIM	VIM	VIM	
ImageNetv2	GradNorm	Ent	Ent	†

rankings are indicative of an underlying bias toward higher failure rates (more false accepts) or higher system unavailability (more false rejects). The failure and unavailability rates of the runtime monitors selected by MS-OD, G-ODIN, and OOKS are given in Table 2.

For all 5 cases where MS-OD selected a different top monitor in Q1, we observe a shift in failure and unavailability rate: the MS-OD-selected monitor achieves a lower unavailability rate at the direct cost of a higher failure rate. This matches theoretical expectations, as prioritizing the acceptance of C-OD ✓ samples structurally biases the selection toward looser thresholds or monitors that favor throughput over safety. Notably, this lower unavailability rate factors in all correct predictions on the covariate-shifted dataset, which includes “lucky guesses” of the classifier alongside true generalization.

For the 2 cases where G-ODIN selected a different top monitor in Q1, the G-ODIN-selected monitoring technique has both a higher failure rate and a higher unavailability rate. In these two cases, penalizing the monitor for not

Table 2: Covariate-shift monitor-selection system-level consequences under alternative treatments of correctly classified covariate-shifted samples according to MS-ODD, G-ODIN and OOKS (*ours*). For each covariate-shifted scenario, the table reports the resulting failure and unavailability rates of the selected monitors. Best (lowest) values are in bold.

CIFAR-10						
C-ODD dataset	Failure rate			Unavailability rate		
	MS-ODD	G-ODIN	OOKS	MS-ODD	G-ODIN	OOKS
Brightness	0.3988	0.3988	0.3988	0.0592	0.0592	0.0592
Contrast	0.378	0.3851	0.378	0.0682	0.0866	0.0682
Gauss. blur	0.1334	0.1334	0.1334	0.2438	0.2438	0.2438
Gauss. noise	0.1168	0.1083	0.1083	0.4171	0.4434	0.4434
Impulse noise	0.1652	0.1578	0.1578	0.347	0.361	0.361
Motion blur	0.409	0.4083	0.4083	0.3261	0.3319	0.3319

CIFAR-100						
C-ODD dataset	Failure rate			Unavailability rate		
	MS-ODD	G-ODIN	OOKS	MS-ODD	G-ODIN	OOKS
Brightness	0.4711	0.4711	0.4711	0.0838	0.0838	0.0838
Contrast	0.315	0.3257	0.315	0.1212	0.134	0.1212
Gauss. blur	0.29	0.29	0.29	0.1776	0.1776	0.1776
Gauss. noise	0.1634	0.1634	0.1634	0.3542	0.3542	0.3542
Impulse noise	0.1062	0.1062	0.1062	0.3731	0.3731	0.3731
Motion blur	0.3818	0.3676	0.3676	0.2925	0.3072	0.3072

ImageNet1k						
C-ODD Dataset	Failure rate			Unavailability rate		
	MS-ODD	G-ODIN	OOKS	MS-ODD	G-ODIN	OOKS
ImageNet-C	0.293	0.293	0.293	0.1366	0.1366	0.1366
ImageNet-R	0.2674	0.2674	0.2674	0.2575	0.2575	0.2575
ImageNetv2	0.5867	0.5662	0.5662	0.4133	0.4338	0.4338

accepting correctly classified covariate-shifted samples resulted in a clear Pareto suboptimal monitor selection.

OOKS consistently selects the monitoring technique with the lowest overall failure rate. By ignoring C-ODD ✓ samples, our benchmarking framework eliminates the trade-off, ensuring that monitor selection is driven purely by a safety-oriented criterion without penalizing benign deployment shifts.

5 Conclusion

Runtime monitors are only useful in safety-critical machine learning systems if their evaluation rewards the behavior that such systems actually require: accepting known-safe predictions while filtering out known-unsafe and unevalu-

ated ones. This paper argued that existing benchmarking frameworks’ handling of correctly classified covariate-shifted samples leads to biased runtime monitor comparison. We introduced OOKS as a safety-oriented benchmarking framework that avoids penalizing monitors for rejecting or accepting these types of samples.

Our empirical results show that this distinction matters in practice. Changing the benchmarking framework changes monitor rankings and changes which monitoring technique is selected. In particular, MS-ODD tends to favor monitors with lower unavailability but higher failure rates. OOKS instead consistently selects monitors with lower failure rates, giving a cleaner safety signal for deployment settings in which false accepts are more costly than unnecessary fallbacks. The G-ODIN-based selection results further show that benchmark assumptions can interact with monitor score distributions in non-intuitive ways, sometimes worsening both failure and unavailability.

Future work should study this interaction more explicitly through the severity of covariate shift. In this paper, we used mixed corruption severities to approximate unstructured real-world deployment conditions, where mild and severe shifts can occur together. However, separating mild from severe covariate shifts would make it possible to test the intuition underlying the different benchmarking framework choices: mild covariate shift may often resemble recoverable generalization, whereas severe covariate shift is more likely to fall outside the known-safe operating zone. Extending the Q1 and Q2 analyses with severity-stratified experiments could therefore clarify when MS-ODD, G-ODIN, and OOKS select different monitors, and why the G-ODIN cases in particular deviate from the expected safety-availability tradeoff.

References

1. Ahmed, F., Courville, A.: Detecting semantic anomalies. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 3154–3162 (2020)
2. Averly, R., Chao, W.L.: Unified out-of-distribution detection: A model-specific perspective. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 1453–1463 (2023)
3. Bayram, F., Ahmed, B.S., Kassler, A.: From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems* **245**, 108632 (2022)
4. Chib, P.S., Singh, P.: Recent Advancements in End-to-End Autonomous Driving Using Deep Learning: A Survey. *IEEE Transactions on Intelligent Vehicles* **9**(1), 103–118 (2024)
5. Chow, C.K.: On optimum recognition error and reject tradeoff. *IEEE Trans. Inf. Theory* **16**, 41–46 (1970)
6. Dadalto, E.: Resnet50_cifar10 (2024), https://huggingface.co/edadaltocg/resnet50_cifar10
7. Dadalto, E.: Resnet50_cifar100 (2024), https://huggingface.co/edadaltocg/resnet50_cifar100
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255. Ieee (2009)
9. El-Yaniv, R., Wiener, Y.: On the foundations of noise-free selective classification. *Journal of Machine Learning Research* **11**(53), 1605–1641 (2010)
10. Face, H.: Hugging face hub (2021)
11. Ferreira, R.S., Guérin, J., Delmas, K., Guiochet, J., Waeselynck, H.: Safety monitoring of machine learning perception functions: A survey. *Computational Intelligence* **41**(2), e70032 (2025)
12. Franc, V., Paplham, J., Prusa, D.: Scod: From heuristics to theory. In: Computer Vision – ECCV 2024. pp. 424–441 (2025)
13. Galil, I., Dabbah, M., El-Yaniv, R.: What can we learn from the selective prediction and uncertainty estimation performance of 523 imagenet classifiers? In: ICLR (2023)
14. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: Advances in Neural Information Processing Systems. vol. 30 (2017)
15. Guerin, J., Delmas, K., Ferreira, R., Guiochet, J.: Out-of-distribution detection is not all you need. Proceedings of the AAAI Conference on Artificial Intelligence **37**(12), 14829–14837 (2023)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
17. Hendrickx, K., Perini, L., Van der Plas, D., Meert, W., Davis, J.: Machine learning with a reject option: A survey. *Machine Learning* **113**(5), 3073–3110 (2024)
18. Hendrycks, D., Basart, S., Mazeika, M., Zou, A., Mostajabi, M., Steinhardt, J., Song, D.X.: Scaling out-of-distribution detection for real-world settings. In: International Conference on Machine Learning (2022)
19. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8340–8349 (2021)

20. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations (2019)
21. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: International Conference on Learning Representations (2017)
22. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15262–15271 (2021)
23. Hesabi, H., Nourelfath, M., Hajji, A.: A deep learning predictive model for selective maintenance optimization. *Reliability Engineering & System Safety* **219**, 108191 (2022)
24. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
25. Huang, R., Geng, A., Li, Y.: On the Importance of Gradients for Detecting Distributional Shifts in the Wild. In: Advances in Neural Information Processing Systems (2021)
26. Kim, J., Koo, J., Hwang, S.: A unified benchmark for the unknown detection capability of deep neural networks. *Expert Systems with Applications* **229**, 120461 (2023)
27. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
28. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) *Advances in Neural Information Processing Systems*. vol. 31. Curran Associates, Inc. (2018)
29. Liang, H., Peng, L., Sun, J.: Selective Classification Under Distribution Shifts. *Transactions on Machine Learning Research* (2024)
30. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based Out-of-distribution Detection. In: *Advances in Neural Information Processing Systems*. vol. 33, pp. 21464–21475. Curran Associates, Inc. (2020)
31. Machin, M., Guiochet, J., Waeselynck, H., Blanquart, J.P., Roy, M., Masson, L.: Smof: A safety monitoring framework for autonomous systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **48**(5), 702–715 (2018)
32. Mucsányi, B., Kirchhof, M., Oh, S.J.: Benchmarking uncertainty disentanglement: Specialized uncertainties for specialized tasks. In: Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C. (eds.) *Advances in Neural Information Processing Systems*. vol. 37, pp. 50972–51038. Curran Associates, Inc. (2024)
33. Paterson, C., Hawkins, R., Picardi, C., Jia, Y., Calinescu, R., Habli, I.: Safety assurance of machine learning for autonomous systems. *Reliability Engineering & System Safety* **264**, 111311 (2025)
34. Pugnana, A., Ruggieri, S.: Auc-based selective classification. In: *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. vol. 206, pp. 2494–2514 (2023)
35. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do ImageNet classifiers generalize to ImageNet? In: *International Conference on Machine Learning*. pp. 5389–5400 (2019)

36. Salehi, M., Mirzaei, H., Hendrycks, D., Li, Y., Rohban, M.H., Sabokrou, M.: A unified survey on anomaly, novelty, open-set, and out of-distribution detection: Solutions and future challenges. *Transactions on Machine Learning Research* (2022)
37. Shen, D., Wu, G., Suk, H.I.: Deep Learning in Medical Image Analysis. *Annual Review of Biomedical Engineering* **19**(Volume 19, 2017), 221–248 (2017)
38. Sun, Y., Ming, Y., Zhu, X., Li, Y.: Out-of-distribution detection with deep nearest neighbors. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) *Proceedings of the 39th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 162, pp. 20827–20840. PMLR (2022)
39. timm: resnet50.a1_in1k (2026), https://huggingface.co/timm/resnet50.a1_in1k
40. Ullrich, L., Buchholz, M., Dietmayer, K., Graichen, K.: Toward fully autonomous driving: Ai, challenges, opportunities, and needs. *IEEE Access* **14**, 17971–17997 (2026)
41. Wang, H., Li, Z., Feng, L., Zhang, W.: ViM: Out-Of-Distribution with Virtual-logit Matching. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 4911–4920. IEEE (2022)
42. Xia, G., Bouganis, C.S.: Augmenting softmax information for selective classification with out-of-distribution data. In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. pp. 1995–2012 (2022)
43. Yang, J., Zhou, K., Li, Y., Liu, Z.: Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision* **132**(12), 5635–5662 (2024)
44. Yang, J., Zhou, K., Liu, Z.: Full-spectrum out-of-distribution detection. *International Journal of Computer Vision* **131**(10), 2607–2622 (2023)
45. Zhang, J., Yang, J., Wang, P., Wang, H., Lin, Y., Zhang, H., Sun, Y., Du, X., Zhou, K., Zhang, W., Li, Y., Liu, Z., Chen, Y., Li, H.: OpenOOD v1.5: Enhanced benchmark for out-of-distribution detection. In: *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models* (2024)

Assessing the Impact of Dataset Shift for Automatic *Nephrops norvegicus* Burrows Detection and Classification Through Deep Learning

Oscar Papini¹, Marco Reggiannini¹, Enrico Cecapolli², Filippo Domenichetti², Lorenzo Zacchetti^{2,3}, Michela Martinelli², and Gabriele Pieri¹

¹ Institute of Information Science and Technologies (ISTI), National Research Council of Italy, Via G. Moruzzi 1, 56124 Pisa, Italy

{oscar.papini,marco.reggiannini,gabriele.pieri}@isti.cnr.it

² Institute for Marine Biological Resources and Biotechnology (IRBIM), National Research Council of Italy, Largo Fiera della Pesca 2, 60125 Ancona, Italy

{enrico.cecapolli,lorenzo.zacchetti}@irbim.cnr.it

{filippo.domenichetti,michela.martinelli}@cnr.it

³ Department of Biological, Geological, and Environmental Sciences (BiGeA), Alma Mater Studiorum – University of Bologna, Piazza di Porta S. Donato 1, 40126 Bologna, Italy

Abstract In this work, machine learning and computer vision methods are investigated to support the estimation of the *Nephrops norvegicus* (Norway lobster) population, a key task for sustainable fisheries management and marine ecosystem monitoring. This task is currently carried out by manually counting the animals' burrows that are visually recognisable in video surveys captured in the area of interest, and then using those counts to estimate the species density in the area. To reduce uncertainty in human assessment, a burrow classifier based on a supervised learning model has been developed to be integrated into the pipeline. A central challenge in deploying such models in real-world marine environments is dataset shift, arising from variations in both acquisition conditions and ecological contexts. Indeed, several factors, including water turbidity, irregular lighting, and alterations of colour, potentially hinder the accuracy of the detection. Furthermore, geographic and temporal variability, such as differences between survey campaigns or regions, can exacerbate these shifts, leading to reduced generalisation. This work concerns the application of the proposed classifier to data acquired in two different areas: the Adriatic Sea and the northern Atlantic Ocean, off the coast of Ireland. As shown in the paper, the differences in conformation and species type between those two areas greatly affect the performances of the classifier. Nevertheless, the proposed method can be employed in the entire range of the species distribution, provided that the model is properly fine-tuned for the specific geomorphological features of the area in which it is applied.

Keywords: Dataset shift · *Nephrops norvegicus* · Underwater Television surveys · Deep Learning

1 Introduction

Nephrops norvegicus (Norway lobster) is a burrowing crustacean inhabiting the continental shelf of the northeastern Atlantic Ocean and the Mediterranean Sea, at typical depths ranging between 20 and 800 metres [1]. It is among the most commercially relevant species in Europe, with landings amounting to about 60 000 tonnes per year, even though the trend seems to be decreasing in recent years [2].

Towed Underwater Television (UWTV) surveys are one of the most accredited methods conventionally adopted by many European countries for stock assessment purposes, as they allow estimating the burrows density (number of burrows per square metre), which is considered a proxy for the population abundance [3]. The estimation is provided by trained analysts who visually inspect hours of underwater video footage of a known seabed area to recognise and count the number of *N. norvegicus* burrows. The inspection of a video is usually performed independently by multiple operators, and the outcome is returned as the average of the individual measurements, which is eventually employed to estimate the burrows density.

Since this procedure relies on the direct assessment of human operators, the outcome is prone to subjectivity and uncertainty errors. To attenuate these factors, the International Council for the Exploration of the Sea (ICES) issues specific guidelines and promotes training workshops intended to standardise the video reading methodologies [4]. Despite these efforts of the international scientific community, the counting task keeps being an entirely manual and time-consuming process, as well as highly sensitive to human errors [5]. Undoubtedly, an automatic tool that is able to detect and count *N. norvegicus* burrows would be invaluable, both in terms of manual labour relief and of reduced subjectivity. Nonetheless, this is a challenging task even for state-of-the-art object detection models such as YOLO [6].

Among the issues that an automatic detector of *N. norvegicus* burrows can run into, the vast environmental diversity that can be found in the *N. norvegicus* habitat is one of the main ones. Indeed, the differences in the biological features of the animals, as well as in the geophysical properties of the sea floor, induce a strong differentiation in the visual characteristics of the burrow entrances, to the point that even expert human annotators may have difficulty recognising them in footage collected from geographic regions unfamiliar to them. Moreover, despite having to follow a common standard protocol, the equipment used by the various research institutions for the UWTV recording differs in terms of cameras setup and specifications [7].

In this work, the domain generalization capability of a deep learning-based *N. norvegicus* burrows classifier are investigated through the use of two datasets originating from two different geographic areas: the Pomo/Jabuka Pits area in

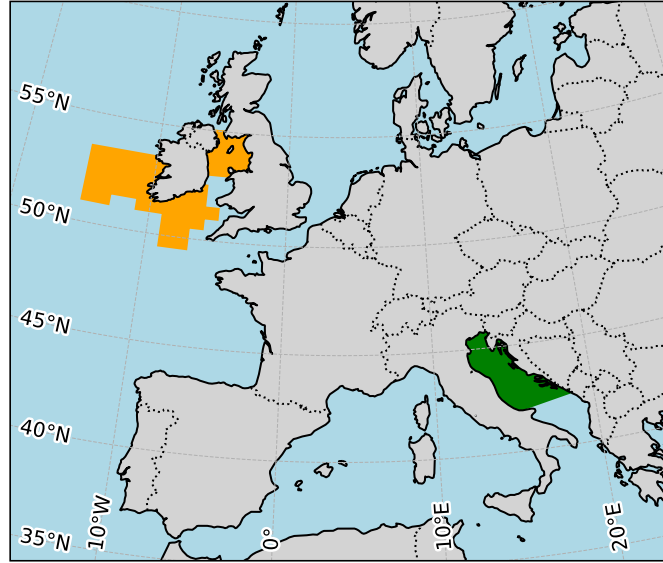


Figure 1: Acquisition zones for the two datasets (green: Adriatic dataset; orange: Irish dataset).

the Adriatic Sea (Central Mediterranean basin), which represents a Vulnerable Marine Ecosystem and an Essential Fish Habitat for various commercially and ecologically important species [8, 9], and the North Atlantic Ocean off the coast of Ireland (see Figure 1). The chosen deep learning model has been described in previous works by the same authors [6] and consists of the cascaded sequence of an encoding block, in charge of extracting features and providing a latent representation of the data, followed by a fully connected block that performs the actual recognition of the candidate burrow entrance. Those two blocks are trained independently to provide both within-region and cross-region evaluations for the classification task.

The remainder of this paper is organised as follows. Section 2 describes the two datasets that are used in this work; Section 3 presents the deep learning-based detection pipeline; Section 4 details the setup of the experiments; in Section 5 the results are reported and briefly discussed; finally, Section 6 concludes the paper and outlines future research directions.

2 Datasets

As mentioned in the Introduction, the distribution of *N. norvegicus* habitats involve marine environments which are greatly diverse in terms of geomorphological characteristics. Moreover, footage is captured by different institutions using different setups in terms of e.g. camera angle and illumination apparatus. These

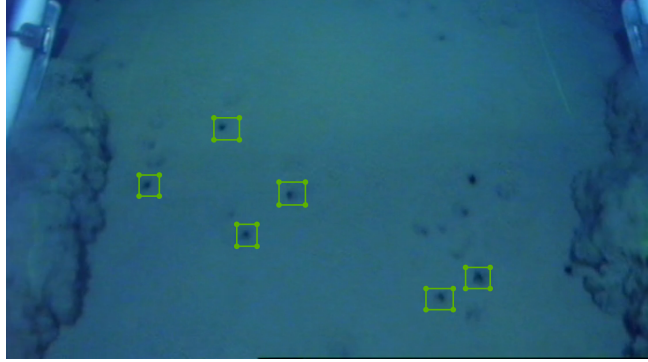


Figure 2: Example of an annotated frame in the Adriatic dataset. Green boxes denote NEPHROPS objects.

differences do not affect the traditional *N. norvegicus* burrows counting method, but automated computer vision models may learn the peculiarities of a specific institute or location instead of the relevant features of *N. norvegicus* burrows entrances.

In this study, two datasets acquired in different geographic domains are used to train the same deep learning architecture and assess the robustness of the resulting models.

2.1 Adriatic dataset

Starting from footage collected during surveys conducted by IRBIM-CNR in the Central Adriatic Sea, more precisely in the Pomo/Jabuka Pits area, a set of 1110 frames has been extracted and, following the guidelines in [10], the images were annotated by tracing a bounding box, labelled NEPHROPS (N), around objects recognised as entrances of *N. norvegicus* burrows. An example of an annotated frame is provided in Figure 2.

Subsequently, a custom automatic segmentation method has been used to identify other objects in the field of view that may be confused with *N. norvegicus* entrances, such as other species' entrances or roundish shadows; these objects have been automatically marked with bounding boxes with a dedicated NON-NEPHROPS (NN) label. In the end, only 793 images contain at least one bounding box, for a total of 717 NEPHROPS and 1240 NON-NEPHROPS objects, but all the 1110 frames are part of the final dataset.

2.2 Irish dataset

A dataset from footage collected during surveys in the Irish Sea, the Celtic Sea and the Northern Atlantic Ocean around Ireland has been published by the Irish Marine Institute [11]. It consists of 468 annotated images with labelled bounding boxes belonging to the following classes:

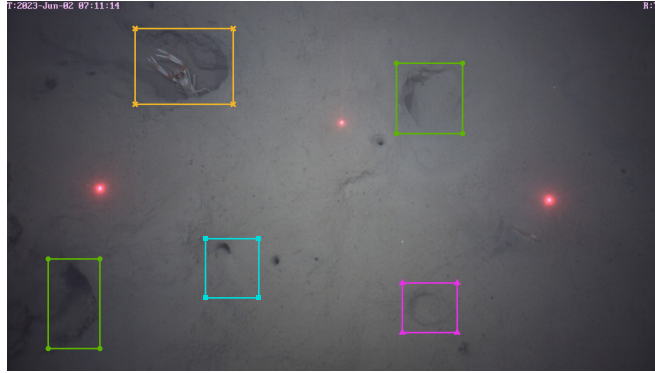


Figure 3: Example of an annotated frame in the Irish dataset. Green boxes with circle corners denote PRAWN_BURROW objects; cyan boxes with square corners denote SMALL_PRAWN_BURROW objects; magenta boxes with triangle corners denote CLOSED_BURROW objects; yellow boxes with cross corners denote GATEKEPT_BURROW objects.

- PRAWN_BURROW (P), corresponding to entrances associated with *N. norvegicus* burrows (872 bounding boxes);
- SMALL_PRAWN_BURROW (sP), also corresponding to entrances associated with *N. norvegicus* burrows, albeit smaller in size (548);
- GATEKEPT_BURROW (Gk), corresponding to *N. norvegicus* burrow entrances where a specimen of *N. norvegicus* (or a part of it, e.g. a pincer) is visible in the entrance (79);
- CLOSED_BURROW (Cl), corresponding to *N. norvegicus* entrances of abandoned burrows that have collapsed (173);
- CRAB_BURROW (Cr), corresponding to entrances associated with burrows of other species (99).

An example of an annotated frame is provided in Figure 3.

2.3 Tiled datasets

Most object detectors work at *image* level, that is to say, a whole frame is fed to the model and a list of candidate objects, in the form of bounding boxes, is returned. In this work, the problem of detecting *N. norvegicus* burrows entrances is tackled as a classification problem: given a portion of suitable dimensions (a *tile*) obtained as a crop of an image similar to the frames extracted from UWTV footage, determine whether it contains an entrance of a *N. norvegicus* burrow by giving a (global) label to the tile. To do so, two datasets of labelled square tiles have been derived from the Adriatic and the Irish datasets through the following procedure:

1. to make the two datasets comparable, images have been preprocessed to take into account the perspective distortion due to the camera inclination and different image acquisition sizes;
2. the images have been divided into 60×60 px tiles with 30 px overlap (this size was selected empirically to be comparable with the average area occupied by the entrances in the images);
3. each tile is given a label depending on the labels of the bounding boxes intersecting it, and an additional BACKGROUND (BG) label is given to tiles that do not intersect any bounding box.

In particular, three labels are possible for tiles from the Adriatic dataset (NEPHROPS, NON-NEPHROPS and BACKGROUND), and six labels for tiles from the Irish dataset (the five labels of [11], plus BACKGROUND).

3 Models

The pipeline outlined in this work consist of two separate phases: a feature extraction step followed by a classification step. The first step was performed using a type of neural networks known as *autoencoders* (AEs), whose main task is to learn an efficient representation, or *encoding*, of data, usually belonging to a lower-dimensional domain (sometimes called *latent space*). AEs are composed of an *encoder* and a *decoder*, the latter being the portion devoted to reconstruct the original data from the latent representation. Since AEs do not use labelled data, it is possible to train them without the involvement of the expert annotators.

The philosophy of the second step, instead, is to use the learned representation as the basis for the classification task—in other words, instead of using samples of the form (image, label) for the supervised training of the classification model, the samples have the form (encoding, label). Thus, the weights of the trained encoder are frozen, and any classifier can be chained to the encoder and trained. In this work, a simple multilayer perceptron (MLP), i.e., a stack of fully connected linear layers, is used as the classifying model.

In summary, the overall training process can be described as (see Figure 4):

1. train an AE using a large number of (unlabelled) tiles and freeze its weights;
2. train a classifier (e.g. a MLP) using as inputs the encodings of the labelled tiles with respect to the AE.

All the experiments have been carried out within a PyTorch (version 2.6.0) framework. Tables 1 and 2 detail the architectures of the networks involved in this study; in those tables, the output of a layer is fed directly to the next one. We remark that this paper represents mainly a proof of concept, and these architectures will be further analysed and improved in future works.

4 Training setup

To assess the effects of the different visual characteristics on the latent features, two different autoencoder models with the same architecture were trained independently, once using tiles from the Adriatic dataset and once using Irish

Table 1: List of autoencoder layers.

Layer	Parameters	Output dim.
Encoder		
(input)	–	$3 \times 60 \times 60$
CONV2D($C = 8, k = 3, s = 1, p = 0$)	224	$8 \times 58 \times 58$
LEAKYReLU	–	–
CONV2D($C = 16, k = 3, s = 2, p = 0$)	1168	$16 \times 28 \times 28$
LEAKYReLU	–	–
CONV2D($C = 32, k = 3, s = 2, p = 0$)	4640	$32 \times 13 \times 13$
LEAKYReLU	–	–
FLATTEN	–	5408
LINEAR($n = 5408, m = 1024$)	5 538 816	1024
LEAKYReLU	–	–
LINEAR($n = 1024, m = 128$)	131 200	128
LEAKYReLU	–	–
Decoder		
(input)	–	128
LINEAR($n = 128, m = 1024$)	132 096	1024
LEAKYReLU	–	–
LINEAR($n = 1024, m = 5408$)	5 543 200	5408
LEAKYReLU	–	–
UNFLATTEN	–	$32 \times 13 \times 13$
UPSAMPLE($n = 2$)	–	$32 \times 26 \times 26$
CONVTRANSPOSE2D($C = 16, k = 3, s = 1, p = 0$)	4624	$16 \times 28 \times 28$
LEAKYReLU	–	–
UPSAMPLE($n = 2$)	–	$16 \times 56 \times 56$
CONVTRANSPOSE2D($C = 8, k = 3, s = 1, p = 0$)	1160	$8 \times 58 \times 58$
LEAKYReLU	–	–
CONVTRANSPOSE2D($C = 3, k = 3, s = 1, p = 0$)	219	$3 \times 60 \times 60$
SIGMOID	–	–

Table 2: List of perceptron layers. n_c refers to the number of output classes ($n_c = 3$ for Adriatic dataset, $n_c = 6$ for Irish dataset).

Layer	Parameters	Output dim.
(input)	–	128
LINEAR($n = 128, m = 64$)	8256	64
LEAKYReLU	–	–
LINEAR($n = 64, m = 16$)	1040	16
LEAKYReLU	–	–
LINEAR($n = 16, m = n_c$)	$17 \cdot n_c$	n_c
SOFTMAX	–	–

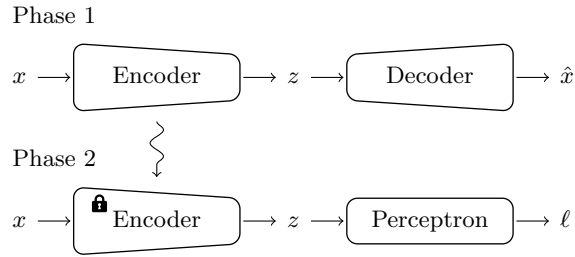


Figure 4: Concept of training phases. Here x is the original tile, z is the tile encoding, $\hat{x}$ is the reconstructed tile, and ℓ is the predicted label.

tiles. More in detail, in both cases a set of 16384 tiles was randomly selected from the relative dataset and used for training. To reduce the impact of almost uniform tiles, which bring little information, the standard deviation σ of the pixels intensity of each tile was computed, and the selection has been carried out so that a substantial part of the chosen tiles shows sufficient variability. The AEs were trained to maximise their tile reconstruction capability using the MSE loss, with a small L^1 loss component added to disfavour an average blurry reconstruction and preserve contrast. A further set of 4096 tiles, with the same variability distribution as the training set, was selected for each dataset to be used as a validation set for the implementation of an early stopping policy.

Perceptrons in the second phase were trained using the standard multi-class cross-entropy as the loss function. A stratified 5-fold cross-validation scheme is implemented: five sets of 768 tiles were extracted independently from both the Adriatic and the Irish datasets, so that a suitable number of samples are represented for each class. Using these sets, five models are trained with one of the folds acting as validation set for early stopping and the union of the others as training set. Another set of 480 tiles, with the same labels proportion as the training folds, is extracted from each dataset to be used as a test set after the training of each model. The tiles were selected at random while ensuring that all tiles extracted from the same frame were assigned to either the training set or the test set, to prevent potential overfitting due to spatial correlations.

To evaluate the transferability of the learned representation, all the four possible combinations of AE/MLP training were considered, namely (a) AE and MLP both trained on Adriatic tiles; (b) AE trained on Irish tiles, MLP trained on Adriatic tiles; (c) AE trained on Adriatic tiles, MLP trained on Irish tiles; (d) AE and MLP both trained on Irish tiles.

5 Results and discussion

Figure 5 shows some examples of reconstructions of tiles coming from different locations, allowing a qualitative evaluation of the autoencoders reconstruction effectiveness. The results of the perceptrons training experiments are shown

in Tables 3 and 4. In those tables, “AE/MLP Adriatic” (respectively, “Irish”) means “AE/MLP trained on the Adriatic dataset” (respectively, on the Irish dataset). The confusion matrices in Table 3 display the 5-fold average of the *test* sets classifications, where each cell (i, j) represents the number of tiles having ground truth (G.T.) label i predicted as label j . In Table 4, the capability of the MLPs to correctly identify *N. norvegicus* burrows entrances is assessed using as evaluation metrics the precision and recall computed for training, validation and test sets limited to the labels related with active *N. norvegicus* burrows (i.e., NEPHROPS for the Adriatic dataset; PRAWN_BURROW, SMALL_PRAWN_BURROW and GATEKEPT_BURROW for the Irish one, combining the three classes in the latter case).

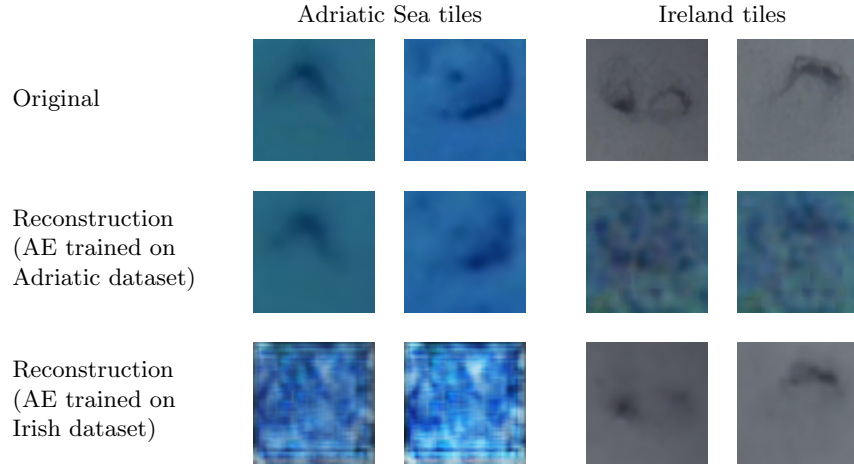


Figure 5: Examples of autoencoder reconstructions.

When evaluating the quality of reconstructions, AEs consistently demonstrate high performance when applied to tiles coming from the same geographic area as the dataset on which they have been trained, thus proving that they effectively learn the underlying distribution and features of that specific domain. However, quality clearly degrades in cross-dataset scenarios. Interestingly, the capability of the Adriatic-trained AE to reconstruct Irish data appears to be slightly superior to the one of the opposite combination. This discrepancy is likely attributable to the higher quality and potentially better-defined visual characteristics of the Irish images, which may offer a more robust structural signal, allowing the Adriatic model to extract a more informative representation of data despite the domain shift.

This behaviour is also reflected in the results of the tiles classification through MLPs. Indeed, classifiers trained on the Irish dataset achieve higher precision and recall than the ones trained on the Adriatic dataset, with an increase of

Table 3: Confusion matrices of MLP experiments computed on the 480 test tiles.

(a) AE Adriatic Sea, MLP Adriatic Sea				(b) AE Ireland, MLP Adriatic Sea			
G.T.	Prediction			G.T.	Prediction		
	BG	N	NN		BG	N	NN
BG	142.4	7	10.6	BG	111.2	21.2	27.6
N	5.4	106.2	48.4	N	38	85.4	36.6
NN	17.2	56	86.8	NN	60.4	60.4	39.2

(c) AE Adriatic Sea, MLP Ireland							(d) AE Ireland, MLP Ireland						
G.T.	Prediction						G.T.	Prediction					
	BG	P	sP	Gk	CL	CR		BG	P	sP	Gk	CL	CR
BG	153.4	15.8	18.6	0.4	1	0.8	BG	144.4	17.2	23.8	2.2	1.4	1
P	18	79.2	17	2	1.2	1.6	P	13	86.2	14.8	2.8	2.2	0
sP	36.4	27	51.8	0	1.2	0.6	sP	31.4	24.8	57.4	0.2	2.4	0.8
Gk	0.6	4	0	3	0	0.4	Gk	0.2	3.4	0	4.2	0	0.2
CL	14.8	8.6	6.2	0	1.4	0	CL	11.4	8.2	5.8	0	5.6	0
CR	7	2.8	3.6	0	0	1.6	CR	5.8	2.2	3.8	0.2	0	3

Table 4: Perceptron performances for *N. norvegicus*-related classes.

			Training		Validation		Test	
			<i>p</i>	<i>r</i>	<i>p</i>	<i>r</i>	<i>p</i>	<i>r</i>
(a)	Adriatic	Adriatic	0.703	0.679	0.625	0.605	0.630	0.664
(b)	Irish	Adriatic	0.549	0.571	0.531	0.551	0.513	0.534
(c)	Adriatic	Irish	0.798	0.757	0.759	0.718	0.769	0.754
(d)	Irish	Irish	0.821	0.841	0.755	0.765	0.754	0.794

All values are rounded to the third decimal digit.

12.4% in precision and 13% in recall on the respective test sets when considering models whose AE and MLP are trained on the same dataset (rows (a) and (d) of Table 4). The more robust feature representation in the encoding of Irish tiles influences the classification result also in cross-dataset cases (rows (b) and (c)). Nonetheless, within-dataset models show a more consistent classification effectiveness, thus highlighting the importance of area-specific models for each possible geographic location.

6 Conclusion

This work highlights the integration of machine learning techniques into *N. norvegicus* population monitoring as a vital step toward sustainable fisheries man-

agement. By automating burrow detection, we could reduce the subjectivity and manual labour inherent in current survey assessments, providing a more reliable foundation for density estimation.

However, our findings highlight that one of the main obstacles to large-scale deployment is dataset shift. As evidenced by differences in the model's performance across the two study areas, model accuracy is strongly influenced by site-specific factors, including turbidity, lighting and geomorphological diversity. These environmental discrepancies act as a barrier to generalisation and direct the research activities towards models that should not be treated as static tools but as an adaptive framework that requires fine-tuning tailored to the specific conditions of each surveyed region.

In the future, further analyses will be conducted, accounting for additional variations in the architectures used for both the AE and the MLP.

Acknowledgments. The authors would like to thank the IRBIM-CNR staff who contributed to collecting UWTV footage, in particular Andrea Belardinelli, Pierluigi Penna and Paolo Scarpini. The authors would also like to thank Giulio Del Corso (ISTI-CNR) for the inspirational conversations.

Funding. This work is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101000825 – NAUTILOS Project (<https://www.nautilus-h2020.eu>).

Data Availability. Due to strict regulations concerning both the preservation of fragile marine ecosystems and economic agreements concerning fishing, related sensitive information potentially contained in the Adriatic Sea dataset cannot be shared without the necessary preprocessing steps, e.g. data anonymisation. In the future, data and code will be released, provided that the mentioned procedures are applied.

Authors' contribution. Conceptualization: O.P., M.R., E.C., F.D., L.Z., M.M., G.P.; Data curation: E.C., F.D., L.Z., M.M.; Investigation: O.P., M.R., G.P.; Methodology: O.P., M.R., G.P.; Resources: O.P., M.R., E.C., F.D., L.Z., M.M., G.P.; Software: O.P., M.R., G.P.; Supervision: M.M., G.P.; Visualization: O.P.; Writing – original draft: O.P., M.R., E.C., F.D., L.Z., M.M., G.P.; Writing – review & editing: O.P., M.R., E.C., M.M., G.P. All authors have read and approved the final manuscript.

Disclosure of Interests. The authors of this work declare that they have no conflicts of interest.

References

1. Bell, M.C., Tuck, I., Dobby, H.: '*Nephrops* Species'. In: Lobsters: Biology, Management, Aquaculture and Fisheries. Ed. by B.F. Phillips. 2nd Edition. John Wiley & Sons, Ltd, 2013. Chap. 12, pp. 357–413. <https://doi.org/10.1002/9781118517444.ch12>.
2. Aguzzi, J. *et al.*: Advancing fishery-independent stock assessments for the Norway lobster (*Nephrops norvegicus*) with new monitoring technologies. *Front. Mar. Sci.* **9** (2022). <https://doi.org/10.3389/fmars.2022.969071>

3. Aguzzi, J. *et al.*: Burrow emergence rhythms of *Nephrops norvegicus* by UWTV and surveying biases. *Sci. Rep.* **11** (2021). <https://doi.org/10.1038/s41598-021-85240-3>
4. ICES, Working Group on *Nephrops* Surveys (WGNEPS; outputs from 2022 meeting). *ICES Sci. Rep.* 5:26, 125 pp. (2023). <https://doi.org/10.17895/ices.pub.22211161.v1>
5. ICES, Report of the Workshop on *Nephrops* Burrow Counting (WKNEPS). *ICES CM 2018/EOSG:25*, 44 pp. 2–5 October 2018, Aberdeen, UK (2019). <https://doi.org/10.17895/ices.pub.8180>
6. Papini, O. *et al.*: Deep Learning–Based Detection of *Nephrops norvegicus* Burrows. In: *Proceedings of the 2025 IEEE International Workshop on Metrology for the Sea*, pp. 39–44 (2025). <https://doi.org/10.1109/MetroSea66681.2025.11245690>
7. Dobby, H. *et al.*: ICES Survey Protocols – Manual for *Nephrops* Underwater TV Surveys, coordinated under ICES Working Group on *Nephrops* Surveys (WGNEPS). *ICES Tech. Mar. Environ. Sci.* **65** (2021). <https://doi.org/10.17895/ices.pub.8014>
8. Martinelli, M. *et al.*: Towed underwater television towards the quantification of Norway lobster, squat lobsters and sea pens in the Adriatic Sea. *Acta Adriat.* **54**(1), 3–12 (2013)
9. Martinelli, M. *et al.*: Changes in Abundance and Distribution of the Sea Pen, *Funiculina quadrangularis*, in the Central Adriatic Sea (Mediterranean Basin) in Response to Variations in Trawling Intensity. *Fishes* **8**(7) (2023). <https://doi.org/10.3390/fishes8070347>
10. Papini, O. *et al.*: Guidelines for the annotation of *Nephrops norvegicus* UWTV videos, (2025). <https://doi.org/10.5281/zenodo.14973160>
11. Melvin, É.: *Nephrops (Nephrops norvegicus) Burrow object detection simple training dataset from Irish Underwater TV surveys*. Zenodo, July 2024. <https://doi.org/10.5281/zenodo.13987957>.

Pre-Generation LLM Failure Prediction Under Distribution Shift

Yael Moros-Daval¹, Álvaro David Gómez Antón¹, Kexin Jiang-Chen¹, Cèsar Ferri¹, José Hernández-Orallo^{1,2}, and Fernando Martínez-Plumed¹

¹ VRAIN, Universitat Politècnica de València, València, Spain

² Leverhulme Centre for the Future of Intelligence, University of Cambridge
ymordav@inf.upv.es, {adgomant,kjiache}@etsinf.upv.es,
{cferri,jorallo,fmartinez}@dsic.upv.es

Abstract. Reliable deployment of LLMs requires anticipating failures when the model is applied to tasks or prompts that differ from those seen during development. We study *pre-generation failure prediction*, where an auxiliary *assessor model* estimates the risk of an incorrect answer from the prompt alone. We evaluate it across four LLMs and six reasoning benchmarks using a leave-one-benchmark-out protocol. Because the benchmarks span disjoint task families (arithmetic, word puzzles, geographic recall, multiple-choice science, geometry, logic) with almost no shared vocabulary or format, this is a near-worst-case transfer test where chance is the natural null; what matters is which representations exceed chance under shift, not their absolute level. Dense embeddings are strongest in-domain but lose most of their advantage under shift, whereas interpretable cognitive-demand features degrade less sharply. The best out-of-domain results combine the two. Benchmark- and model-level analyses reveal substantial heterogeneity. Robust failure prediction therefore cannot rely on semantic representations alone: task-structural features add complementary, shift-resistant signal for interpretable reliability assessment.

Keywords: LLM reliability · Assessor Models · Distribution Shift · ADeLe

1 Introduction

Large language models (LLMs) are increasingly embedded in systems where their outputs influence real decisions, making confidently fluent but incorrect answers a significant reliability concern. A natural question is therefore: *will the model get this prompt wrong?* If failure can be anticipated before generation, the system can abstain, defer to a human, or route the request to a stronger model, avoiding the cost and risk of producing an unreliable answer. This setting falls within the classical *reject option* framework [3,4,8], which we instantiate through a *pre-generation assessor*: a lightweight auxiliary model that estimates failure risk from the prompt alone [14,26].

Failure prediction is a supervised classification problem whose performance can degrade substantially when the deployment distribution differs from the training distribution, a phenomenon known as *dataset shift* [21]. This challenge is particularly relevant for LLM reliability, since assessors are most valuable precisely when deployed on tasks that differ from those used during development. The key question is therefore not only whether failures can be predicted, but whether the signals used to predict them transfer across task families. In this work, we study pre-generation failure prediction under benchmark shift and investigate which prompt representations remain informative when the assessor is applied to previously unseen tasks.

To study this question, we evaluate failure prediction across four LLMs and six reasoning benchmarks using a leave-one-benchmark-out protocol. In each experiment, the assessor is trained on five task families and evaluated on a previously unseen sixth, creating a controlled benchmark-shift setting that tests whether failure signals transfer beyond the tasks on which they were learned. Our contributions are:

1. **Pre-generation failure prediction under benchmark shift.** We study LLM failure prediction as a transfer problem, evaluating whether a pre-generation assessor trained on some task families can anticipate failures on previously unseen ones. To this end, we introduce a leave-one-benchmark-out protocol spanning four LLMs and six reasoning benchmarks.
2. **Characterising the shift.** We choose six maximally disjoint task families, so leave-one-benchmark-out induces a near-worst-case shift: train and test prompts share almost no vocabulary, format, or solution schema, combining near-total covariate shift with large prevalence differences. The transfer we measure is thus a conservative lower bound on milder, within-family deployment drift.
3. **The limits of embedding-based transfer.** Dense semantic embeddings achieve the strongest in-domain discrimination, but much of this advantage disappears under benchmark shift. Strong in-domain failure prediction therefore does not necessarily translate into transferable failure prediction.
4. **Cognitive-demand features are less sensitive to benchmark shift.** While cognitive-demand features achieve only modest out-of-domain discrimination on their own, their performance degrades less sharply than that of embeddings. When combined with embeddings, they consistently improve out-of-domain results, suggesting that they capture aspects of failure risk that are less benchmark-specific than purely semantic representations.
5. **Benchmark- and model-level analysis of failure predictability.** We find that transferability varies substantially across both task families and LLMs. Some benchmarks remain near chance regardless of representation, while others benefit strongly from cognitive-demand information; similarly, some models exhibit much more predictable failure patterns than others.

2 Related work

2.1 Selective prediction and failure assessment for LLMs

Selective prediction studies systems that may abstain when their predictions are likely to be unreliable, trading coverage for accuracy through a reject option [3,4]. In modern neural models, abstention is commonly implemented either as a post-hoc policy over a fixed predictor [8] or as an explicit selection mechanism trained jointly with the predictor [9]. Related work has also explored calibration [13], uncertainty estimation [7,18], and conformal prediction [27,1] as mechanisms for identifying predictions on which a model should abstain.

For LLMs, reliability assessment is complicated by free-form generation, semantic equivalence, and reasoning variability. Existing approaches estimate correctness either after generation, using confidence scores, self-evaluation, agreement across samples, or semantic uncertainty [16,20,17,5]; or before generation through assessor models and related predictive approaches [14,26,25,19]. Such methods support early abstention, escalation to stronger models, or routing in multi-model systems [22,15].

2.2 Dataset shift and domain generalisation

A central challenge in machine learning is that models are often deployed on data that differ from those seen during training, a phenomenon broadly referred to as *dataset shift* [21]. Under such shifts, predictors may exploit patterns that are highly informative in the source domain but fail to generalise to new environments. This problem has motivated extensive work on domain adaptation, domain generalisation, and robust learning, all of which seek representations that remain predictive when the data distribution changes [32,30].

Dataset shift is particularly relevant for LLM reliability assessment. An assessor trained on one collection of tasks may encounter prompts with different linguistic characteristics, reasoning requirements, and error rates when deployed on another. Consequently, strong in-domain performance does not necessarily imply reliable failure prediction on unseen tasks. Recent work has highlighted the sensitivity of uncertainty estimates, confidence scores, and selective prediction mechanisms to distribution shift [6,31], motivating evaluations that explicitly measure transfer beyond the training distribution. From this perspective, an assessor built on dense embeddings is prone to rely on *shortcut* features—surface regularities such as digit strings or proper nouns that correlate with failure within a task family but carry no signal once the family changes—mirroring the shortcut-learning and spurious-correlation phenomena documented in supervised models [10,2].

Our work studies this problem through benchmark shift: assessors are trained on a set of reasoning task families and evaluated on a previously unseen one. Rather than proposing a new adaptation method, we investigate which prompt representations are most resistant to this shift and whether interpretable task-level information degrades differently from semantic embeddings when transferred across domains.

3 Methodology

3.1 LLM failure as a classification problem

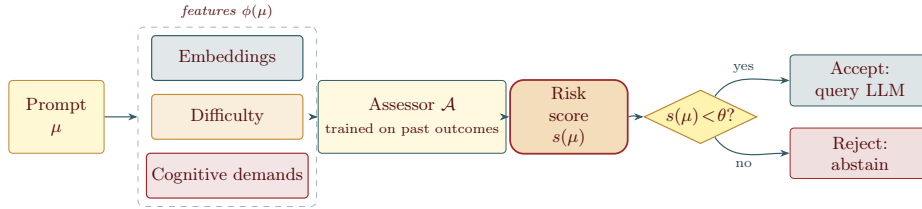


Fig. 1: Pre-generation failure assessment. Three descriptions of the prompt (embeddings, intrinsic difficulty, and ADeLe cognitive demands) feed a small trained assessor that outputs a risk score $s(\mu)$ before the LLM is run. Thresholding this score gives a reject rule: prompts predicted to be safe are sent to the LLM, while high-risk prompts are rejected, deferred, or routed elsewhere.

Let $\mathcal{M}$ denote an LLM and let $\mu_{i=1}^N$ be a benchmark of prompts. For each prompt, an external automatic procedure labels the model’s response as *correct* (C), *incorrect* (I), or *avoidance* (A), where the model explicitly declines to answer. We regard C and A as *safe* outcomes, since both a correct answer and an honest refusal are acceptable, whereas I is considered *unsafe*. A *pre-generation assessor* is a model $\mathcal{A}$ that predicts the risk associated with a prompt before the LLM is queried [14]. Specifically, it maps prompt features $\phi(\mu)$ to a risk score $s(\mu) \in [0, 1]$ without accessing the response that $\mathcal{M}$ would generate. Given a threshold θ , the resulting abstention policy allows $\mathcal{M}$ to answer when $s(\mu) < \theta$ and abstains otherwise. Prompts rejected by the policy are assigned the label R. Figure 1 summarises this pipeline.

3.2 Prompt feature families

The assessor input, $\phi(\mu)$, is a feature-based representation of the prompt. We consider three feature families and evaluate seven feature views: each family individually, each pairwise combination, and the combination of all three families.

Embeddings (Emb). A dense vector representation of the prompt produced by a strong text encoder [23]. This representation captures both semantic content and surface-level characteristics, such as vocabulary and phrasing. Embeddings provide a rich and highly informative description of the prompt, but they may also encode benchmark-specific patterns that do not necessarily transfer across tasks, making them potentially sensitive to covariate shift.

Intrinsic difficulty (Diff). A scalar feature intended to capture the intrinsic difficulty of a prompt independently of any particular LLM. The difficulty proxy is task-specific and derived from properties of the problem itself, such as the number of carry operations (addition), word length (anagram), inverse city popularity (locality), estimated human difficulty (science), or problem size (e.g., grid or polygon size in structured tasks). This representation is transparent and readily interpretable, but necessarily coarse: a single scalar can indicate that a prompt is difficult without explaining the source of that difficulty. Moreover, because the proxy is defined separately for each benchmark, its interpretation may not transfer consistently across tasks.

Cognitive demands (ADeLe). ADeLe represents a prompt through an interpretable profile of the abilities required to solve it [33]. Rather than describing the content or surface form of a prompt, it characterises the extent to which different cognitive abilities and knowledge types are required for successful completion.

Each prompt is assigned a score from 0 to 5 on 18 dimensions using explicit rubric-based criteria (Table 1). These dimensions comprise 11 cognitive dimensions grouped into seven capability families, 5 knowledge dimensions, and 2 additional dimensions capturing *atypicality* (how unusual an instance is) and *volume* (the amount of information to process). The scores are automatically assigned by an LLM annotator using the ADeLe rubrics, yielding an 18-dimensional feature vector for each prompt.

ADeLe has two properties that make it attractive for transfer. First, it is *interpretable*: the representation identifies which abilities contribute to the difficulty of a prompt (e.g., high logical reasoning and high atypicality), making risk predictions auditable along meaningful dimensions. Second, it is *task-structural and model-agnostic*: it describes the abilities required to solve a task rather than its surface form. Consequently, the meaning of a given demand level is intended to remain stable across task families. Whereas two prompts with similar difficulty may require very different skills, ADeLe explicitly captures this structure, which we hypothesise is more likely to transfer across tasks.

3.3 Evaluation measures and protocols

Consider a labelled evaluation set of N prompts with outcomes $y(\mu) \in \{\mathbf{C}, \mathbf{l}, \mathbf{A}\}$. Let $\mathcal{S} = \{\mu : y(\mu) \in \{\mathbf{C}, \mathbf{A}\}\}$ denote the set of *safe* prompts and $\mathcal{U} = \{\mu : y(\mu) = \mathbf{l}\}$ the set of *unsafe* prompts. The true error prevalence is therefore $p = |\mathcal{U}|/N$.

The assessor assigns each prompt a risk score $s(\mu) \in [0, 1]$, where larger values indicate higher estimated failure risk. Given a threshold θ , the policy answers when $s(\mu) < \theta$ and abstains otherwise. This induces a rejection rate $r(\theta) = \frac{1}{N} \sum_{\mu} \mathbf{1}[s(\mu) \geq \theta]$, and coverage $\text{Cov} = 1 - r$.

We evaluate the assessor as a classifier that distinguishes safe from unsafe prompts using Eqs. (1)–(3).

Table 1: The 18 ADeLe demand dimensions used as features, with their short codes and rubric descriptions [33]. Each prompt is scored 0–5 on every dimension by an LLM annotator following explicit per-level rubrics. In the cognitive block, shared code prefixes mark the seven capability families (**CE**: comprehension/expression; **MC**: metacognition/critical thinking; **QL**: quantitative/logical reasoning).

Code	Dimension	Demand description
<i>Cognitive demands (11)</i>		
AS	Attention & Scan	Focus on or locate specific elements within a stream of information or environment while solving the task.
CEc	Verbal Comprehension	Understand text, stories, or the semantic content of other representations of ideas across formats and modalities.
CEe	Verbal Expression	Generate and articulate ideas, stories, or semantic content across formats and modalities.
CL	Conceptualisation, Learning & Abstraction	Build new concepts, reason inductively and by analogy, map relations between domains, and abstract from concrete examples.
MCr	Identifying Relevant Information	Recognise which information does or does not help solve the task, and how that recognition unfolds while working toward a solution.
MCt	Critical Thinking Processes	Monitor and regulate multiple thought processes, from simple recall to high-level critical thinking.
MCu	Calibrating Knowns & Unknowns	Recognise the boundaries of one’s own knowledge: what one knows, does not know, or is uncertain about.
MS	Mind Modelling & Social Cognition	Model other agents’ beliefs, desires, intentions, and emotions, and how they interact to shape behaviour.
QLl	Logical Reasoning	Match and apply rules, procedures, or algorithms to premises to derive conclusions and make decisions.
QLq	Quantitative Reasoning	Work with and reason about quantities, numbers, and numerical relationships.
SNs	Spatio-physical Reasoning	Understand spatial relationships between objects and predict physical interactions.
<i>Knowledge demands (5)</i>		
KNa	Applied Sciences	Knowledge or conceptual understanding in applied sciences (medicine, law, education, business, engineering, ...).
KNc	Customary Everyday Knowledge	Information most people in a society acquire through daily life, social interaction, and media.
KNf	Formal Sciences	Knowledge or conceptual understanding in formal sciences (mathematics, logic, computer science, statistics).
KNn	Natural Sciences	Knowledge or conceptual understanding in natural sciences (physics, chemistry, biology, astronomy, ...).
KNs	Social Sciences & Humanities	Knowledge or conceptual understanding in social sciences and humanities (history, psychology, literature, philosophy, ...).
<i>Extraneous demands (2)</i>		
AT	Atypicality	How uncommon the task is: how unlikely it is that the instance has appeared in sources such as the internet, textbooks, or tests.
VO	Volume	Proportional to the logarithm of the time a fully competent human needs to read and complete the task under ideal conditions.

$$\text{S-Acc}(\theta) = \frac{1}{|S|} \sum_{\mu \in S} \mathbf{1}[s(\mu) < \theta], \quad \text{U-Rej}(\theta) = \frac{1}{|\mathcal{U}|} \sum_{\mu \in \mathcal{U}} \mathbf{1}[s(\mu) \geq \theta], \quad (1)$$

$$\text{FOC-AUC} = \Pr_{u \in \mathcal{U}, v \in S} [s(u) > s(v)], \quad (2)$$

$$\text{Ret} = \frac{\text{FOC-AUC}^{\text{OOD}} - \frac{1}{2}}{\text{FOC-AUC}^{\text{ID}} - \frac{1}{2}}. \quad (3)$$

Operating-point rates. At a fixed threshold θ , S-Acc is the fraction of safe prompts that remain answerable under the policy, while U-Rej is the fraction of unsafe prompts on which the policy abstains (see eqs. (1)). Equivalently, these are the recalls of the safe and unsafe classes, respectively. Both metrics lie in $[0, 1]$, and increasing one typically decreases the other: a stricter threshold raises U-Rej by catching more errors, but lowers S-Acc by refusing more safe responses.

Threshold-free discrimination. Varying θ from accept-all to reject-all traces the *filter operating characteristic* (FOC), which plots S-Acc against U-Rej. Its area, FOC-AUC (see eq. (2)), summarises discrimination independently of any particular operating point. By the Mann–Whitney identity, FOC-AUC equals the probability that a randomly chosen unsafe prompt receives a higher risk score than a randomly chosen safe prompt, with ties counted as $\frac{1}{2}$. Equivalently, it is the AUROC of the unsafe-versus-safe ranking. A value of 1 indicates perfect separation, while $\frac{1}{2}$ corresponds to chance performance. We use FOC-AUC as our primary classification metric. In practice, θ is an application-specific design choice that reflects the desired trade-off between answering more prompts and avoiding unsafe responses.

Transfer. Retention measures how much discrimination survives a distribution shift. It compares out-of-domain (OOD) and in-domain (ID) performance after subtracting the chance baseline of $\frac{1}{2}$ (see eq. (3)). Thus, $\text{Ret} = 1$ (100%) indicates no loss in discrimination, $\text{Ret} = 0$ corresponds to a collapse to chance performance, and $\text{Ret} < 0$ indicates worse-than-chance transfer. Subtracting the chance floor makes retention comparable across tasks with different in-domain difficulty levels, which a raw ratio of FOC-AUCs would not.

Protocols. We use two regimes. *In-domain* (ID): train and test on disjoint splits of the *same* benchmark, so train and test are identically distributed. *Cross-domain* (OOD): *leave-one-benchmark-out*, train on five benchmarks and test on the held-out sixth, so test prompts come from an unseen task family. The OOD protocol is our controlled dataset shift. Unless stated otherwise, FOC-AUC is reported as the average across the four LLMs, both per held-out benchmark and pooled.

4 Experimental setup

Models and benchmarks. We use four widely deployed LLMs, two open-weight and two proprietary: LLaMA-3.3-70B [12], Qwen-2.5-72B [29], GPT-5.2 [24], and Gemini-3-Pro [11], queried at temperature 0. We evaluate on six reasoning benchmarks, each with a per-prompt intrinsic difficulty proxy (Table 2): four from a reliability suite (arithmetic, word puzzles, geographic knowledge, multiple-choice science) [34] and two families of programmatically generated structured-reasoning problems (geometry, logic puzzles) [28]. Answers are labelled C/I/A by task-specific automatic procedures verified against human checks. To make difficulty comparable across the two benchmark sources in the OOD setting,

difficulty is mapped onto a common scale by a monotone calibration fitted on a reference model’s success rates.

Table 2: The six reasoning benchmarks, their task types, and the intrinsic prompt difficulty proxy $\delta(\mu)$. Difficulty is computed from the prompt alone and used as the Diff feature. Benchmarks 1–4 are from ReliabilityBench [34]; benchmarks 5–6 are structured-reasoning tasks from OMEGA [28].

Benchmark	Task (what the prompt asks)	Difficulty proxy $\delta(\mu)$
Addition [34]	Add two integers of up to 100 digits.	δ = number of carry operations in the sum $a+b$.
Anagram [34]	Unscramble a letter string into an English word w .	$\delta = w $, the word length (permutations grow as $ w !$).
Locality [34]	Name the most populous city within a fixed radius of a given city.	δ = inverse popularity (population rank) of the answer city; rarer cities are harder.
Science [34]	Answer a multiple-choice science question (school to graduate level).	δ = the item’s annotated human-difficulty level.
Geometry [28]	Counting/colouring problems on a regular polygon.	$\delta = n$, the polygon’s number of sides ($n(n-3)/2$ diagonals).
Logic Puzzles [28]	Grid paths, “zebra”-style constraint puzzles, chip placement.	δ = number of constrained entities (grid size).

5 Results

5.1 Characterising the shift

Before evaluating transfer, we first characterise the distribution shifts induced by the leave-one-benchmark-out protocol. Since the assessor’s target is the unsafe outcome $\mathbf{l}$, the relevant class prior is the prevalence of incorrect responses, $p = P(\mathbf{l})$.

Figure 2 reports this prevalence for every (model, benchmark) pair, corresponding to the LLM’s baseline error rate before any abstention ($r = 0$). The resulting variation is substantial: error rates range from 0.001 (**Gemini-3-Pro** on **Anagram**) to 0.86 (**Qwen-2.5-72B** on **Anagram**), with a mean of 0.44. The variation is large both across models and across tasks. Within a single benchmark, error prevalence can span nearly the entire unit interval (e.g., 0.001–0.86 on **Anagram**), while within a single model it can vary dramatically across benchmarks (e.g., **LLaMA** ranges from 0.19 on **Science** to 0.79 on **Logic Puzzles**).

The leave-one-benchmark-out protocol induces substantial distribution shift. Because the assessor for a held-out benchmark is trained on the other five, the prior it encounters during training is approximately their average, whereas the target prior is that of the held-out task. Treating benchmarks as equal-sized, the resulting prior-probability shift $|\Delta p| = |p_{\text{test}} - \bar{p}_{\text{train}}|$ averages 0.15 across the 24 (model, benchmark) cells and reaches 0.48 (Table 3).

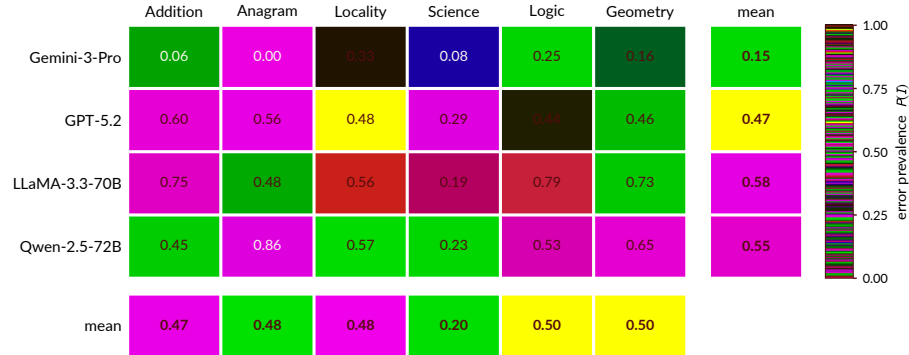


Fig. 2: Base error prevalence $p = P(l)$ per model and benchmark (the base error rate the assessor must anticipate). It spans 0.001–0.86, so leaving out one benchmark moves the class prior far from the training mix: this is severe *prior-probability (label) shift*.

Table 3: Per-benchmark prevalence and the prior-probability shift induced by leave-one-benchmark-out. $\bar{p}$: error prevalence pooled over the four LLMs. $|\Delta p|$: mean over LLMs of $|p_{\text{test}} - \bar{p}_{\text{train}}|$ when that benchmark is held out.

	Addition	Anagram	Locality	Science	Logic	Geometry	mean
$\bar{p}$ (prevalence)	0.47	0.48	0.48	0.20	0.50	0.50	0.44
$ \Delta p $ (shift)	0.15	0.19	0.07	0.29	0.11	0.09	0.15

The largest shifts occur for **Science**, whose low error rates ($\bar{p} \approx 0.20$) differ substantially from those of the remaining benchmarks ($\bar{p} \approx 0.47$ – 0.50), yielding an average shift of 0.29. This label shift is accompanied by strong covariate shift: prompt distributions differ sharply across task families, ranging from digit strings and city names to polygons and logical puzzles. The leave-one-benchmark-out protocol, therefore, constitutes a genuinely challenging dataset-shift setting rather than a mild train–test shift. Because the six families share essentially no surface vocabulary, input format, or solution procedure, the covariate supports of the training mix and the held-out task are close to disjoint. Under such near-total covariate shift, chance-level transfer is the appropriate null expectation for any prompt representation, and the substantive question is not whether out-of-domain discrimination is high in absolute terms (few representations could be) but which representation exceeds chance at all, and why.

5.2 Classification under shift: which features transfer?

We now ask which prompt feature lets the assessor *classify* unsafe prompts across this shift, using the error-risk assessor $\mathcal{A}_{\text{ER}}$. Table 4 gives, per feature view, in-domain and cross-domain FOC-AUC, the drop, and retention; Table 5

breaks the cross-domain numbers down by held-out benchmark; Fig. 3 visualises the trade-off.

Because the leave-one-benchmark-out setting places train and test on near-disjoint task supports (Section 5.1), an out-of-domain FOC-AUC near 0.5 is the expected null rather than a sign of method failure. We therefore read Table 4 primarily through *retention*, i.e., the fraction of above-chance discrimination that survives the shift, and through the *ordering* of feature views, rather than through absolute OOD levels.

Table 4: Average FOC-AUC (over four LLMs) for the error-risk assessor $\mathcal{A}_{\text{ER}}$ by feature view. **ID**: in-domain; **OOD**: leave-one-benchmark-out; **Δ** : drop; **Ret.**: above-chance discrimination retained. 0.5 is random. Best per block in bold.

Feature view	ID $\uparrow$	OOD $\uparrow$	Δ $\downarrow$	Ret. $\uparrow$
<i>Interpretable only (describe the task, not its words)</i>				
Difficulty (Diff)	0.613	0.495	0.118	−4.5%
Cognitive demands (ADeLe)	0.662	0.536	0.126	22.3%
Diff + ADeLe	0.696	0.522	0.174	11.0%
<i>Using embeddings (describe meaning and surface form)</i>				
Embeddings (Emb)	0.738	0.535	0.203	14.7%
Emb + Diff	0.746	0.528	0.217	11.5%
Emb + ADeLe	0.759	0.567	0.192	25.9%
Emb + Diff + ADeLe	0.769	0.569	0.201	25.5%

Embeddings dominate in-domain, but their advantage shrinks under shift. Embedding-based views achieve the strongest in-domain performance, reaching FOC-AUC values around 0.75 compared with 0.61–0.70 for interpretable-only views (Table 4). Under leave-one-benchmark-out evaluation, however, this advantage largely disappears. While embedding-based models attain the highest in-domain discrimination, their performance drops substantially under shift, yielding OOD scores comparable to those of cognitive-demand features. For example, embeddings alone achieve 0.535 OOD FOC-AUC, essentially matching ADeLe (0.536) despite a much larger in-domain advantage. This pattern suggests that embeddings capture highly predictive information that is benchmark-dependent, whereas cognitive-demand features encode weaker but more stable signals that transfer across tasks.

Cognitive-demand features provide complementary signal under shift. The leave-one-benchmark-out results show that failure prediction remains challenging under benchmark shift. Even the strongest feature views achieve only modest OOD discrimination (FOC-AUC $\approx$ 0.57), and several benchmark-feature combinations remain close to chance. Within this difficult setting, however, ADeLe features

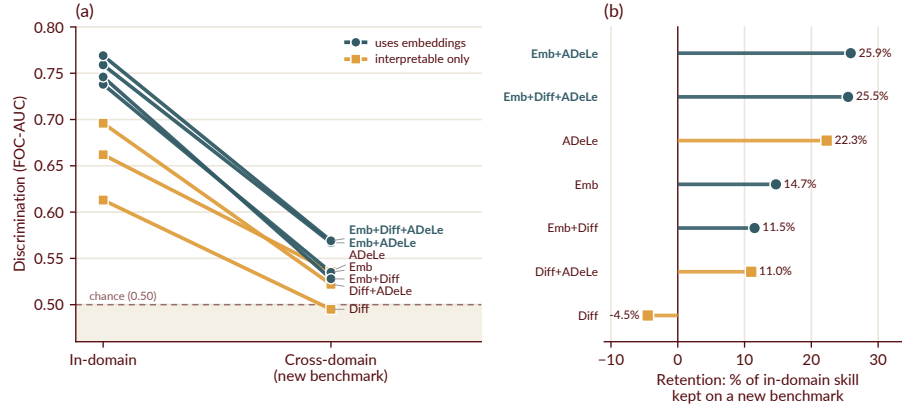


Fig. 3: Classification under shift, error-risk assessor. **(a)** Each line runs from a feature view’s in-domain to its cross-domain FOC-AUC: embedding views (teal) start highest and fall furthest; interpretable-only views (gold) start lower and fall less. **(b)** Above-chance discrimination retained under shift: adding ADeLe raises retention wherever applied; Emb+ADeLe keeps the most.

provide a consistent benefit. Adding cognitive-demand information improves embedding-based models, increasing OOD FOC-AUC from 0.535 to 0.567 for Emb→Emb+ADeLe and from 0.528 to 0.569 for Emb+Diff→Emb+Diff+ADeLe (Table 4). Among interpretable feature views, ADeLe alone achieves the strongest OOD performance, matching or slightly exceeding embeddings despite relying only on a compact, human-interpretable task description. The strongest transfer results overall are obtained by the ADeLe-bearing embedding models, Emb+ADeLe and Emb+Diff+ADeLe. Taken together, these findings suggest that cognitive-demand features capture aspects of failure risk that complement the information encoded by semantic embeddings.

Transfer behaviour varies substantially across benchmarks. The benchmark-level analysis (Table 5) reveals considerable heterogeneity in transfer performance. No feature view is best across all held-out benchmarks, and some tasks remain difficult regardless of representation. For example, performance on Locality is close to chance for all feature views, suggesting limited transferability of failure patterns from other domains. In contrast, other benchmarks show substantial differences between representations. On Anagram, embeddings alone perform below chance (0.467), whereas ADeLe-based models exceed 0.60. Geometry and Logic achieve their strongest results with embedding–ADeLe combinations. These patterns indicate that benchmark shift is not a single phenomenon: different task families preserve different aspects of the failure signal, and no representation is uniformly robust across all domains.

Table 5: Cross-domain (leave-one-benchmark-out) FOC-AUC of the error-risk assessor $\mathcal{A}_{\text{ER}}$ per held-out benchmark and feature view (average over LLMs). Best per column in bold. Embedding-bearing views are strongest in-domain (Table 4) yet here hover near chance on several tasks; demand-bearing views transfer more evenly. The final column reports the average over held-out benchmarks.

Feature view	Addition	Anagram	Locality	Science	Logic	Geometry	avg.
Diff	0.483	0.396	0.504	0.488	0.551	0.547	0.495
ADeLe	0.507	0.604	0.504	0.575	0.530	0.513	0.539
Diff+ADeLe	0.458	0.512	0.507	0.588	0.531	0.531	0.521
Emb	0.523	0.467	0.501	0.543	0.579	0.600	0.535
Emb+Diff	0.551	0.429	0.485	0.533	0.595	0.575	0.528
Emb+Diff+ADeLe	0.540	0.611	0.483	0.596	0.608	0.585	0.570
Emb+ADeLe	0.512	0.612	0.499	0.585	0.602	0.605	0.569

Table 6: Average FOC-AUC for the error-risk assessor $\mathcal{A}_{\text{ER}}$ per LLM and distribution.

Model	ID $\uparrow$	OOD $\uparrow$
gemini-3-pro-preview	0.680	0.511
GPT5.2	0.722	0.557
llama3v3chat-70B	0.714	0.589
Qwen2.5_72B	0.729	0.484

Failure predictability depends strongly on the target model. Variation across LLMs is larger than variation across feature views, as we see in Table 6. While average OOD discrimination ranges from 0.535 to 0.570 across feature representations, it ranges from 0.484 to 0.589 across models. The assessor is therefore substantially more effective for some LLMs than others. In particular, **Llama3v3-Chat-70B** achieves the strongest transfer performance (0.589 OOD FOC-AUC), whereas **Qwen2.5-72B** remains close to chance overall (0.484). This suggests that failure behaviour is not equally predictable across models: some LLMs exhibit error patterns that generalise across benchmarks, while others appear substantially harder to anticipate from prompt-level features alone.

Benchmark transfer is model-specific. The benchmark-level results reveal substantial interactions between task family and model (Table 7). **Llama3v3Chat-70B** achieves strong discrimination on Logic (0.681) and Geometry (0.663), whereas Gemini performs near chance on Anagram (0.222) despite competitive results on Science (0.615). **Qwen2.5-72B** remains close to chance on most held-out benchmarks. These patterns indicate that transferability is not solely a property of the benchmark or the feature representation. Instead, the structure of model failures appears to differ across LLMs, with some models exhibiting more stable and predictable error patterns than others.

Table 7: Average OOD FOC-AUC for the error-risk assessor $\mathcal{A}_{\text{ER}}$ per LLM and held-out benchmark.

Model	Addition	Anagram	Locality	Science	Logic	Geometry
Gemini-3-Pro	0.462	0.222	0.501	0.615	0.579	0.519
GPT-5.2	0.567	0.582	0.496	0.556	0.570	0.569
LLaMA-3.3-70B	0.549	0.548	0.504	0.588	0.681	0.663
Qwen-2.5-72B	0.464	0.516	0.489	0.474	0.453	0.510

6 Discussion

Representation governs transferability. For pre-generation LLM failure prediction, the central challenge is not achieving high in-domain discrimination but maintaining it under benchmark shift. Embedding-based representations are strongest in-domain, yet much of this advantage disappears in leave-one-benchmark-out evaluation. Cognitive-demand features transfer more robustly, achieving OOD performance comparable to embeddings despite relying on a compact, interpretable description of the task. However, this robustness should not be overstated: ADeLe alone reaches only modest discrimination (FOC-AUC ≈ 0.54), and even the strongest feature combinations remain around 0.57 OOD FOC-AUC. Failure prediction therefore remains difficult under benchmark shift. The main contribution of cognitive-demand features is not to solve the transfer problem, but to provide complementary information that consistently improves embedding-based assessors. The strongest OOD results are obtained by Emb+ADeLe and Emb+Diff+ADeLe, suggesting that robust failure prediction benefits from combining semantic and task-structural information rather than relying on either alone. Because our leave-one-benchmark-out suite is deliberately adversarial, these OOD figures should be read as a lower bound: the differential robustness of task-structural features, rather than their absolute level, is the property most likely to carry over to the milder shifts seen in practice.

Practical guidance. Our results suggest different feature choices for different deployment scenarios. When the deployment distribution closely matches the training distribution and labelled data are available, embedding-rich assessors provide the strongest per-instance predictions. Under task drift, however, their advantage shrinks considerably. In such settings, augmenting embeddings with cognitive-demand features yields the most reliable transfer performance, while also providing an interpretable description of why a prompt may be risky. At the same time, practitioners should calibrate expectations: even the strongest feature views achieve only modest OOD discrimination (FOC-AUC ≈ 0.57), indicating that benchmark-shifted failure prediction remains a difficult problem. Rather than eliminating uncertainty, cognitive-demand features provide a small but consistent improvement in robustness across unseen tasks.

Limitations. Several limitations should be considered. Difficulty proxies are benchmark-specific and coarse, particularly for the structured tasks where difficulty is represented by a small number of discrete levels. Our cross-domain protocol leaves out one benchmark at a time from a suite of deliberately disjoint task families that share almost no vocabulary, format, or solution schema. This makes leave-one-benchmark-out a particularly stringent transfer setting: the assessor is trained and tested on different task families with largely non-overlapping prompt distributions, rather than the more gradual within-task drift commonly encountered in deployment. The transfer we report should therefore be read as a conservative lower bound: under more realistic, related-domain shift we would expect more in-domain signal (including part of the embedding advantage) to survive. The flip side is that this suite cannot characterise that milder regime.

Transfer performance also varies across benchmarks and models. Some benchmark-model combinations achieve useful discrimination, whereas others remain near chance, suggesting that the predictability of LLM failures is itself model-dependent. Models with more stable relationships between prompt characteristics and failure appear to admit more transferable assessors, although the reasons remain an open question. All results rely on LLM-generated cognitive-demand annotations; future work should explore alternative annotation procedures and broader benchmark collections.

7 Conclusion

We studied pre-generation LLM failure prediction under benchmark shift using task difficulty, cognitive-demand profiles, and semantic embeddings. A leave-one-benchmark-out evaluation revealed that transfer remains challenging: feature views that achieve strong in-domain discrimination often degrade substantially when applied to unseen task families. In particular, embeddings provide the strongest in-domain performance but lose much of their advantage under shift.

Our results show that cognitive-demand features offer a complementary and interpretable source of information about failure risk. Although ADeLe alone achieves only modest OOD discrimination, it consistently improves embedding-based assessors and contributes to the strongest transfer performance observed in our experiments. The benchmark-level analysis further reveals that transferability is highly heterogeneous, varying across both task families and target models.

Overall, our findings suggest that robust failure prediction cannot rely solely on semantic representations of prompts. Information about what a task requires provides additional signal that remains useful under distribution shift. While current performance remains far from sufficient for reliable deployment, combining semantic and task-structural representations appears to be a promising direction for more transferable and interpretable LLM reliability assessment.

References

1. Angelopoulos, A.N., Bates, S.: A gentle introduction to conformal prediction and distribution-free uncertainty quantification. arXiv preprint arXiv:2107.07511 (2021)
2. Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint arXiv:1907.02893 (2019)
3. Chow, C.K.: On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory* **16**(1), 41–46 (1970)
4. El-Yaniv, R., Wiener, Y.: On the foundations of noise-free selective classification. *Journal of Machine Learning Research* **11**(53), 1605–1641 (2010)
5. Farquhar, S., Kossen, J., Kuhn, L., Gal, Y.: Detecting hallucinations in large language models using semantic entropy. *Nature* **630**(8017), 625–630 (2024)
6. Fisch, A., Jaakkola, T., Barzilay, R.: Calibrated selective classification (2022)
7. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. *Proceedings of Machine Learning Research*, vol. 48, pp. 1050–1059 (2016)
8. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. arXiv preprint arXiv:1705.08500 (2017)
9. Geifman, Y., El-Yaniv, R.: Selectivenet: A deep neural network with an integrated reject option. In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. *Proceedings of Machine Learning Research*, vol. 97, pp. 2151–2159 (2019)
10. Geirhos, R., Jacobsen, J.H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., Wichmann, F.A.: Shortcut learning in deep neural networks. *Nature Machine Intelligence* **2**(11), 665–673 (2020)
11. Google DeepMind: Gemini 3 pro. <https://deepmind.google/models/gemini/pro/> (2025)
12. Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al.: The llama 3 herd of models. arXiv preprint arXiv:2407.21783 (2024)
13. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. *Proceedings of Machine Learning Research*, vol. 70, pp. 1321–1330 (2017)
14. Hernández-Orallo, J., Schellaert, W., Martínez-Plumed, F.: Training on the test set: Mapping the system-problem space in ai. In: *Proceedings of the AAAI conference on artificial intelligence*. vol. 36, pp. 12256–12261 (2022)
15. Hu, Q.J., Bieker, J., Li, X., Jiang, N., Keigwin, B., Ranganath, G., Keutzer, K., Upadhyay, S.K.: Routerbench: A benchmark for multi-llm routing system. arXiv preprint arXiv:2403.12031 (2024)
16. Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., et al.: Language models (mostly) know what they know (2022)
17. Kuhn, L., Gal, Y., Farquhar, S.: Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In: *International Conference on Learning Representations (ICLR)* (2023)
18. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2017)

19. Lugoloobi, W., Foster, T., Bankes, W., Russell, C.: Llms encode their failures: Predicting success from pre-generation activations. arXiv preprint arXiv:2602.09924 (2026)
20. Manakul, P., Liusie, A., Gales, M.J.F.: Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 9004–9017 (2023)
21. Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. *Pattern Recognition* **45**(1), 521–530 (2012)
22. Ong, I., Almahairi, A., Wu, V., Chiang, W.L., Wu, T., Gonzalez, J.E., Kadous, M.W., Stoica, I.: Routellm: Learning to route llms with preference data. arXiv preprint arXiv:2406.18665 (2024)
23. OpenAI: New embedding models and api updates: text-embedding-3-large. <https://platform.openai.com/docs/models/text-embedding-3-large> (2024)
24. OpenAI: Introducing gpt-5.2. <https://openai.com/es-ES/index/introducing-gpt-5-2/> (2025)
25. Pacchiardi, L., Voudouris, K., Slater, B., Martínez-Plumed, F., Hernández-Orallo, J., Zhou, L., Schellaert, W.: Predictaboard: Benchmarking llm score predictability. In: Findings of the Association for Computational Linguistics: ACL 2025. pp. 15245–15266 (2025)
26. Schellaert, W., Martínez-Plumed, F., Hernández-Orallo, J.: Analysing the predictability of language model performance. *ACM Transactions on Intelligent Systems and Technology* **16**(2), 1–26 (2025)
27. Shafer, G., Vovk, V.: A tutorial on conformal prediction. *Journal of Machine Learning Research* **9**, 371–421 (2008)
28. Sun, Y., Hu, S., Zhou, G., Zheng, K., Hajishirzi, H., Dziri, N., Song, D.: Omega: Can llms reason outside the box in math? evaluating exploratory, compositional, and transformative generalization. arXiv preprint arXiv:2506.18880 (2025)
29. Team, Q.: Qwen2.5: A party of foundation models (September 2024), <https://qwenlm.github.io/blog/qwen2.5/>
30. Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W., Yu, P.S.: Generalizing to unseen domains: A survey on domain generalization. *IEEE transactions on knowledge and data engineering* **35**(8), 8052–8072 (2022)
31. Wenzel, F., Dittadi, A., Gehler, P., Simon-Gabriel, C.J., Horn, M., Zietlow, D., Kernert, D., Russell, C., Brox, T., Schiele, B., et al.: Assaying out-of-distribution generalization in transfer learning. *Advances in Neural Information Processing Systems* **35**, 7181–7198 (2022)
32. Zhou, K., Liu, Z., Qiao, Y., Xiang, T., Loy, C.C.: Domain generalization: A survey. *IEEE transactions on pattern analysis and machine intelligence* **45**(4), 4396–4415 (2022)
33. Zhou, L., Pacchiardi, L., Martínez-Plumed, F., Collins, K.M., Moros-Daval, Y., Zhang, S., Zhao, Q., Huang, Y., Sun, L., Prunty, J.E., et al.: General scales unlock ai evaluation with explanatory and predictive power. arXiv preprint arXiv:2503.06378 (2025)
34. Zhou, L., Schellaert, W., Martínez-Plumed, F., Moros-Daval, Y., Ferri, C., Hernández-Orallo, J.: Larger and more instructable language models become less reliable. *Nature* **634**(8032), 61–68 (2024)

On Binary Quantification Calibration

Isabella Caroline Sachini Lorena¹, André Maletzke², and Willian Zalewski^{1,3}

Western Paraná State University, Foz do Iguaçu, Brazil

{isabella.lorena, andre.maletzke}@unioeste.br

Federal University for Latin American Integration, Foz do Iguaçu, Brazil

willian.zalewski@unila.edu.br

Abstract. Quantification is a supervised learning task whose goal is to estimate the class distribution of unlabeled samples. Many quantification methods rely on a classifier as an intermediate component and therefore inherit its biases, especially under label shift, where class prevalences differ between training and test data. Calibration, the requirement that a predicted quantity match its true value on average, has been studied extensively for classifiers, but the literature has treated it almost exclusively as a property of the classifier rather than of the quantifier itself. In this work, we extend the classical calibration condition from classifiers to quantifiers, and show that existing correction methods such as Adjusted Classify and Count (ACC) and its probabilistic counterpart PACC already instantiate this notion, though only through a fixed parametric correction tied to a specific classifier error-rate model. Building on this formalization, we propose *QuantCal* (QCal), a post-hoc calibration method that learns a non-parametric correction directly from a quantifier’s own prevalence estimates, without assuming a fixed functional form and without requiring a separate derivation for each quantifier. We evaluate QCal on 106 binary datasets across eight base quantifiers. The results show that QCal significantly reduces the mean absolute error of every base quantifier considered.

Keywords: Quantification · Calibration · Label shift

1 Introduction

Calibration is a general requirement for any predictor that outputs a quantity meant to be interpreted probabilistically: the predicted value should match the true value it purports to estimate, on average, across all instances for which that same value is predicted. This notion, originating in the evaluation of probabilistic weather forecasts [18,19,20], has been studied extensively for classifiers [22], where it requires that, among all instances assigned a given confidence score, the fraction correctly classified matches that score. A wide range of post-hoc techniques has been developed to enforce this property when it fails to hold, correcting a classifier’s scores after training without altering the underlying model [26].

Quantification, or counting, is a supervised learning task whose goal is to estimate the class distribution of an unlabeled sample, rather than the class of

each individual instance [6,8]. Most quantification methods rely on a classifier as an intermediate step, either through its hard predictions or its posterior probabilities. The canonical baseline, *Classify and Count* (CC) [6], simply counts the classifier’s predictions, but this strategy is known to produce systematically biased estimates whenever the class distribution shifts between training and test data [8]. This limitation motivated a large body of work specifically designed for quantification, establishing it as a task in its own right rather than a by-product of classification [5].

Throughout this literature, calibration is framed almost exclusively as a property of the classifier. Yet methods such as *Adjusted Classify and Count* (ACC) [7] and PACC [3] already correct the quantifier’s own prevalence estimate, using a linear adjustment derived from the classifier’s estimated error rates; this correction, however, has not been framed as an instance of the same calibration principle introduced above, nor generalized beyond its fixed affine form. To the best of our knowledge, no prior work formalizes what it means for a binary quantifier itself to be calibrated. This is the gap addressed in this paper: we extend the classical calibration condition from classifiers to quantifiers, and propose *QuantCal* (QCal), a post-hoc method that learns to correct a binary quantifier’s prevalence estimates directly, without assuming a parametric form for the correction and without requiring a separate derivation for each quantifier. The main contributions of this work are: (i) a formalization of quantifier calibration as an extension of Murphy’s general calibration condition, under which ACC and PACC are shown to be parametric instances of a broader framework; (ii) QCal, a non-parametric instantiation of this framework that estimates the correction function directly from data, and applies without modification to any binary quantifier; and (iii) an empirical evaluation of QCal across 106 binary datasets and eight base quantifiers, assessing its effectiveness as a general-purpose calibration method.

2 Background

2.1 Quantification

Quantification is a supervised learning task whose goal is to estimate the class distribution of an unlabeled sample, rather than to predict the class of individual instances [6,8]. Formally, let $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$ be a labeled training set, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{c_1, \dots, c_\ell\}$. A quantifier q is induced from D such that

$$q : \mathcal{A}^{\mathcal{X}} \rightarrow [0, 1]^\ell, \quad (1)$$

where $\mathcal{A}^{\mathcal{X}}$ denotes the universe of possible samples of $\mathcal{X}$. Given a test sample $a \in \mathcal{A}^{\mathcal{X}}$, the quantifier produces a vector $\hat{p} = [\hat{p}_1, \dots, \hat{p}_\ell]$ with $\sum_j \hat{p}_j = 1$. In the binary case, $\hat{p} = \hat{p}_\oplus$ denotes the estimated prevalence of the positive class, and p the corresponding true prevalence in a [15]. The goal is thus to produce $\hat{p}_\oplus$ as close as possible to p .

Most quantification methods rely on a classifier as an intermediate step, either a probabilistic classifier $h : \mathcal{X} \rightarrow [0, 1]^\ell$ or the corresponding crisp classifier $\hat{h}(x) = \arg \max_j h(x)_j$. The *Classify and Count* (CC) method [6] is the canonical baseline: given $\hat{h}$ trained on D , CC counts the predicted labels in a ,

$$\hat{p}_\oplus^{\text{CC}} = \frac{|\{x \in a : \hat{h}(x) = \oplus\}|}{|a|}. \quad (2)$$

CC has a well-known limitation: classification errors propagate directly to the prevalence estimate, producing systematically biased results whenever the class distribution shifts between training and test data, precisely the condition under which quantification is most needed [6,8]. This motivated the development of methods specifically designed for quantification, establishing it as a task in its own right rather than a by-product of classification.

Since quantification is a relatively recent area, methods have appeared under different assumptions and modeling strategies. *Aggregative* methods rely on the outputs of an underlying classifier, either hard predictions, posterior probabilities, or scores; *non-aggregative* methods estimate prevalences directly, without a classifier as an intermediate step [5].

A first line of work responds directly to CC's bias under label shift, which depends on the classifier's true and false positive rates, $\text{tpr} = P(\hat{h}(x) = \oplus \mid y = \oplus)$ and $\text{fpr} = P(\hat{h}(x) = \oplus \mid y = \ominus)$. Since these rates remain stationary under label shift, *Adjusted Classify and Count* (ACC) estimates them from held-out data to correct CC's raw counts [7]; in practice, however, this estimate can be fragile under imbalanced or small-sample conditions [8]. A second line replaces hard assignments with probabilistic information: PCC and PACC estimate prevalences from posterior probabilities rather than hard predictions [3], and EMQ [24] iteratively adjusts posteriors to account for shifting class priors, though both introduce a stronger dependence on the quality of the classifier's probabilistic outputs. A third, influential line is based on distribution matching, estimating the prevalences that best align the training and test mixtures of class-conditional distributions; recent comparative studies show these methods are highly competitive, although no single method dominates across all types of shift [25,9].

Overall, quantification has moved from simple counting strategies toward methods that explicitly model dataset shift, yet a large part of this literature still depends on the reliability of an underlying classifier's scores or posterior probabilities, a dependence that raises the broader question of calibration.

2.2 Calibration

Formally, a probabilistic predictor is said to be perfectly calibrated if, for every predicted value $v \in [0, 1]$,

$$P(\text{event occurs} \mid \text{prediction} = v) = v. \quad (3)$$

That is, among all occasions on which the predictor outputs v , the event must occur with frequency exactly v . This is a condition of statistical consistency

in aggregate, not per-instance accuracy, and the two notions are independent: a predictor may be well calibrated yet imprecise, or accurate on average yet miscalibrated at specific values.

Classifier calibration. Applied to classifiers by Platt [22], this condition requires that, given a classifier $h : \mathcal{X} \rightarrow [0, 1]^\ell$ with prediction $\hat{y}_i = \arg \max_j h(x_i)_j$ and confidence $\hat{s}_i = \max_j h(x_i)_j$, the fraction of instances correctly classified among those assigned confidence v equals v , for every $v \in [0, 1]$. Post-hoc techniques enforce this by fitting a transformation $\tau : [0, 1]^\ell \rightarrow [0, 1]^\ell$ on held-out data such that

$$h^{\text{cal}}(x_i)_j = \tau(h(x_i))_j, \quad (4)$$

where h^{cal} is the calibrated classifier. Proposed transformations range from parametric families, Platt Scaling [22], Beta calibration [12], Temperature Scaling [10], and Dirichlet calibration [13], to the non-parametric isotonic regression [2, 29], which imposes only monotonicity.

Calibration under dataset shift. When training and test data no longer follow the same distribution, calibration itself must be adapted. Under covariate shift, TransCal [28] and CPCS [21] reweight validation examples by importance weights. Under label shift, Bias-Corrected Temperature Scaling (BCTS) [1] augments Temperature Scaling with per-class bias terms, and LasCal [23] proposes a label-shift-robust objective; Head2Tail [30] instead targets shift arising from long-tailed class distributions. More recently, Moreo [16] proposed DMCAL, built directly on top of a quantification method to construct a corrected calibration map.

3 Related Works

The theoretical connection between calibration and quantification under dataset shift has recently been formalized. Tasche [27] shows that different dataset-shift assumptions lead to structurally distinct adjustment methods, some of which extend naturally from classification to quantification. Moreo et al. [16] further demonstrate that classifier calibration, quantification, and accuracy prediction are intrinsically related under dataset shift; and factors specific to each quantifier modulate how much calibration of the base classifier actually improves prevalence estimates.

In practice, this connection has mainly been exploited indirectly, by calibrating the base classifier before quantification. Probabilistic quantifiers such as PCC, PACC, and EMQ [24] depend directly on posterior probability estimates. A separate line of work corrects prevalence estimates directly, without recalibrating the base classifier, though without framing this correction as a calibration problem in its own right. The *Adjusted Classify and Count* (ACC) method [7] estimates the classifier’s true and false positive rates on held-out data and applies a linear adjustment to the raw CC estimate; PACC [3] applies the same adjustment to PCC using posterior-probability-based estimates of these rates. Viewed through the lens of calibration, both methods already correct the quantifier’s output

rather than the classifier’s scores, but they do so through a fixed, affine correction whose parameters are tied to a specific error-rate model. This restricts their applicability: the correction does not generalize to quantifiers whose bias is not well captured by an affine model, and it presupposes reliable estimates of the underlying error rates.

The idea of generalizing ACC and PACC’s linear correction through a learned regressor was first proposed by Moreo *et al.* [17] in the multi-label setting, where a regression function is trained to correct a quantifier’s prevalence estimates by exploiting correlations between class prevalences, without assuming the correction to be linear. This regression-based view is conceptually the closest antecedent to the present work, since it also learns a correction directly from the quantifier’s own output rather than from the classifier’s scores. The two approaches, however, address different problems and differ structurally rather than merely in scope. Moreo *et al.* use regression as a multi-label aggregation mechanism whose main purpose is to model dependencies among classes and improve their joint prevalence estimates; this mechanism requires at least two independent prevalence values to define a correlation, and is therefore only defined for three or more classes. In the binary case, the positive and negative prevalences are complementary by construction, so no such correlation exists to be exploited. The present work reinterprets this general idea from the perspective of calibration instead: it formalizes quantifier calibration as a distinct task, defines the condition a calibrated quantifier should satisfy, and introduces a generic post-hoc framework for learning correction functions over prevalence estimates.

Taken together, calibration at the quantifier level is theoretically motivated and, through ACC and PACC, already present in the literature, though only in a restricted parametric form; the closest attempt to generalize this correction targets multi-label class correlation rather than quantifier calibration itself. What remains open is a calibration method that imposes no parametric form and applies uniformly to any binary base quantifier. This is the gap the present work is designed to fill.

4 Method

This work proposes a post-hoc calibration method for binary quantifiers, called *QuantCal* (QCal), which systematically learns to correct prevalence estimates by exploiting the systematic relationship between the estimates produced by a quantifier and the true prevalence of validation samples with artificially controlled prevalences. By observing how the quantifier responds across the whole prevalence range, QCal characterizes the bias it leaves in its estimates and learns a monotone correction function that maps raw estimates to corrected ones directly at the prevalence level [5]¹, thereby producing more reliable estimates under label shift.

¹ We use the term *batch* (also referred to as *sample* in the quantification literature [5]) to denote a set of unlabeled instances for which the class prevalence is to be estimated.

Before describing the proposal, we formalize the notion of quantifier calibration and show that ACC and PACC already instantiate this framework through a restricted parametric correction.

4.1 Quantifier Calibration

Section 2.2 established that classifier calibration is an instance of Murphy’s general calibration condition (Equation (3)), applied to the classifier’s confidence scores as the predicted quantity and the correctness of the prediction as the outcome. We now extend this condition to quantifiers, where the predicted quantity is the prevalence estimate $\hat{p}_\oplus = q(a)_\oplus$ and the outcome is the true prevalence p of the positive class in the test sample a .

Definition 1 (Quantifier Calibration). *A quantifier q is said to be perfectly calibrated if, for every prevalence value $p \in [0, 1]$:*

$$\mathbb{E}[\hat{p}_\oplus \mid p] = p, \quad (5)$$

where $\hat{p}_\oplus = q(a)_\oplus$ is the prevalence estimate produced by q on a test sample a , and conditioning on p denotes that the true prevalence of the positive class in a is p . That is, among all test samples whose true positive-class prevalence is p , the expected prevalence estimate produced by q must equal p exactly.

Definition 1 is a direct instantiation of Equation (3): the predictor is the quantifier q , the prediction is $\hat{p}_\oplus$, and the outcome is the true prevalence p . The two equations express the same condition in different but equivalent forms. For binary outcomes, such as the occurrence of an event or the correctness of a classification, the calibration condition is naturally stated as a conditional probability, since $P(A \mid B) = \mathbb{E}[1[A] \mid B]$ for any event A . For quantifiers, the outcome, namely the true prevalence, is a continuous quantity in $[0, 1]$ rather than a binary event; thus, the condition is stated directly as a conditional expectation. In both cases, the requirement is the same: the predicted value must be, on average, exactly the value it purports to predict.

Remark 1 (Calibration and accuracy are independent notions). A quantifier may achieve low expected error while systematically violating Equation (5) at specific prevalence values, that is, it may be accurate on average yet miscalibrated across the prevalence range. Conversely, a quantifier may satisfy Equation (5) at every prevalence point yet exhibit high variance, yielding poor accuracy. Calibration and accuracy are therefore complementary properties, mirroring their well-known independence for classifiers.

Post-hoc quantifier calibration. By analogy with post-hoc classifier calibration (Equation (4)), a post-hoc quantifier calibration method learns a correction function $\phi : [0, 1] \rightarrow [0, 1]$ from held-out data, such that the calibrated quantifier q^{cal} satisfies:

$$\hat{p}_\oplus^{\text{cal}} = \phi(\hat{p}_\oplus), \quad (6)$$

with the goal that $\mathbb{E}[\phi(\hat{p}_{\oplus}) \mid p] = p$ for all $p \in [0, 1]$. The following result establishes that such a correction function exists for the CC quantifier, and that ACC and PACC already instantiate this general framework through a specific parametric choice of ϕ .

Proposition 1 (Existence of a perfect correction for CC). *Let h be a binary classifier with $tpr \neq fpr$, and let $\hat{p}^{CC}$ denote the CC estimator. Then there exists a correction function $\phi^* : [0, 1] \rightarrow [0, 1]$ such that the calibrated quantifier q^{cal} obtained by applying ϕ^* to $\hat{p}^{CC}$ is perfectly calibrated in the sense of Definition 1. Moreover, ϕ^* is strictly monotone.*

Proof. Under label shift, the expected CC estimate conditional on the true prevalence p is:

$$\mathbb{E}[\hat{p}^{CC} \mid p] = p \cdot tpr + (1 - p) \cdot fpr = fpr + p(tpr - fpr). \quad (7)$$

The assumption $tpr \neq fpr$ holds for any classifier that performs better than random, since $tpr = fpr$ would mean that the probability of predicting the positive class is the same regardless of the true class, and the classifier's predictions would carry no information about the true label. Under this assumption, Equation (7) is a strictly monotone affine function of p , hence invertible. Its inverse

$$\phi^*(z) = \frac{z - fpr}{tpr - fpr} \quad (8)$$

is strictly monotone and satisfies $\mathbb{E}[\phi^*(\hat{p}^{CC}) \mid p] = p$ for all $p \in [0, 1]$, which is precisely the condition of Definition 1.

Note that Equation (7) also shows that CC itself is not calibrated: its expected estimate equals p only when $tpr = 1$ and $fpr = 0$, that is, only for a perfect classifier.

Corollary 1 (ACC and PACC are parametric instances). *ACC and PACC are post-hoc quantifier calibration methods in the sense of Equation (6), with a parametric affine correction function.*

Proof. The correction applied by ACC [7] is:

$$\hat{p}^{ACC} = \frac{\hat{p}^{CC} - fpr_c}{tpr_c - fpr_c}, \quad (9)$$

where tpr_c and fpr_c are estimated from held-out data. This is exactly ϕ^* of Equation (8) with estimated parameters, applied to the output of CC. PACC [3] applies the same affine correction to the output of PCC, with parameters estimated from posterior probability scores rather than hard predictions. By Proposition 1, both yield perfectly calibrated quantifiers when their parameters are known exactly, which is the sense in which they are Fisher-consistent.

Corollary 1 characterizes ACC and PACC as instances of the general post-hoc quantifier calibration framework of Equation (6), in which ϕ is restricted to a parametric affine family and its two parameters are estimated from class-conditional error rates. A general-purpose post-hoc quantifier calibration method need not impose either restriction: it may instead learn ϕ directly from data, without assuming an affine (or any other fixed) parametric form and without requiring explicit estimation of tpr and fpr . Section 4.2 instantiates this broader framework by proposing QCal, a post-hoc quantifier calibration method that learns ϕ nonparametrically from data.

4.2 Proposal

Section 4.1 established that ACC and PACC instantiate the general post-hoc quantifier calibration framework of Equation (6) through a parametric affine correction function, estimated from class-conditional error rates. We now describe how QCal instantiates this same framework without either restriction, learning ϕ directly from data. The central challenge is learning the correction function ϕ without access to labeled test batches. QCal addresses this by exploiting two properties of the base quantifier that can be measured on the training data: the variability of its prevalence estimates across different prevalence conditions, and the systematic relationship between those estimates and the true prevalence of validation samples with artificially controlled prevalences. By collecting the quantifier’s behavior over validation batches with artificially controlled prevalences, QCal builds a characterization of how the quantifier errs, and fits a monotone correction function ϕ that maps raw estimates to corrected ones directly at the prevalence level. The complete architecture, illustrated in Fig. 1, proceeds in six steps.

- Step 1 - Dataset splitting:** In this step, given the training set D , we split it into two parts, D_{train} and D_{val} .
- Step 2 - Generation of validation batches:** From D_{val} , the Uniform Prevalence Protocol (UPP) is applied [5], producing a collection of batches. Unlike the Artificial Prevalence Protocol (APP), which sweeps a fixed grid of target prevalences, the UPP draws the target prevalence of each batch uniformly at random from the unit simplex by means of the Kraemer algorithm [5], thereby covering the prevalence range continuously rather than at a discrete set of points. Here, R denotes the number of batches sampled and *batch_size* the number of instances per batch.
- Step 3 - Quantifier training:** In this step, the quantifier, along with its underlying base classifier, is trained using the training data D_{train} .
- Step 4 - Validation (Repetition of Steps 1–3):** The three previous steps are executed n_{val} times, where $n_{\text{val}} \in \mathbb{N}^*$, each with a distinct stratified partition. Each repetition generates a new internal quantifier $q^{(k)}$ and produces a new collection of pairs $(\hat{p}_{j,\oplus}, p_{j,\oplus})$ from the batches generated by the UPP, where $\hat{p}_{j,\oplus}$ is the positive-class prevalence predicted by $q^{(k)}$ for the j -th batch and $p_{j,\oplus}$ is its corresponding true positive-class prevalence. Since the task is

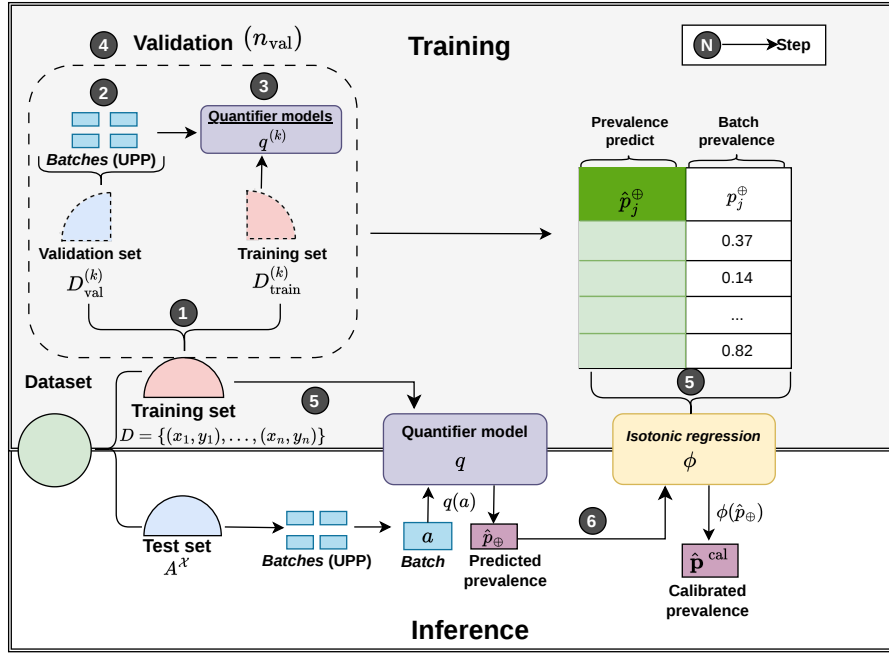


Fig. 1: Architecture of the method

binary, the negative-class component is fully determined by complement and is therefore redundant; only the positive-class component is retained. At the end of the n_{val} validation repetitions, all pairs across all repetitions and all batches are accumulated into a single collection $\mathcal{D}_{\phi}$, without any aggregation, yielding $n_{\text{val}} \cdot R$ pairs in total. These pairs depart systematically from the true prevalence across the whole prevalence range, as illustrated in Fig. B.1.

Step 5 - Fitting the correction function and the final quantifier: Using the pairs $(\hat{p}_{j,\oplus}, p_{j,\oplus}) \in \mathcal{D}_{\phi}$ accumulated across all n_{val} repetitions in Step 4, the correction function ϕ is fitted by isotonic regression, taking each raw estimate $\hat{p}_{j,\oplus}$ as input and the corresponding true prevalence $p_{j,\oplus}$ as target. The function is constrained to be non-decreasing and bounded to $[0, 1]$. Monotonicity encodes the natural requirement that a higher raw estimate should never be mapped to a lower corrected prevalence: it preserves the ordering induced by the base quantifier and acts as a strong regularizer, allowing ϕ to absorb systematic bias while remaining resistant to overfitting the calibration pairs. The $[0, 1]$ bound guarantees that every corrected value is a valid prevalence, so no external clipping is required. In parallel, the final base quantifier q is trained on the complete set D , maximizing the data available for the production model; this quantifier is entirely distinct from the internal quantifiers of Steps 3 and 4, which served only for calibration.

The trained method corresponds to the pair (q, ϕ) . The complete training procedure is summarized in Algorithm A.1.

Step 6 - Inference: At inference time, detailed in Algorithm A.2, given an unlabeled test sample $a \in \mathcal{A}^{\mathcal{X}}$, the base quantifier q is first applied to obtain an initial estimate, from which the positive-class component $\hat{p}_{\oplus} = (q(a))_{\oplus}$ is taken. The correction function ϕ is then applied to this scalar, and the negative-class prevalence is recovered by complement: $\hat{\mathbf{p}}^{\text{cal}} = (1 - \phi(\hat{p}_{\oplus}), \phi(\hat{p}_{\oplus}))$. By construction, $\hat{\mathbf{p}}^{\text{cal}}$ has non-negative components that sum to one, so the calibrated output is guaranteed to lie in the probability simplex.

Fig. B.1 illustrates the calibration pairs of Step 4 and the correction function ϕ fitted on them in Step 5, for two representative datasets with CC as the base quantifier. In both cases the point clouds depart systematically from the identity, confirming that CC is not calibrated in the sense of Definition 1, and the magnitude of the required correction differs markedly between them. On balance.3 the fitted curve lies only slightly above the identity, indicating a mild underestimation that QCal barely needs to correct. On ctg.3 the same base classifier leaves a pronounced systematic underestimation, and the fitted curve rises well above the identity across most of the prevalence range. Even between two datasets on which the classifier separates the classes well, the correction that restores calibration differs substantially in shape and steepness. This is what motivates learning ϕ nonparametrically rather than fixing a parametric form in advance: the affine correction of ACC and PACC (Corollary 1) can only rescale and shift the estimate, whereas isotonic regression recovers an arbitrary monotone relationship and adapts its shape to each dataset.

Fig. B.2 shows the effect of this learned correction on held-out test samples, confirming that it transfers beyond the calibration pairs. On balance.3, where CC is already close to calibrated and ϕ is nearly the identity, the corrected estimates are almost indistinguishable from the raw ones and the MAE barely moves ($0.0504 \rightarrow 0.0352$): QCal correctly leaves an already-calibrated quantifier essentially untouched. On ctg.3, the raw CC estimates lie systematically below the diagonal (underestimation), and ϕ lifts them onto it, cutting the MAE by more than a factor of three ($0.1416 \rightarrow 0.0501$). The contrast is exactly the behaviour a bias-correction method should exhibit: it intervenes in proportion to the systematic error present, correcting the biased ctg.3 strongly while leaving the already-calibrated balance.3 essentially unchanged. The residual scatter that remains on ctg.3 is expected, since QCal removes the systematic component of the error but not its variance. The systematic evaluation over all datasets is reported in Section 6.

5 Experiments

This section describes the experimental setup used to evaluate QCal empirically as a general-purpose calibration method. We first present the collection of binary

datasets used throughout the evaluation (Section 5.1), and then detail how each step of the architecture introduced in Section 4 was instantiated, including the eight base quantifiers to which QCal is applied independently (Section 5.2).

5.1 Datasets

We selected 106 binary classification datasets from the OpenML [4] and UCI Machine Learning [11] repositories. As selection criterion, we retained only datasets whose minority class contains at least 150 instances, ensuring a sufficient number of minority-class examples for reliable estimation of quantification performance. The resulting collection is intentionally diverse: the number of instances ranges from 625 to 245,057, the number of attributes from 2 to 970, and the minority class proportion from $\approx 4\%$ to 50%, covering balanced, mildly imbalanced, and severely imbalanced scenarios. All experiments were repeated 30 times to mitigate the effect of individual run variability.

5.2 Setup

The configuration of each step in the architecture proposed in Section 4 is as follows: in **Step 1**, we apply a stratified split allocating 70% of the instances to D_{train} and 30% to D_{val} . In **Step 2**, we apply UPP over D_{val} by means of the Kraemer algorithm [5], with *batch_size* = 100 instances and $R = 1100$ batches, whose target prevalences are drawn uniformly at random from the unit simplex. In **Step 3**, the eight base quantifiers are CC, ACC, PACC, EMQ, FM, DyS, and MS, from the *mlquantify* library² [14], and KDEy in its maximum-likelihood variant (KDEy-ML), from the QuaPy library³, in which the mixture parameter is estimated by maximum likelihood, which is equivalent to minimizing the Kullback-Leibler divergence between the test distribution and the mixture of the class-conditional densities [16]. Each of them is calibrated by QCal independently, so that the base quantifier and the internal quantifiers of Step 4 always belong to the same method. In **Step 4**, we set $n_{\text{val}} = 3$, yielding $n_{\text{val}} \cdot R = 3300$ prediction pairs in $\mathcal{D}_\phi$. In **Step 5**, ϕ is fitted by isotonic regression from *scikit-learn*⁴, under the constraints described in Section 4. **Step 6** requires no additional configuration.

All base quantifiers are aggregative and share the same base classifier, a *Logistic Regression* with default parameters. Since it is fitted by minimizing the log-loss, a strictly proper scoring rule, it already yields approximately calibrated posteriors on the training distribution; no additional probability calibration was thus applied, leaving QCal as the only source of correction in the pipeline. Each dataset of Section 5.1 is evaluated under 30 stratified train/test splits (70%/30%), which mitigates the effect of individual run variability; UPP is applied over the test portion with the same configuration of Step 2. Performance is measured by

² <https://luizfernandolj.github.io/mlquantify/>

³ <https://hlt-isti.github.io/QuaPy/index.html>

⁴ <https://scikit-learn.org/stable/index.html>

the Mean Absolute Error (MAE), averaged over the 1100 batches of each split and then over the 30 splits, yielding one score per method and per dataset. Each dataset contributes a single observation, and each pair $(q, \text{QCal}(q))$ is compared with the two-sided Wilcoxon signed-rank test at $\alpha = 0.05$, with p -values Holm-corrected for the eight comparisons. Reproducibility: the source code and datasets used in this work are publicly available⁵.

6 Results

We first assess whether QCal improves the base quantifier it is applied to. Table 1 reports, for each of the eight base quantifiers q , the MAE of q and of $\text{QCal}(q)$ averaged over the 106 datasets, together with the number of datasets on which QCal wins or loses, the average rank of both methods among the 16 competitors, and the p -value of a two-sided Wilcoxon signed-rank test over the per-dataset scores.

QCal reduces the mean MAE of every base quantifier, and all eight improvements remain significant at the 5% level after Holm correction. The magnitude of the gain, however, varies widely. It is largest for CC (48.9%), the only method that performs no correction of its own, and smallest for the quantifiers that already model the shift explicitly, such as FM (14.7%) and PACC (15.1%). The win/loss counts follow the same pattern: QCal wins on 102 of the 106 datasets when applied to CC, but the margin narrows to roughly 55–70 wins for the shift-aware quantifiers, and MS is the only case in which wins and losses are evenly split (53/53) despite a 21.0% reduction in mean MAE, an indication that, for this quantifier, QCal acts mainly on the datasets where the base method fails badly rather than uniformly across the benchmark. Taken together, these results are consistent with QCal acting as a bias-correction mechanism, as discussed in Section 4: the more residual bias the base quantifier leaves in its estimates, the more there is for QCal to remove.

The average ranks in Table 1 summarize each method with a single number and therefore hide how stable the improvement is across datasets. Fig. 2 shows the full distribution of the per-dataset ranks of the 16 methods, ordered from left to right by increasing average rank. Three observations follow.

First, the ordering is strictly *paired*: every $\text{QCal}(q)$ appears to the left of its own base quantifier q , without exception. The gain reported in Table 1 is thus not an artifact of a few datasets with large errors, it is visible in the bulk of the rank distribution.

Second, the top positions are held by $\text{QCal}(\text{DyS})$ (3.77), $\text{QCal}(\text{KDEy-ML})$ (3.83), and $\text{QCal}(\text{EMQ})$ (4.38), which interleave with and displace the uncorrected DyS (4.30), KDEy-ML (5.70), and EMQ (6.41). The violins of these three corrected variants are markedly concentrated in the first ranks, whereas their uncorrected counterparts spread over a wider range of positions: QCal not only lowers the expected rank of the strongest quantifiers but also makes their behaviour more consistent across datasets.

⁵ <https://github.com/isabellaloren4/Quantcal/>

Table 1: Effect of QCal on each base quantifier q over the 106 datasets. MAE: mean $\pm$ standard deviation across datasets; lower is better. Δ : relative reduction of the mean MAE obtained by QCal(q) over q . W/L: number of datasets on which QCal(q) achieves a lower / higher MAE than q (wins and losses of QCal; they sum to 106, as no exact ties occurred). Rank: average rank of the method among all 16 methods, lower is better. p-value: two-sided Wilcoxon signed-rank test over the 106 per-dataset MAE scores, Holm-corrected for the eight comparisons.

	MAE		Δ	W/L	Rank		
	q	QCal(q)			q	QCal(q)	p-value
CC	.1780 $\pm$.1543	.0910 $\pm$.0810	48.9%	102/4	15.27	11.72	6.5e-18
ACC	.1191 $\pm$.1361	.0913 $\pm$.0809	23.4%	62/44	13.10	12.18	8.8e-05
PACC	.0808 $\pm$.0972	.0686 $\pm$.0626	15.1%	57/49	8.93	8.41	2.3e-02
EMQ	.0820 $\pm$.0963	.0656 $\pm$.0644	20.0%	70/36	6.41	4.38	4.3e-06
FM	.0861 $\pm$.1004	.0735 $\pm$.0663	14.7%	58/48	10.34	9.95	4.1e-03
DyS	.0757 $\pm$.0991	.0637 $\pm$.0614	15.9%	59/47	4.30	3.77	2.3e-02
MS	.0962 $\pm$.1182	.0759 $\pm$.0682	21.0%	53/53	9.31	8.40	2.3e-02
KDEy-ML	.0849 $\pm$.1147	.0636 $\pm$.0629	25.2%	70/36	5.70	3.83	3.0e-07

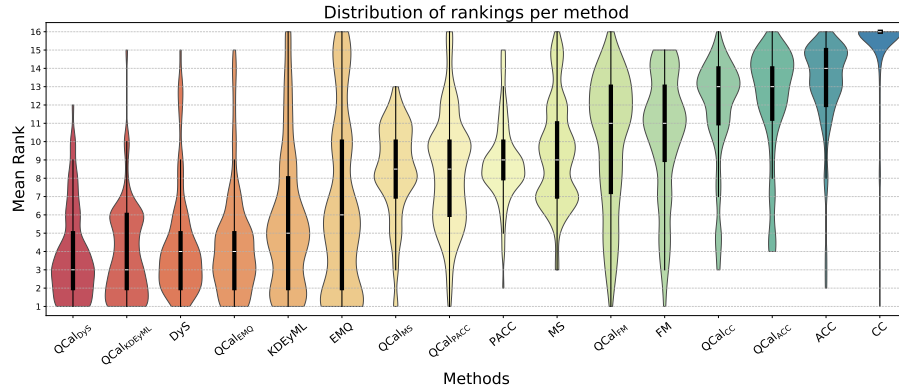


Fig. 2: Distribution of the ranks obtained by each of the 16 methods over the 106 datasets. For every dataset, the methods are ranked by MAE (rank 1 = best, rank 16 = worst); each violin shows the resulting distribution of ranks, and the inner box reports the median and the interquartile range. Methods are sorted from left to right by increasing average rank. QCal(q) denotes the base quantifier q corrected by QCal. Every corrected variant precedes its own base quantifier.

Third, the effect on the weakest methods is the most pronounced. CC has a rank distribution concentrated at the very bottom of the scale, as expected from a method that ignores the classifier’s bias entirely; once corrected, QCal(CC) moves to an average rank of 11.72, overtaking not only CC but also ACC (13.10), a

method explicitly designed to correct CC. The same holds for QCal(ACC) (12.18) with respect to ACC. In other words, a single post-hoc isotonic correction applied to the naive classify-and-count estimate is enough to outperform the classical adjusted count, which is precisely the behaviour one expects if the dominant error term of these quantifiers is a systematic, prevalence-dependent bias.

7 Conclusion

We extended calibration from classifiers to quantifiers, formalizing the condition a binary quantifier must satisfy to be calibrated and showing that ACC and PACC already instantiate it through a fixed parametric form. Building on this, we introduced QCal, which learns a monotone correction directly from a quantifier’s own prevalence estimates via isotonic regression, without assuming a parametric form or requiring a separate derivation for each base quantifier.

QCal significantly reduces the mean absolute error of all eight base quantifiers evaluated across 106 binary datasets. The gain is largest for CC, the only method with no correction of its own, and remains significant even for quantifiers that already model label shift explicitly. Notably, a single post-hoc correction applied to CC is enough to outperform ACC, indicating that the dominant error of these quantifiers is a systematic, prevalence-dependent bias that QCal is particularly well suited to remove.

Several directions remain open for future work: evaluating correction estimators beyond isotonic regression, assessing the effect of different batch sizes during calibration and testing, and extending QCal to the multiclass setting.

Acknowledgments. We thank the Graduate Program in Electrical and Computer Engineering (PGECE) for the funding and support. We also thank the Computational Entomology Laboratory (COEN) for providing the mlquantify library.

References

1. Amr Alexandari, Anshul Kundaje, and Avanti Shrikumar. Maximum likelihood with bias-corrected calibration is hard-to-beat at label shift adaptation. In *International conference on machine learning*, pages 222–232. PMLR, 2020.
2. Richard E. Barlow, David J. Bartholomew, James M. Bremner, and Hugh D. Brunk. *Statistical Inference under Order Restrictions: The Theory and Application of Isotonic Regression*. John Wiley & Sons, London and New York, 1972.
3. Antonio Bella, Cesar Ferri, Jose Hernandez-Orallo, and Maria Jose Ramirez-Quintana. Quantification via Probability Estimators. In *2010 IEEE International Conference on Data Mining*, pages 737–742, Sydney, Australia, December 2010. IEEE.
4. Bernd Bischl, Giuseppe Casalicchio, Taniya Das, Matthias Feurer, Sebastian Fischer, Pieter Gijsbers, Subhaditya Mukherjee, Andreas C Müller, László Németh, Luis Ojala, et al. Openml: Insights from 10 years and more than a thousand papers. *Patterns*, 6(7), 2025.

5. Andrea Esuli, Alessandro Fabris, Alejandro Moreo, and Fabrizio Sebastiani. *Learning to Quantify*, volume 47 of *The Information Retrieval Series*. Springer International Publishing, 2023.
6. George Forman. Counting Positives Accurately Despite Inaccurate Classification. In *Machine Learning: ECML 2005*, volume 3720, pages 564–575. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
7. George Forman. Quantifying counts, costs, and trends accurately via machine learning. Technical report, Technical report, HP Laboratories, Palo Alto, CA, 2008.
8. Pablo González, Alberto Castaño, Nitesh V. Chawla, and Juan José Del Coz. A Review on Quantification Learning. *ACM Computing Surveys*, 50(5):74:1–74:40, September 2017.
9. Pablo González, Alejandro Moreo, and Fabrizio Sebastiani. Binary quantification and dataset shift: An experimental investigation. *Data Mining and Knowledge Discovery*, 38(4):1670–1712, 2024.
10. Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.
11. Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The UCI Machine Learning Repository. <https://archive.ics.uci.edu>, 2026. Last accessed: 2026/05/19.
12. Meelis Kull, Telmo M. Silva Filho, and Peter Flach. Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics*, 11(2):5052 – 5080, 2017.
13. Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. *Advances in neural information processing systems*, 32, 2019.
14. Luiz Fernando Luth. MLQuantify: A Python Package for Quantification, 2025.
15. André Maletzke, Denis Reis, Waqar Hassan, and Gustavo Batista. Accurately Quantifying under Score Variability. December 2021.
16. Alejandro Moreo. On the interconnections of calibration, quantification, and classifier accuracy prediction under dataset shift. *arXiv preprint arXiv:2505.11380*, 2025.
17. Alejandro Moreo, Manuel Francisco, and Fabrizio Sebastiani. Multi-label quantification. *ACM Transactions on Knowledge Discovery from Data*, 18(1):1–36, 2023.
18. Allan H Murphy. Scalar and vector partitions of the probability score: Part i. two-state situation. *Journal of Applied Meteorology and Climatology*, 11(2):273–282, 1972.
19. Allan H. Murphy and Robert L. Winkler. Reliability of subjective probability forecasts of precipitation and temperature. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 26(1):41–47, 1977.
20. Allan H. Murphy and Robert L. Winkler. A general framework for forecast verification. *Monthly Weather Review*, 115:1330–1338, 1987.
21. Sangdon Park, Osbert Bastani, James Weimer, and Insup Lee. Calibrated prediction with covariate shift via unsupervised domain adaptation. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3219–3229. PMLR, 2020.
22. John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, volume 10, pages 61–74. MIT Press, 1999.

23. Teodora Popordanoska, Gorjan Radevski, Tinne Tuytelaars, and Matthew B. Blaschko. Lascal: Label-shift calibration without target labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, 2024.
24. Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1):21–41, 2002.
25. Tobias Schumacher, Markus Strohmaier, and Florian Lemmerich. A comparative evaluation of quantification methods. *Journal of Machine Learning Research*, 26(55):1–54, 2025.
26. Telmo Silva Filho, Hao Song, Miquel Perello-Nieto, Raul Santos-Rodriguez, Meelis Kull, and Peter Flach. Classifier calibration: a survey on how to assess and improve predicted class probabilities. *Machine Learning*, 112(9):3211–3260, 2023.
27. Dirk Tasche. Recalibrating binary probabilistic classifiers. In *Proceedings of the 5th International Workshop on Learning to Quantify, LQ 2025, ECML/PKDD Workshop*, pages 47–63, 2025.
28. Ximei Wang, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Transferable calibration with lower bias and variance in domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 19212–19223, 2020.
29. Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 694–699, 2002.
30. Zhisheng Zhong, Jiequan Cui, Shu Liu, and Jiaya Jia. Improving calibration for long-tailed recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16489–16498, 2021.

A QCal Algorithm

A.1 Training

Algorithm A.1: QCal: Training

Input : D , labeled training set
Input : q , quantification algorithm
Input : $n_{\text{val}} \in \mathbb{N}^*$, R , *batch_size*, validation loop and UPP parameters
 /* Calibration pair extraction phase */
 1 $\mathcal{D}_\phi \leftarrow \emptyset$;
 2 **for** $k \leftarrow 1$ **to** n_{val} **do**
 3 $(D_{\text{train}}^{(k)}, D_{\text{val}}^{(k)}) \leftarrow \text{stratified-split}(D, 0.7)$;
 4 train $q^{(k)}$ on $D_{\text{train}}^{(k)}$ using algorithm q ;
 5 $\mathcal{S}^{(k)} \leftarrow \text{UPP}(D_{\text{val}}^{(k)}; R, \text{batch_size})$;
 6 **for each** $(S_j, p_{j,\oplus}) \in \mathcal{S}^{(k)}$ **do**
 7 $\hat{p}_{j,\oplus}^{(k)} \leftarrow (q^{(k)}(S_j))_{\oplus}$;
 8 $\mathcal{D}_\phi \leftarrow \mathcal{D}_\phi \cup \{(\hat{p}_{j,\oplus}^{(k)}, p_{j,\oplus})\}$;
 9 **end**
 10 **end**
 /* Isotonic regressor and final quantifier */
 11 $\phi \leftarrow \text{IsotonicRegression}(\mathcal{D}_\phi; y_{\min} = 0, y_{\max} = 1, \text{increasing})$;
 12 train q on D ;
Return: q, ϕ

A.2 Inference

Algorithm A.2: QCal: Inference

Input : $a \in \mathcal{A}^{\mathcal{X}}$, unlabeled test sample
Input : q , trained base quantifier
Input : ϕ , trained isotonic correction function
 1 $\hat{p}_{\oplus} \leftarrow (q(a))_{\oplus}$;
 2 $\hat{p}_{\oplus}^{\text{cal}} \leftarrow \phi(\hat{p}_{\oplus})$;
 3 $\hat{p}_{\ominus}^{\text{cal}} \leftarrow 1 - \hat{p}_{\oplus}^{\text{cal}}$;
Return: $\hat{\mathbf{p}}^{\text{cal}} = (\hat{p}_{\ominus}^{\text{cal}}, \hat{p}_{\oplus}^{\text{cal}})$

B Charts

B.1 Calibration

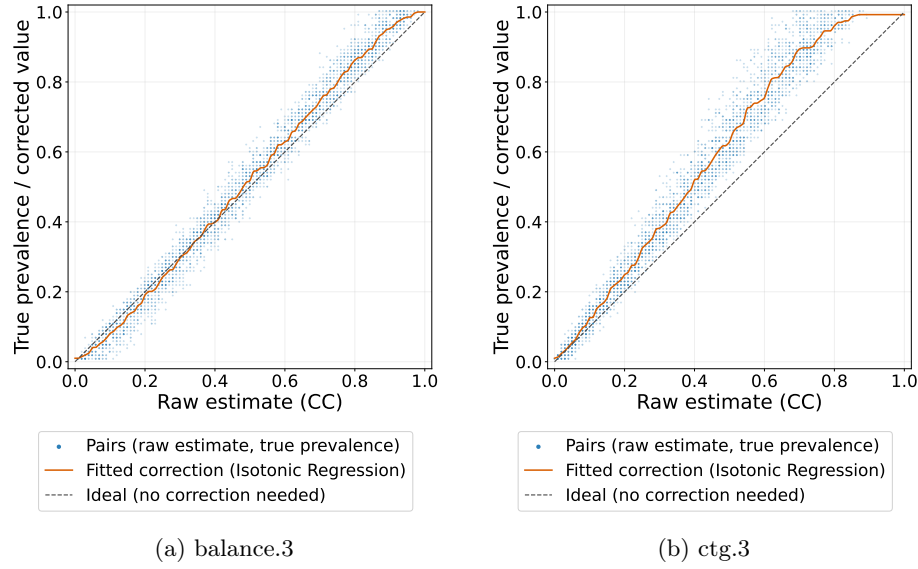


Fig. B.1: Calibration pairs and the resulting correction function ϕ on two representative datasets, with CC as the base quantifier. Each point is a pair $(\hat{p}_{j,\oplus}, p_{j,\oplus}) \in \mathcal{D}_\phi$ from a UPP validation batch over the n_{val} repetitions of Step 4: the horizontal axis is the raw estimate produced by CC, and the vertical axis the true prevalence of the batch. The solid line is the isotonic regression fitted on these pairs, i.e. the correction function ϕ of Step 5; the dashed line marks the identity, along which no correction would be needed. The pairs shown are those on which ϕ was fitted, so the figure is a diagnostic of the learned correction rather than evidence of generalization; the latter is assessed on held-out test samples in Fig. B.2 and Section 6.

B.2 Calibrated Predictions

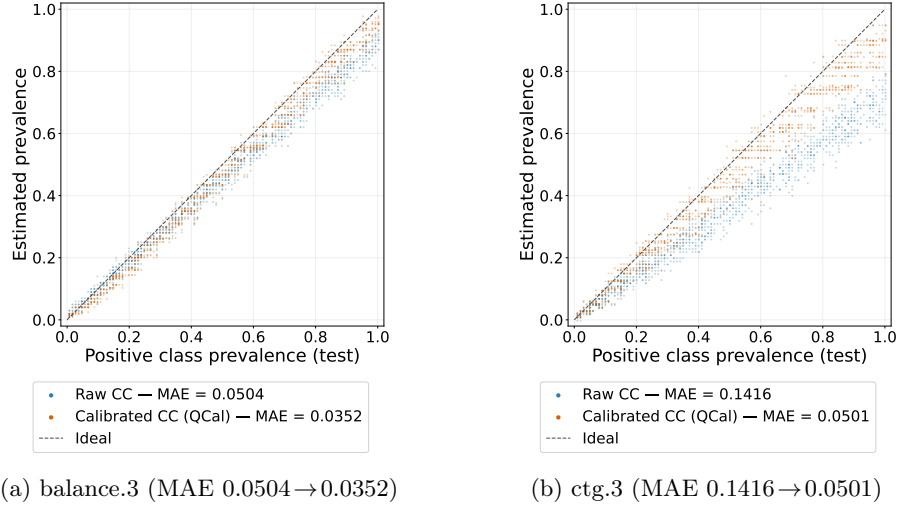


Fig. B.2: Effect of the correction function ϕ of Fig. B.1 on held-out test samples, for the same two datasets and CC as the base quantifier. Each panel plots the estimated prevalence against the true test prevalence: raw CC estimates (blue) versus QCal-corrected estimates (orange), each annotated with its mean absolute error (MAE); the dashed line marks the identity. Unlike Fig. B.1, these estimates come from test samples not seen during fitting, so the figure measures generalization of the learned correction. On both datasets the corrected estimates lie closer to the identity and the MAE decreases, with the larger gain on the more strongly biased ctg.3. A full evaluation across all datasets is reported in Section 6.

Drift Happens: An Empirical Study of Neural Architecture Robustness to Temporal Distribution Shift

Robin Holzinger* and Riccardo Colletti*

Department of Electrical Engineering and Computer Sciences,
University of California, Berkeley, USA

Abstract. Real-world data distributions evolve over time, inducing temporal distribution shift that can substantially degrade the reliability of deployed machine learning systems. However, the extent to which architectural choices and their associated inductive biases affect temporal robustness remains insufficiently understood.

We present a systematic empirical comparison of temporal robustness across three heterogeneous, time-indexed domains encompassing image classification, multi-label text classification, and text regression tasks. Using a unified evaluation framework based on temporal drift matrices, we train models on cumulative historical data and evaluate their performance on both earlier and later time periods, thereby quantifying cross-temporal generalization. Our study spans model families ranging from simple multilayer perceptrons and convolutional networks to recurrent networks and pretrained Transformer-based encoders.

Collectively, the results show that architectural inductive biases systematically shape temporal robustness: models whose inductive biases lead them to exploit localized, highly discriminative features attain the highest in-distribution accuracy, yet those features are often the ones that change most over time, so these models degrade fastest, while pretrained encoders that draw on coarser, more stable representations drift more gradually. These observations offer practical guidance for selecting architectures for real-world systems subject to temporal drift.

Keywords: Temporal distribution shift · Concept drift · Inductive bias · Neural architecture robustness · Pretrained encoders

1 Introduction

Machine learning models are typically trained under the assumption that training and test data are drawn from the same distribution. In practice, this assumption rarely holds: real-world data evolves over time, exhibiting *temporal distribution shift* that can degrade model performance in ways that standard held-out evaluation fails to capture. A model that achieves high accuracy on held-out data from the same time period may fail when deployed on future inputs.

* Both authors contributed equally.

While distribution shift is well-documented, less understood is how architectural choices influence a model’s robustness to temporal drift.

Do different inductive biases (the translation invariance of convolutions, the sequential modeling of recurrent nets, the attention of Transformers) lead to different rates of temporal degradation?

Do frozen, pretrained encoders resist temporal drift better than models trained end to end?

These questions have practical implications for model selection, yet systematic comparisons across architectures and domains remain scarce.

This work investigates the temporal robustness of neural classifiers across three domains: image classification (**Yearbook**), text regression (**Amazon Reviews**), and multi-label text classification (**arXiv**). For each domain, we evaluate diverse architectures, from simple baselines to pretrained transformers, using a unified framework based on temporal drift matrices and provide qualitative explanations for model degradation via gradient saliency maps. Our contributions are threefold:

- We provide a unified empirical assessment of temporal robustness across three long-range, time-indexed domains, enabling direct comparison of how neural architectures behave under temporal distribution shift.
- We organize time-indexed evaluation in the spirit of Wild-Time [33] into temporal drift matrices, a compact representation that quantifies cross-temporal generalization by measuring performance when training on cumulative historical data and testing on both earlier and later time periods.
- We systematically compare a broad spectrum of model families, from multi-layer perceptrons, convolutional and residual networks, and recurrent networks to transformers trained on the data and frozen pretrained encoders, highlighting how architectural assumptions relate to degradation patterns across modalities and tasks.

Taken together, our results characterize how performance deteriorates as the temporal gap between training and evaluation widens across domains and model classes. **The features an inductive bias extracts within the training period are what lift in-distribution performance, yet they are the most tied to it and the first to degrade as the data drifts, so the very features that make a model accurate in distribution are the least robust over time.** Frozen pretrained encoders, relying on coarser and more transferable representations, trade in-distribution accuracy for steadier degradation. These findings guide architecture selection in non-stationary real-world environments.

2 Background and Related Work

2.1 Temporal Distribution Shift

A central challenge in real-world machine learning systems is that the data-generating process is rarely stationary. As models are deployed over months,

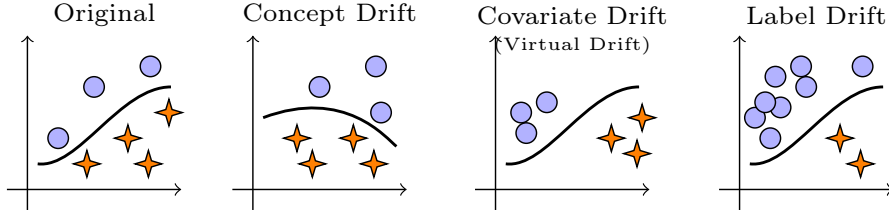


Fig. 1: Drift types in a binary classification setting. Circles and stars indicate the label classes; the curve represents the decision boundary. *Concept drift* induces a change in the decision boundary, *Covariate drift* (virtual drift) changes the input distribution, and *Label drift* alters the relative class frequency [10].

years, or decades, both inputs and label semantics may evolve, causing systematic discrepancies between distributions encountered during training and those observed at inference. Such temporal evolution, commonly referred to as *temporal distribution shift*, is pervasive across domains. Understanding the structure of this shift is essential for assessing and improving temporal robustness. However, prior work has largely focused on static or domain-level distribution shifts [2, 12, 20], with limited attention to drift that unfolds sequentially over time, particularly in non-generated, in-the-wild settings [33].

Following established terminology in concept drift research [10], temporal distribution shift can be characterized along three complementary axes (Figure 1):

Covariate shift occurs when the input distribution $P(X)$ changes while the conditional $P(Y | X)$ remains fixed. This type of shift commonly arises in natural data streams where observational setups, sociocultural conventions, or user behavior gradually evolve.

Label shift refers to changes in the marginal label distribution $P(Y)$. Long-term textual or behavioral datasets frequently exhibit such imbalance drift as the prevalence of topics, categories, or rating patterns changes over time.

Concept drift denotes changes in the conditional distribution $P(Y | X)$, implying that identical inputs may correspond to different labels at different time points. This form of drift is particularly pronounced in domains where semantics or visual attributes evolve, such as historical portraits [11] or online platforms.

In practice, these forms of drift rarely occur in isolation and often manifest in combination [1]. As a result, the temporal robustness of a model depends not only on the magnitude of drift but also on how its architectural assumptions and inductive biases interact with the evolving data distribution.

2.2 Model Robustness to Distribution Shift

Understanding how learning algorithms behave under distribution shift has become an important research direction. Early work approached robustness through domain adaptation [2] and out-of-distribution generalization [12], formalizing shift as a transition between a small number of discrete source and target domains.

The introduction of large-scale benchmarks such as WILDS [20], which focuses on naturally occurring distribution shifts without explicit temporal indexing, and more recently Wild-Time [33], which explicitly models time-indexed data and temporal distribution shifts, has broadened this perspective by providing evaluation protocols that more closely reflect real-world deployment scenarios.

A complementary line of research has examined robustness through model diagnostics and monitoring. Methods for detecting drift onset by examining changes in latent representations or predictive uncertainty have shown promise in deep learning settings [25]. Recent empirical analyses have characterized how concept drift manifests in practice, including its locality and temporal progression across large-scale data streams [1]. At the systems level, work on data-centric and continual-learning infrastructures has explored how to maintain model quality over time through cost-aware retraining and pipeline orchestration [23,3].

Despite these advances, robustness studies commonly focus either on a single architecture evaluated across multiple datasets or on a single dataset used to compare a narrow set of architectures. As a result, comparatively little is known about how architectural design choices interact with long-range temporal drift across heterogeneous modalities and tasks. This gap is particularly relevant given the diversity of inductive biases exhibited by modern neural models: convolutional networks encode locality and translation equivariance, recurrent networks capture sequential structure [16,4], and Transformer-based encoders rely on self-attention with minimal structural priors [30]. Likewise, large-scale pretraining in vision and language [7,21] introduces representations shaped by broad historical data, yet their behavior under extended temporal drift remains poorly characterized.

The present study addresses this open question by comparing these architectural families under a shared temporal evaluation protocol and across multiple modalities, enabling a controlled analysis of how model design influences robustness under real-world temporal distribution shift.

2.3 Robustness Evaluation vs. Temporal Adaptation

This work evaluates the inherent temporal robustness of neural architectures under distribution shift, measuring how different model families, trained only on cumulative historical data, perform as the temporal gap between training and evaluation widens. This isolates the robustness arising from architectural design and pretraining alone, prior to any adaptive intervention.

A complementary line of research instead investigates how models can *adapt* once drift is detected, through scheduled or cost-aware retraining [23] and continual-learning pipelines [3], addressing when to retrain, how much data to incorporate, and how to trade performance against cost. Our study is orthogonal to that literature and provides a foundation on which such adaptation strategies can be designed, evaluated, and compared.

3 Methods and Experimental Approach

Temporal distribution shift manifests differently across modalities, tasks, and time scales, yet existing empirical studies typically vary a single axis at a time, leaving open how *architectural design*, *label structure*, and *drift mechanism* jointly shape temporal robustness (Section 2). Our setup targets exactly this cross-cutting comparison: we combine three long-range, time-indexed datasets with a diverse suite of neural architectures spanning multiple inductive biases, all evaluated under the unified temporal protocol of Section 4 (temporal drift matrices), so that architectural differences, rather than differences in splitting or evaluation, drive the observed robustness patterns.

3.1 Datasets

Our empirical analysis spans three qualitatively distinct temporal scenarios, chosen to cover different modalities, tasks, and sources of distribution shift. Each dataset captures multiple decades of real-world temporal evolution, making it suitable for cross-temporal evaluation.

Yearbook: A Century of Portraits The Yearbook dataset [11] contains 37,921 frontal portraits of American high-school seniors from 1905 to 2013, which we align and process as 3-channel 32×32 tensors. Although acquisition is largely standardized, stylistic attributes (e.g., hairstyles, clothing, accessories) vary substantially across decades. The dataset provides binary sex labels that are approximately balanced over time. It is a canonical benchmark for temporal shift: Wild-Time [33] uses a pre-/post-1970 split and reports marked out-of-distribution degradation, and Modyn [3] observes accuracy decay as the train–test time gap grows. We reproduce this trend (Fig. 2), consistent with covariate and concept drift.

Amazon Reviews 2023: E-commerce The Amazon Reviews dataset [17] comprises 571.54 million reviews across 33 product categories, spanning May 1996 to September 2023. Each review has a timestamp, star rating, and free-text content. We cast a review-level sentiment regression task, predicting the 1–5 rating from the text, which exhibits strong covariate and concept drift due to evolving language, consumer behavior, and platform usage. We focus on seven categories, restrict to 2014–2023, and draw a stratified sample of 300,000 reviews.

arXiv: Scientific Discourse The arXiv dataset [6] defines a multi-label task over 2,866,787 title–abstract records annotated with 176 subject categories. We concatenate titles and abstracts, keep seven leaf categories (Section E), and use 2000–2025 submissions whose categories fall within them (papers may carry several). Category mix and terminology shift, inducing label and covariate drift. Wild-Time [33] reports roughly 20% degradation under temporal splits for a related arXiv task, and the gap is not substantially closed by domain generalization or continual learning methods, motivating our architectural comparison.

3.2 Model Implementation

We evaluate architectures spanning different inductive biases to understand how model design affects temporal robustness. Rather than covering the full architecture landscape, we select families that span the spectrum of inductive-bias strength: from simple baselines without structural assumptions that establish lower bounds on performance, to modern architectures that incorporate structural priors, to pre-trained models that leverage large-scale external data. This spread is what lets us attribute robustness differences to the priors themselves.

Image Classification For the **Yearbook** dataset, we evaluate four model families trained on the data, each at three sizes (small, medium, large), for 12 architectures, complemented by 9 pretrained vision encoders used as frozen backbones.

The four families differ in the spatial prior they encode. At one extreme, the *multilayer perceptron* (MLP) flattens the image and applies only fully connected layers, imposing no spatial structure. The *convolutional network* (CNN) builds in locality and translation equivariance through convolutional filters with batch normalization and pooling, and the *residual network* (ResNet) extends it with skip connections [14] that ease optimization at greater depth. At the other extreme, the *vision Transformer* (ViT) drops convolution for self-attention over patches [8], a far weaker spatial prior, with a small patch size suited to the 32×32 inputs. Across all four, the small, medium, and large variants scale depth and width.

Finally, we assess *transfer learning* using pretrained vision encoders trained on large-scale curated or web-scale datasets. The underlying hypothesis is that representations learned from diverse data may exhibit stronger temporal robustness than features learned solely from historical portraits. We consider 9 pretrained models. The self-supervised DINOv2-S [24] and DINOv3-S [28] are pretrained on large curated image collections; CLIP-B32 [26] and SigLIP-B [34] use contrastive image-text pretraining, the latter with a sigmoid loss; ConvNeXt-S [22], ResNet50-IN [14], and ViT-S16-IN21k [8] are trained with supervised ImageNet labels, the last on ImageNet-21k; and MAE-B [13] and EVA02-B [9] are pretrained with masked image modeling. In all cases, we freeze the pretrained backbone and train only a linear classification head, isolating the contribution of pretrained representations from the effects of fine-tuning dynamics.

Text Models For the **Amazon Reviews** and **arXiv** datasets, we evaluate model families that differ in their inductive biases for representing textual structure. The same architectures serve both datasets, differing only in output layer and loss, with **Amazon Reviews** using regression under a weighted mean squared error and **arXiv** multi-label classification under a weighted binary cross-entropy.

All four families operate on cached RoBERTa token embeddings. The *feed-forward baseline* (FFN) averages them into a single 768-dimensional vector and passes it through a small MLP head that discards order entirely, with variants scaling the hidden width from 128 to 2048. The *convolutional model* (TextCNN) [18] applies one-dimensional convolutions to capture local n-gram patterns, scaling

the number and width of its filters. The *recurrent* models read the sequence token by token, as a single-layer bidirectional GRU [4], a single-layer bidirectional LSTM [16], and a two-layer bidirectional LSTM with attention. The *Transformer* encoders replace recurrence with self-attention [30] and learnable positional embeddings, ranging from 1 to 5 layers and 4 to 6 heads.

Finally, we assess transfer learning from pretrained language encoders. We consider frozen encoders with a light head trained on the pooled output: BERT [7], RoBERTa [21], DeBERTa-v3 [15], ELECTRA [5], MPNet [29], ModernBERT [32], and DistilBERT [27] on both text tasks, plus MiniLM-L6 [31] on arXiv. As with the vision encoders, freezing isolates pretrained representations from fine-tuning.

4 Evaluation Framework

Standard held-out evaluation measures performance on data from the training distribution, which under temporal shift can look strong even as the model fails on future data. We therefore measure cross-temporal generalization explicitly.

4.1 Temporal Drift Matrices

Let $\mathcal{D} = \{(x_i, y_i, t_i)\}_{i=1}^N$ denote a dataset where each example is associated with a timestamp t_i . We divide the timeline into K disjoint intervals $\mathcal{T} = \{T_1, \dots, T_K\}$, where $T_k = [t_k^{\text{start}}, t_k^{\text{end}})$. Let $\mathcal{D}_k = \{(x, y, t) \in \mathcal{D} : t \in T_k\}$ denote the subset of data from interval k .

For training, we construct cumulative datasets $\mathcal{D}_{\leq k} = \bigcup_{j=1}^k \mathcal{D}_j$ containing all data up to and including interval k . A model f_k trained on $\mathcal{D}_{\leq k}$ has access to historical data over time t_k^{end} but no knowledge of future periods. This cumulative strategy reflects realistic deployment scenarios in which models are periodically retrained on all historical data.

The *temporal drift matrix* $M \in \mathbb{R}^{K \times K}$ captures cross-temporal generalization:

$$M_{ij} = \text{perf}(f_i, \mathcal{D}_j) \quad (1)$$

where $\text{perf}(\cdot, \cdot)$ denotes a performance metric (accuracy, macro AUC, or balanced MSE depending on the task). Entry M_{ij} measures how well a model trained on data through period i performs on data from period j .

The structure of M reveals aspects of temporal robustness. Following our plotting convention, rows are the training cutoff (“trained up to”) and columns the evaluation period (“evaluated on”), with time increasing upward and to the right from a lower-left origin. The diagonal M_{ii} is in-distribution performance on data held out from the training period; the upper-left ($j < i$) is held-out performance on earlier in-training periods; and the lower-right ($j > i$) is forward generalization to data unseen during training. This lets us read how performance degrades (or improves) as the temporal gap between training and evaluation widens.

4.2 Temporal Splitting Strategy

Each slice $\mathcal{D}_k$ is partitioned once into a stratified training split $\mathcal{D}_k^{\text{train}}$ (70%) and a held-out test split $\mathcal{D}_k^{\text{test}}$ (30%), shared across all models via a fixed split seed. For in-distribution evaluation (when $j \leq i$), we use only the held-out test split of $\mathcal{D}_j$ to prevent data leakage:

$$\mathcal{D}_{\text{eval}}^{(i,j)} = \begin{cases} \mathcal{D}_j^{\text{test}} & \text{if } j \leq i \text{ (in-distribution)} \\ \mathcal{D}_j^{\text{train}} \cup \mathcal{D}_j^{\text{test}} & \text{if } j > i \text{ (out-of-distrib.)} \end{cases}$$

For out-of-distribution evaluation on future time slices, we use all available samples from that period to maximize statistical power, since by definition none of this data was seen during training. Evaluating on periods that overlap the training data ($j \leq i$) is deliberate: these held-out entries verify that a model retains performance across the historical periods it was trained on and provide the in-distribution reference from which forward decay is measured.

4.3 Training Protocol

The image models are trained with Adam [19] and the text models with its decoupled-weight-decay variant AdamW, a standard choice that behaves robustly across heterogeneous architectures; fixing one optimizer per modality avoids per-model optimizer tuning as a confound. Learning rates are task-specific, with exact hyperparameters pinned in versioned experiment presets. Training proceeds for a fixed number of epochs, and model selection is based on the final checkpoint. Every configuration is trained under multiple random seeds, five on **Yearbook** and three on the text tasks (Section A), and all results average over seeds.

We adopt a cumulative temporal training strategy: for each slice T_k we train a model f_k on $\mathcal{D}_{\leq k}$, all examples observed up to T_k , yielding a sequence $\{f_1, \dots, f_K\}$ that each represent the best model obtainable from the data available at that point in time. All checkpoints are stored and evaluated on every slice to construct the full temporal drift matrix M .

The loss follows the task structure and is tailored to address label imbalance.

For *binary classification* on **Yearbook**, we minimize standard two-class cross-entropy on logits and labels, the maximum-likelihood objective for a two-class softmax model.

For *multi-label classification* on **arXiv**, where each paper’s categories all lie within the seven leaf categories and negatives greatly outnumber positives, we use a class-weighted binary cross-entropy with logits over the $C = 7$ categories. Each class c is weighted by its negative-to-positive ratio $w_c = |\{i : y_{ic} = 0\}| / |\{i : y_{ic} = 1\}|$, which amplifies rare positives so the all-zero vector cannot score well.

For *regression* on **Amazon Reviews**, we predict the star rating $y_i \in \{1, \dots, 5\}$ with an inverse-frequency-weighted mean squared error, weighting each rating r by $w_r = n/n_r$ (n total samples, n_r with rating r). Because the rating distribution is skewed toward high scores, this keeps errors on rare low ratings visible, so temporal degradation cannot be explained by rating prevalence alone.

4.4 Evaluation Metrics

To populate each entry of the drift matrix M , we require a scalar performance metric per train-test time pair. We match each metric to the task’s label structure: accuracy for **Yearbook**, whose binary labels are approximately balanced (Section B.1); balanced MSE (MSE_{bal}) for **Amazon Reviews**, which reweights its skewed rating distribution (Section B.2); and macro AUC ($\overline{\text{AUC}}$) for **arXiv**, whose imbalanced multi-label categories require a threshold-free, per-class average (Section B.3). The two classification tasks thus receive different metrics because their label structures differ; the drift protocol itself is identical.

4.5 Summary Statistics of the Drift Matrix

In the drift matrix M , the row index i is the training cutoff and the column index j is the evaluation period, so the entry M_{ij} is the performance of a model trained on all data up to period i when it is tested on period j . Three numbers summarize the matrix, each averaged only over filled cells (a train-test pair with no completed run leaves its cell empty): the *in-distribution* score $\text{ID}(M) = \text{mean}_i M_{ii}$, a model’s performance on the period it was trained through; the *future* score $\text{Fut}(M) = \text{mean}_{j>i} M_{ij}$, averaging the cells whose evaluation period falls after the training cutoff; and the *decay* $\text{Dec}(M) = \text{ID}(M) - \text{Fut}(M)$ (the two terms swapped for MSE_{bal} , which is better when low), oriented so a larger value always means worse temporal robustness. A positive decay is a loss on future periods and a negative one, uncommon, a gain; together, future score and decay order every model from most to least temporally robust.

Beyond these whole-matrix scores, we look at how robustness depends on the training time itself. Fixing one cutoff i , a single row of the matrix, we pair its diagonal cell M_{ii} with the average over the later periods in that row ($\text{mean}_{j>i} M_{ij}$) and the decay between them, reading these at a few cutoffs spread evenly across the timeline (the last one is left out, as it has no future). Averaging the same quantities within each architecture family lets us compare the families directly.

To place each model against the others, we average the matrices over the cohort $\mathcal{C}$ into the *cohort-mean matrix* $\bar{M}_{ij} = |\mathcal{C}|^{-1} \sum_{m \in \mathcal{C}} M_{ij}^{(m)}$, defined at the cells every model fills; each model’s *deviation* is $\Delta_{ij}^{(m)} = M_{ij}^{(m)} - \bar{M}_{ij}$.

5 Results

Each model is summarised through its drift matrix by the in-distribution, forward, and decay scores of Section 4.5, namely its performance on the period it was trained through, its mean performance on the later periods held out from training, and the difference between the two. How far a model degrades between them is governed by two factors that recur across the three domains, the strength of its inductive bias and whether it relies on a frozen pretrained encoder.

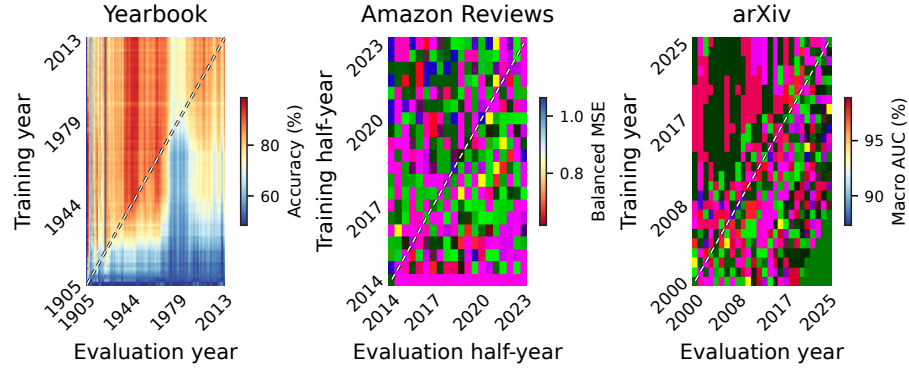


Fig. 2: Cohort-mean drift matrices, one panel per domain. Each cell (i, j) is mean performance across models when trained through period i (row) and evaluated on period j (column): accuracy for **Yearbook**, balanced MSE for **Amazon Reviews** (lower is better), and macro AUC for **arXiv**. The dashed diagonal marks in-distribution evaluation; the lower-right region is forward generalization. Vertical white bands mark periods with insufficient samples.

5.1 Image Classification

On **Yearbook**, the model families separate sharply in distribution, along the diagonal of the cohort-mean matrix in the **Yearbook** panel of Figure 2. MLP-S, which flattens the image into a vector and imposes no spatial structure, sits at about 69.3%, while the CNNs and ResNets reach near 92.8% for CNN-M and CNN-L (Table 5). What sets these apart is their inductive bias toward locality and translation equivariance, by which they build features from small local regions of the image and recognise a pattern wherever it appears, and this lets them exploit the cues that most sharply separate male from female portraits within a given period. The ViTs and frozen encoders fall between these two ends.

Temporal robustness reverses this ordering, and we read it from the decay, the accuracy a model loses once the evaluation year moves past its training cutoff (Table 5). The CNNs and ResNets, strongest in distribution, decay the most, by 13.7 and 13.9 points for CNN-L and CNN-M, so their accuracy on future years falls to about 79.0%. MLP-S is the steadiest model of all, with only 7.1 points of decay, though from its lower starting accuracy of 62.3%. The frozen pretrained encoders form a third group, more stable than the CNNs but less accurate to begin with, decaying by 7.9 to 10.4 points, with the self-supervised DINOv3-S steadiest among them and the supervised ImageNet-21k backbone least.

What lifts a model in distribution is also what undermines it over time. The features a strong inductive bias extracts to separate the classes **raise its in-distribution accuracy**, but they are **the most specific to the years it was trained on, and the first to lose their meaning** once hairstyles, clothing, and image quality drift. The sharper a model’s in-distribution lead, the faster it erodes as the data ages. Robustness on **Yearbook** is therefore not a property an

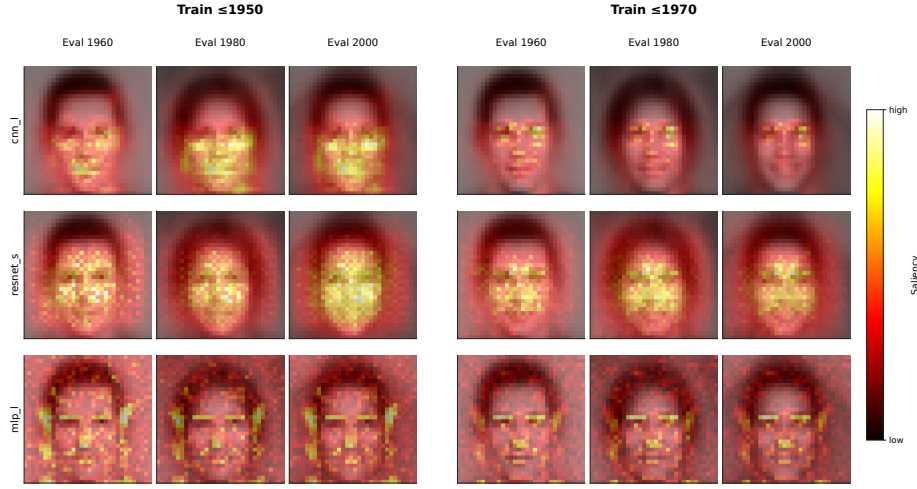


Fig. 3: Gradient saliency maps on *Yearbook* for CNN-L, ResNet-S, and MLP-L, each trained through 1950 (left) and 1970 (right) and evaluated on later years. Each panel averages gradient saliency over the selected portraits from the indicated evaluation year rather than a single example.

architecture optimises on its own, but the other side of fitting a single period too closely, closer to overfitting across time than across samples (Section C).

The forgetting curves in Figure 4 show this playing out year by year. Each curve plots a model’s accuracy against the gap between its training cutoff and the evaluation year, so reading from left to right traces how quickly it forgets (Section C.7). The strongly biased networks start far above the rest, near 90% at a zero gap, and their curves descend the steepest. As the gap widens the curves draw together and then cross, so the families that led in distribution lose their edge on the most distant years, and every model settles near two-class chance.

Saliency Maps The gradient saliency maps in Figure 3 show where this fragility comes from. Extending the training window from 1950 to 1970 sharpens where the convolutional models look: the CNN tightens from a diffuse scatter over the cheeks and mouth to a compact, near-symmetric pair of bright spots at the eyes, the detail that most sharply tells the classes apart, and the ResNet shifts the same way while keeping a broader spread across the central face. The MLP, without that bias, spreads its attribution across the whole frame at either cutoff, the background included. The same precision that lets the convolutional models read this discriminative detail binds them to it: those localized features are the most specific to the training period and the most exposed to drift, while the MLP’s blunter reading is less tied to any era.

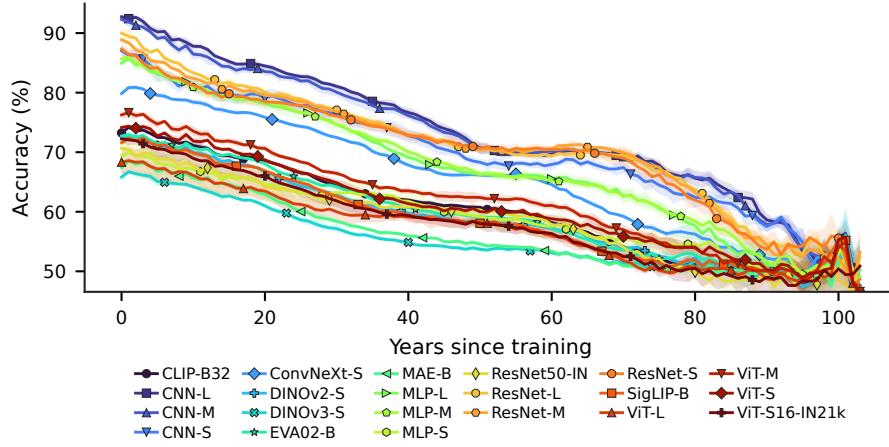


Fig. 4: Forgetting curves on **Yearbook**. Each curve is one model’s mean accuracy as the gap between its training year and the evaluation year grows, averaged over training years and seeds. A zero gap is in-distribution, and the slope is the rate of forgetting (Section C.7).

5.2 Text Models

Each text model is a light head trained on cached, frozen RoBERTa embeddings (Section 3), a shared representation over which the families differ only in inductive bias. **Amazon Reviews** is a rating regression scored by balanced MSE, where lower is better, and **arXiv** is a multi-label classification scored by macro AUC.

Amazon Reviews On **Amazon Reviews** the recurrent and Transformer models fit the training reviews most closely, since both read the wording in context, the recurrent networks token by token and the Transformers through self-attention, and so capture how a review’s words compose into its rating. That fit shows up as the lowest in-distribution error, down to 0.680 balanced MSE for BiGRU-S, with the Transformers alongside them at 0.686 for TX-S (Figure 2, Table 13). The FFN, which averages the embeddings and discards word order, sits higher at about 0.761, and the frozen encoders higher still, between 0.806 and 1.022.

Temporal robustness runs the other way, and the models that fit the reviews most tightly are the ones whose error grows fastest. The recurrent networks decay the most, by up to 0.128 balanced MSE, so their error on future reviews climbs to about 0.824, while the FFN is the steadiest of the trained models, gaining only 0.075 to 0.089. The decay is sharpest for models trained on the earliest reviews, where the recurrent family worsens by 0.242, and it narrows toward the recent cutoffs as fewer years remain ahead (Section D). The frozen encoders again sit at higher error but drift less than the trained models, and DeBERTa-v3 is the most stable model anywhere in the study, at 0.043 of decay.

What makes these models fit the reviews so well is also what later undermines them. The way they compose the wording into a rating is the most specific to how reviews were written in the training years, and the first to lose its meaning as the language shifts. **Far enough forward the in-distribution ranking inverts, and the models that fit the reviews most tightly end among the least accurate**, while the order-free FFN, never sharp, stays the steadiest.

arXiv On **arXiv** a stronger inductive bias yields no in-distribution advantage, and so costs no temporal robustness. Sorting a paper into its subject categories is close to recognising its topic, and the topic is already encoded in the frozen RoBERTa embeddings every model is built on, so no architecture finds structure the others miss. The feed-forward, convolutional, recurrent, and Transformer families therefore reach almost the same in-distribution macro AUC, between about 97.2% and 98.1%, within a point of one another (Figure 2, Table 21).

Robustness is just as uniform, and each family loses a similar small 2.7 to 3.4 points going forward, with the frozen encoders in the same band apart from DeBERTa-v3 and ELECTRA, which fall further at 6.5 and 7.3 points (Section E). **Where the bias gains nothing in distribution it forms no period-specific features to lose, so no family decays faster than the rest.** The later years stay close to the training distribution: the backbone learned this vocabulary before any model saw it, leaving temporal shift little to take away.

6 Conclusion and Future Work

Across image classification, text regression, and multi-label text classification, in-distribution accuracy turns out to neither guarantee nor predict temporal robustness. What governs the rate of degradation is instead the strength of a model’s inductive bias and whether it relies on a frozen pretrained encoder. A stronger bias extracts the most discriminative, period-specific features and leads in distribution, yet those same features are the first to lose their meaning as the data drifts, so the architectures that fit the training period most tightly are the ones that degrade fastest. Frozen pretrained encoders occupy a different regime, conceding in-distribution accuracy in return for steadier behaviour over time, and on **arXiv**, where the task is already solved by the pretrained representation and the bias buys no advantage, no family decays faster than the rest. The practical reading is that the model with the best held-out score at training time is often the least robust once deployed, so architecture selection should weigh the expected horizon between retraining cycles, not in-distribution accuracy alone.

6.1 Limitations

Our comparison covers the most common inductive biases, convolutional, recurrent, and attention-based, against simple baselines and frozen encoders, but many architectures remain untested and the study is best read as a first systematic pass rather than an exhaustive one. It spans three datasets across two modalities, and

broadening it to further domains and other forms of distribution shift would show how widely the pattern holds. The text tasks are the most constrained, using a narrow label space, a five-point rating for **Amazon Reviews** and seven categories for **arXiv**, spanning only about a decade and a quarter of a century against a full century for **Yearbook**, and running on a stratified subsample rather than the complete corpora. Because too few examples fall within each time slice to train a text encoder from scratch, the text models also share a single frozen **RoBERTa** representation, so their absolute scores should be read in that light. We average over a small fixed seed set rather than conducting formal significance tests, and hyperparameter choices and seed variance may still affect fine-grained rankings; we therefore emphasize family-level patterns over individual placements. The families are also not capacity-matched. The small, medium, and large variants within each family expose scale effects, and on **Yearbook** the smallest MLP, CNN, and **ResNet** variants decay less than their larger siblings, so inductive bias and capacity remain partially confounded. The analysis is also descriptive rather than mechanistic. We measure how robustness varies with inductive bias and pretraining without isolating the precise cues responsible, and we evaluate inherent robustness under cumulative training rather than any adaptive policy, which leaves the question of retraining orchestration to systems such as **Modyn** [3].

6.2 Future Work

These limitations point to several extensions. The comparison can be widened to a broader range of inductive biases, to capacity-matched pairs that disentangle bias from scale, and to additional datasets, modalities, and drift mechanisms, and carried to longer temporal spans with the full corpora and richer label spaces, where language drift should be more pronounced and the gap between architectures wider. The drift matrices can also be paired with scheduled or drift-triggered retraining under explicit compute budgets, measuring not only how much a model decays but how cheaply that decay can be undone [23]. Most of all, moving from description to mechanism, by tying inductive bias and pretraining back to the specific features that drift, would turn the observed regularity into an account of why temporal robustness behaves as it does.

6.3 Availability

Code, experiment definitions, and paper-generation scripts are available at github.com/learning-mechanisms/drift-happens; public run histories and matrix artifacts at wandb.ai/drift-happens/drift-happens. The [extended preprint](#) and drift-happens.org retain the complete per-model drift-matrix galleries. Code is Apache-2.0; paper, figures, and documentation are CC BY 4.0 where we hold the rights, and external datasets and models keep their original licenses.

Disclosure of Interests. The authors have no competing interests to declare.

References

1. Aguiar, G.J., Cano, A.: A comprehensive analysis of concept drift locality in data streams (2023)
2. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. *Machine Learning* **79**(1), 151–175 (2010)
3. Böther, M., Robroek, T., Gsteiger, V., Holzinger, R., Ma, X., Tözün, P., Klimovic, A.: Modyn: Data-centric machine learning pipeline orchestration. *Proc. ACM Manag. Data* **3**(1) (Feb 2025)
4. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pp. 1724–1734. Association for Computational Linguistics (2014)
5. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: ELECTRA: Pre-training text encoders as discriminators rather than generators. In: *International Conference on Learning Representations (ICLR)* (2020)
6. Cornell University: arxiv dataset (2024)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. pp. 4171–4186. Association for Computational Linguistics (2019)
8. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: *International Conference on Learning Representations (ICLR)* (2021)
9. Fang, Y., Sun, Q., Wang, X., Huang, T., Wang, X., Cao, Y.: EVA-02: A visual representation for neon genesis. *Image and Vision Computing* **149**, 105171 (2024)
10. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM Computing Surveys* **46**(4), 1–37 (2014)
11. Ginosar, S., Rakelly, K., Sachs, S., Yin, B., Efros, A.A.: A century of portraits: A visual historical record of american high school yearbooks. In: *IEEE International Conference on Computer Vision Workshops*. pp. 1–7 (2015)
12. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: *International Conference on Learning Representations* (2021)
13. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)
15. He, P., Gao, J., Chen, W.: DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In: *International Conference on Learning Representations (ICLR)* (2023)
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Computation* **9**(8), 1735–1780 (1997)
17. Hou, Y., Li, J., He, Z., Yan, A., Chen, X., McAuley, J.: Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952* (2024)

18. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751. Association for Computational Linguistics (2014)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015)
20. Koh, P.W., Sagawa, S., Marklund, H., Xie, S.M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., et al.: WILDS: A benchmark of in-the-wild distribution shifts. In: International Conference on Machine Learning. pp. 5637–5664. PMLR (2021)
21. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., et al.: RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
22. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
23. Mahadevan, A., Mathioudakis, M.: Cost-aware retraining for machine learning. *Knowledge-Based Systems* **293**, 111610 (2024)
24. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., et al.: DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)* (2024)
25. Rabanser, S., Günnemann, S., Lipton, Z.: Failing loudly: An empirical study of methods for detecting dataset shift. In: *Advances in Neural Information Processing Systems*. vol. 32 (2019)
26. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of the 38th International Conference on Machine Learning (ICML) (2021)
27. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
28. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., et al.: DINOv3. arXiv preprint arXiv:2508.10104 (2025)
29. Song, K., Tan, X., Qin, T., Lu, J., Liu, T.Y.: MPNet: Masked and permuted pre-training for language understanding. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2020)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: *Advances in Neural Information Processing Systems*. vol. 30. Curran Associates, Inc. (2017)
31. Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., Zhou, M.: MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2020)
32. Warner, B., Chaffin, A., Clavié, B., Weller, O., Hallström, O., Taghadouini, S., Gallagher, A., Biswas, R., et al.: Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. arXiv preprint arXiv:2412.13663 (2024)
33. Yao, H., Choi, C., Cao, B., Lee, Y., Koh, P.W., Finn, C.: Wild-time: A benchmark of in-the-wild distribution shift over time. In: *Advances in Neural Information Processing Systems* (2022)
34. Zhai, X., Mustafa, B., Kolesnikov, A., Beyer, L.: Sigmoid loss for language image pre-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)

A Reproducibility

Experiments are defined as versioned preset snapshots and run through the repository command-line interface in a Pixi-pinned environment; run metadata records the git state and `pixi.lock` hash, and regenerated paper and website assets are checked against a committed checksum manifest. Each experiment is a staged run for one dataset, architecture, and seed: the training stage fits all cumulative temporal checkpoints, and the evaluation stage scores them on every time slice to form the drift matrix. We use seeds 0–4 for **Yearbook** and 0–2 for **Amazon Reviews** and **arXiv**, averaging over them. Public run histories and per-run matrix artifacts are available in the Weights & Biases project at <https://wandb.ai/drift-happens/drift-happens>.

Table 1 summarizes the measured compute on NVIDIA A100 GPUs, summing the `W&B_runtime` of each finished train and evaluation stage. GPU-hours are successful-stage wall times under one GPU per run; the four-GPU node-hour column divides these by four for ideal packed execution, excluding queueing, data preprocessing, failed attempts, and scheduling idle time.

Table 1: Approximate measured compute for the conference experiment campaign.

Dataset	Model-seed configs	Train GPU-h	Eval GPU-h	4-GPU node-h
Yearbook	105	23.7	9.3	8.2
Amazon Reviews	57	116.6	136.7	63.3
arXiv	60	272.3	101.9	93.5
Total	222	412.6	247.9	165.1

B Per-Dataset Protocol and Metrics

The drift-matrix protocol of Section 4 is shared across the three datasets, but each instantiates it at its own time granularity and is scored with the metric matched to its label structure (Table 2). For each task we require a scalar performance measure that is both aligned with the task and robust to the class imbalance present in that dataset; a metric dominated by the label distribution would conflate shifts in the data with changes in the model.

Table 2: Per-dataset instantiation of the drift-matrix protocol: time span, slice granularity, number of slices K , and primary metric.

Dataset	Span	Slice	K	Metric
Yearbook	1905–2013	yearly	104	accuracy
Amazon Reviews	2014–2023	half-yearly	20	balanced MSE
arXiv	2000–2025	yearly	26	macro AUC

B.1 Yearbook

We slice the 1905–2013 timeline into one-year intervals, keeping the $K = 104$ years with enough samples. The task is binary classification with roughly balanced classes, so we score it with standard accuracy, the fraction of portraits classified correctly. Balance is what makes accuracy trustworthy here: when neither class dominates, a model cannot earn a good score by always predicting the same label, so accuracy rises and falls only as the model genuinely classifies more or fewer cases correctly. Higher values are better.

B.2 Amazon Reviews

We slice the 2014–2023 timeline into half-year intervals, giving $K = 20$ slices. The task is regression: the model predicts a star rating from one to five and is scored by squared error. The difficulty is that the ratings are strongly skewed toward five-star reviews, so a single mean squared error taken over all reviews would mainly measure performance on the majority and would barely react to the rarer low ratings. To keep every rating level visible, we first compute the mean squared error separately within each rating and then average those per-rating values equally over the rating levels present,

$$\text{MSE}_{\text{bal}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \text{MSE}_r, \quad \text{MSE}_r = \frac{1}{|\mathcal{S}_r|} \sum_{i \in \mathcal{S}_r} (y_i - \hat{y}_i)^2,$$

where $\mathcal{S}_r = \{i : y_i = r\}$ is the set of reviews whose true rating is r and $\mathcal{R} = \{r : |\mathcal{S}_r| > 0\}$ is the set of rating levels present in the slice (at most the five levels one to five). Because every present level contributes equally, a model that began to drift on the scarce one- and two-star reviews would reveal it here. Unlike the other two metrics, lower values are better.

B.3 arXiv

We slice the 2000–2025 timeline into one-year intervals, giving $K = 26$ slices. The task is multi-label: a paper can belong to several of the $C = 7$ subject areas at once, and those areas are very unevenly populated. This raises two problems. First, fixing a single decision threshold is arbitrary and tends to favour the frequent subjects, so a threshold-based score such as accuracy is misleading. Second, an average that weights papers equally is again dominated by the common subjects. We address the first by scoring each subject with the Area Under the ROC Curve, which summarises performance over all thresholds at once: AUC_c is the probability that a randomly chosen paper carrying subject c is given a higher score for that subject than a randomly chosen paper that does not carry it. We address the second by averaging the per-subject values uniformly, so each subject counts the same regardless of how many papers it contains,

$$\overline{\text{AUC}} = \frac{1}{C} \sum_{c=1}^C \text{AUC}_c, \quad \text{AUC}_c = \int_0^1 \text{TPR}_c(t) d\text{FPR}_c(t),$$

where $\text{TPR}_c(t)$ and $\text{FPR}_c(t)$ denote the true- and false-positive rates for subject c at threshold t . Higher values are better, and a model that ranked every paper correctly within every subject would reach 1.

C Yearbook – Drift Matrices

The cohort-mean and per-model deviation matrices shown here, and the in-distribution, future, and decay quantities tabulated below, are defined in Section 4.5.

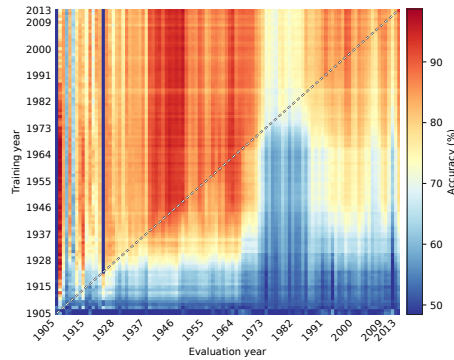


Fig. 5: Cohort-mean Accuracy matrix $\bar{M}$ over the Yearbook models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j .

C.1 Model Roster

Table 3: Yearbook: models trained from scratch.

Model	Family	Parameters
ViT-S	ViT	75k
CNN-S	CNN	94k
ResNet-S	ResNet	98k
MLP-S	MLP	98k
ResNet-M	ResNet	400k
MLP-M	MLP	410k
CNN-M	CNN	542k
ViT-M	ViT	545k
MLP-L	MLP	2.1M
ResNet-L	ResNet	2.1M
CNN-L	CNN	2.1M
ViT-L	ViT	2.2M

Table 4: Yearbook: frozen pretrained encoders, with trainable head and total parameters.

Model	Family	Trainable	Total
DINOv3-S	Transfer	770	21.6M
ViT-S16-IN21k	Transfer	770	21.7M
DINOv2-S	Transfer	770	22.1M
ResNet50-IN	Transfer	4k	23.5M
ConvNeXt-S	Transfer	2k	49.5M
EVA02-B	Transfer	2k	85.8M
MAE-B	Transfer	2k	85.8M
CLIP-B32	Transfer	2k	87.5M
SigLIP-B	Transfer	2k	92.9M

C.2 MLP

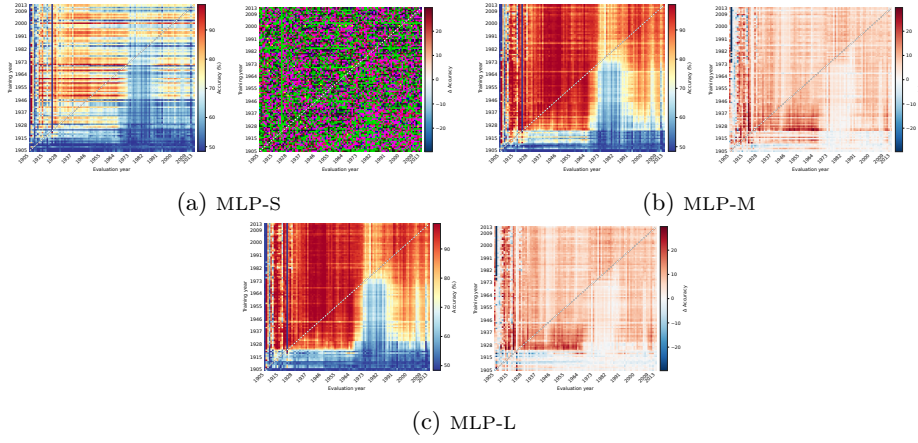


Fig. 6: MLP models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

C.3 CNN

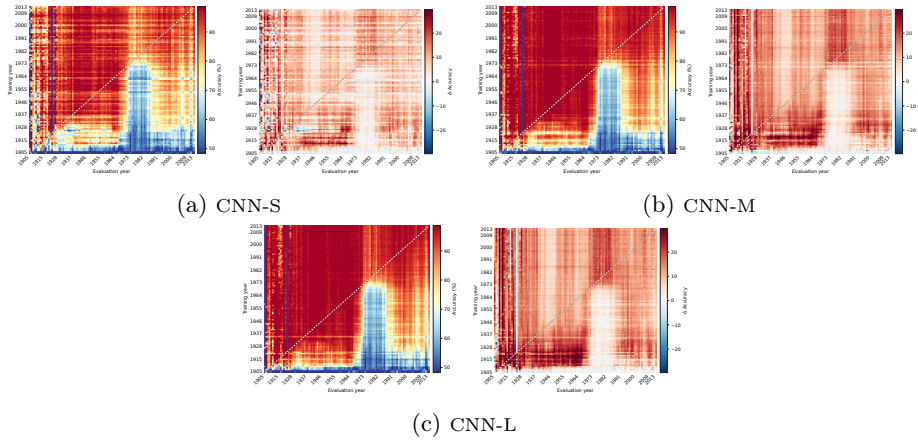


Fig. 7: CNN models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

C.4 ResNet

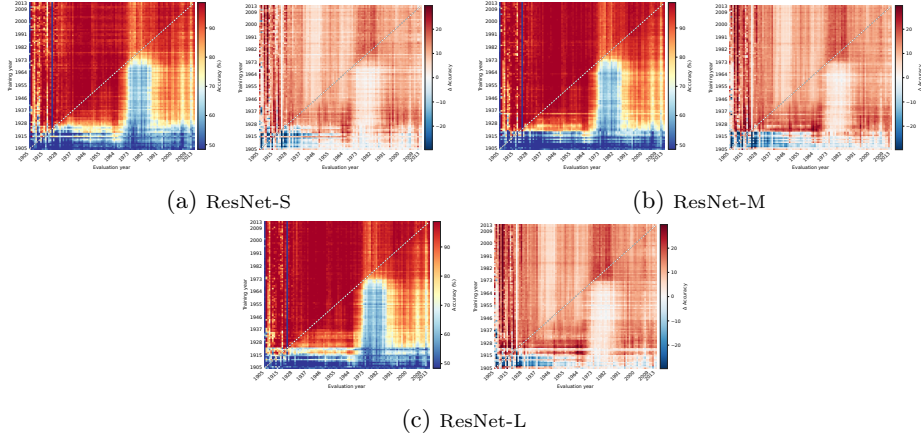


Fig. 8: ResNet models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

C.5 ViT

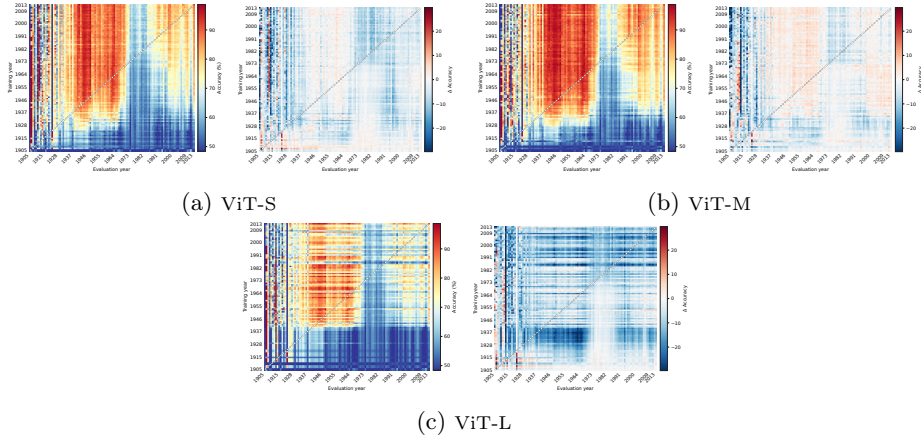


Fig. 9: ViT models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

C.6 Transfer

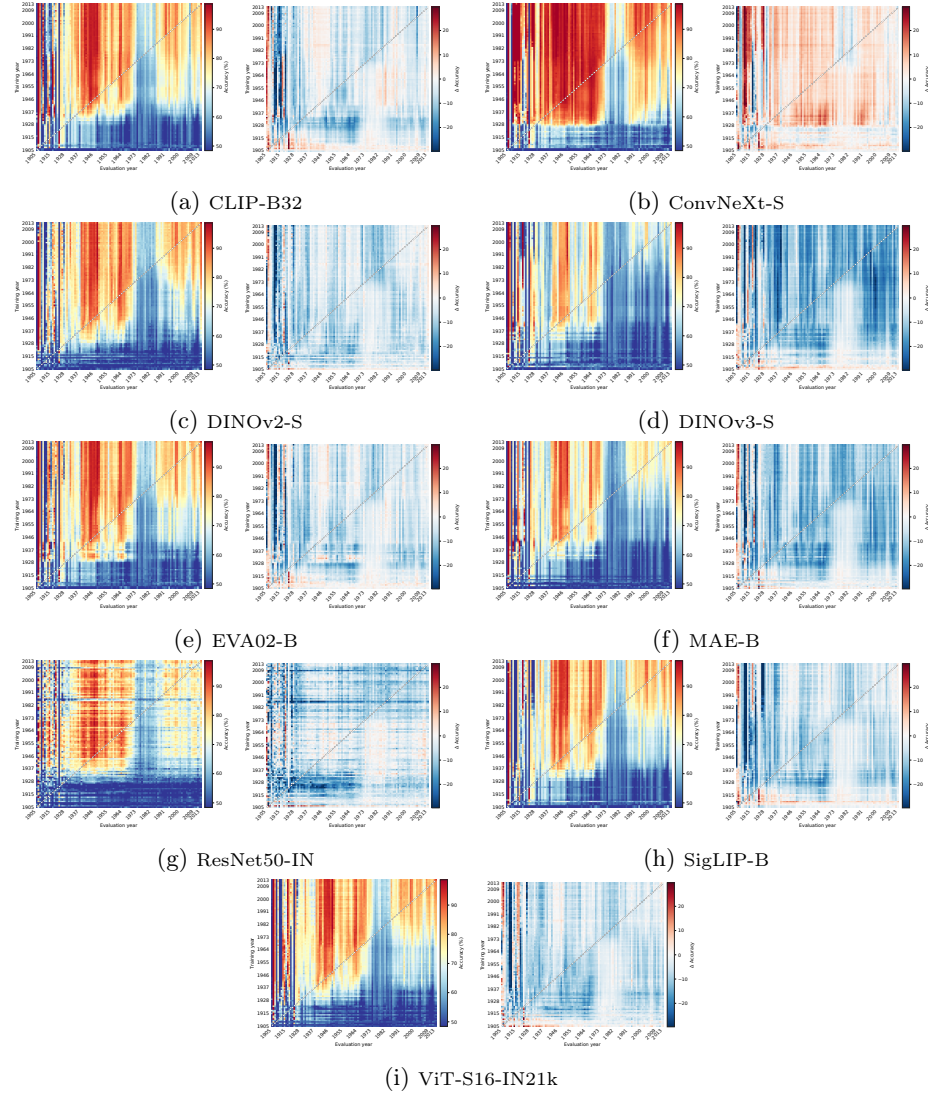


Fig. 10: Transfer models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

C.7 Forgetting and Rankings

To see how quickly each model forgets, we summarize its drift matrix as a forgetting curve. The curve plots the Accuracy against the lag $\ell = j - i$, the

number of slices between the training cutoff i and the evaluation slice j . At each lag we average over all training cutoffs,

$$F(\ell) = \text{mean}_i M_{i, i+\ell}.$$

The result is the Accuracy at a fixed temporal distance, independent of which period a model was trained on. This separates the effect of temporal distance from the difficulty of any single slice.

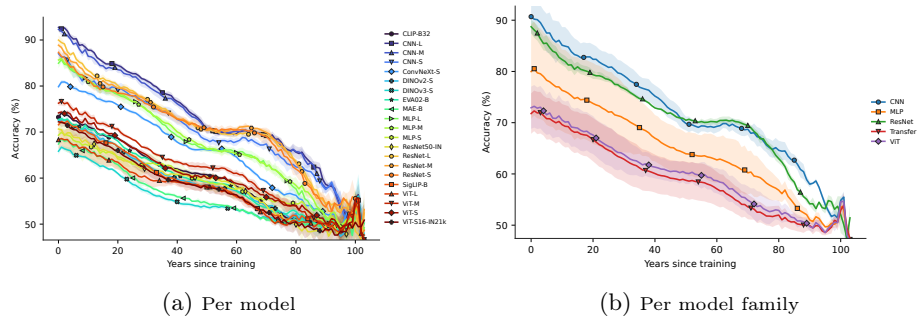


Fig. 11: Forgetting curves: each model (left) and averaged within each family (right).

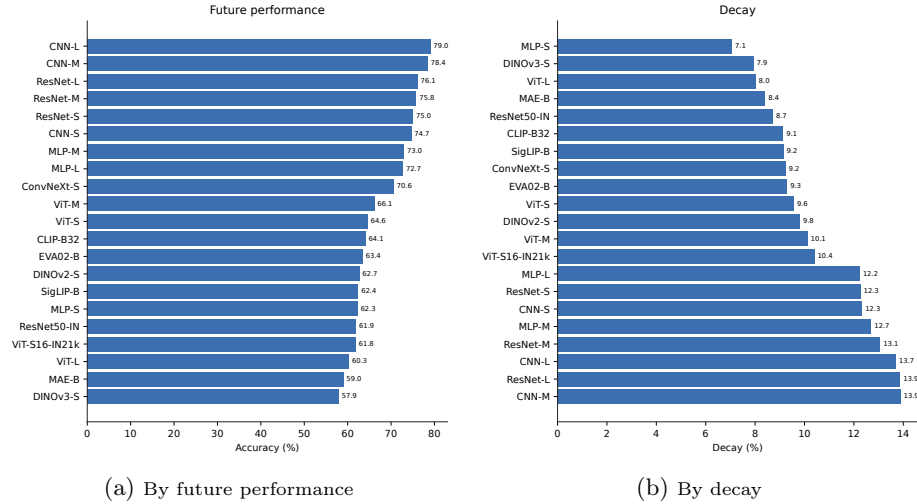


Fig. 12: Models ranked by mean future performance and by temporal decay.

C.8 Result Tables

Table 5: Temporal robustness on Yearbook.

Model	Future (%)	Decay (%)	Model	Future (%)	Decay (%)
MLP-S	62.3	7.1	ViT-M	66.1	10.1
DINOv3-S	57.9	7.9	ViT-S16-IN21k	61.8	10.4
ViT-L	60.3	8.0	MLP-L	72.7	12.2
MAE-B	59.0	8.4	ResNet-S	75.0	12.3
ResNet50-IN	61.9	8.7	CNN-S	74.7	12.3
CLIP-B32	64.1	9.1	MLP-M	73.0	12.7
SigLIP-B	62.4	9.2	ResNet-M	75.8	13.1
ConvNeXt-S	70.6	9.2	CNN-L	79.0	13.7
EVA02-B	63.4	9.3	ResNet-L	76.1	13.9
ViT-S	64.6	9.6	CNN-M	78.4	13.9
DINOv2-S	62.7	9.8			

Table 6: Yearbook: models trained up to 1905, ordered by future performance. Table 7: Yearbook: models trained up to 1944, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)	Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	ViT-S16-IN21k	50.0	50.7	-0.7	1	ResNet-L	98.4	82.9	15.5
2	ResNet-M	60.0	50.3	9.7	2	ResNet-M	98.3	81.9	16.4
3	ResNet-L	70.0	50.1	19.9	3	CNN-M	98.8	81.4	17.4
4	MLP-L	40.0	49.8	-9.8	4	ResNet-S	98.0	81.1	16.9
5	MAE-B	40.0	49.5	-9.5	5	CNN-L	97.1	80.2	16.8
6	ResNet-S	70.0	49.3	20.7	6	CNN-S	95.0	79.1	15.9
7	DINOv2-S	50.0	48.5	1.5	7	ConvNeXt-S	96.3	78.5	17.8
8	DINOv3-S	50.0	48.3	1.7	8	MLP-M	93.4	77.6	15.9
9	ViT-L	50.0	46.8	3.2	9	MLP-L	87.9	74.6	13.3
10	MLP-S	40.0	46.3	-6.3	10	ViT-M	92.6	71.4	21.2
11	ConvNeXt-S	50.0	46.1	3.9	11	ViT-L	91.0	70.7	20.3
12	ViT-M	50.0	45.7	4.3	12	ViT-S	89.4	69.9	19.5
13	CNN-L	50.0	45.4	4.6	13	CLIP-B32	94.3	69.9	24.5
14	ViT-S	50.0	45.1	4.9	14	EVA02-B	93.1	67.9	25.2
15	ResNet50-IN	50.0	45.1	4.9	15	DINOv2-S	88.5	67.1	21.5
16	EVA02-B	50.0	44.9	5.1	16	SigLIP-B	88.9	66.8	22.1
17	MLP-M	50.0	44.9	5.1	17	ViT-S16-IN21k	86.4	65.0	21.4
18	CNN-S	50.0	44.9	5.1	18	ResNet50-IN	80.2	64.8	15.3
19	CLIP-B32	50.0	44.7	5.3	19	MAE-B	86.7	62.6	24.1
20	CNN-M	60.0	44.7	15.3	20	DINOv3-S	82.7	62.4	20.3
21	SigLIP-B	50.0	44.7	5.3	21	MLP-S	65.8	57.9	7.9

Table 8: Yearbook: models trained up to 1978, ordered by future performance. Table 9: Yearbook: models trained up to 2012, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)	Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	CNN-L	85.9	91.4	-5.4	1	CNN-L	93.2	98.2	-5.0
2	ResNet-M	88.7	90.4	-1.6	2	CNN-M	93.2	97.7	-4.6
3	ResNet-L	87.6	89.3	-1.7	3	ResNet-L	88.9	97.5	-8.5
4	CNN-M	86.8	88.8	-2.0	4	ResNet-M	92.6	97.0	-4.3
5	MLP-M	79.4	84.5	-5.1	5	ResNet-S	86.8	95.4	-8.5
6	ResNet-S	83.4	84.4	-1.0	6	CNN-S	83.7	93.4	-9.7
7	MLP-L	75.5	83.2	-7.7	7	MLP-L	87.4	91.9	-4.6
8	CNN-S	72.7	81.1	-8.5	8	MLP-M	86.3	91.7	-5.4
9	ConvNeXt-S	80.6	78.6	1.9	9	ConvNeXt-S	73.7	91.2	-17.5
10	ViT-M	72.4	76.8	-4.4	10	DINOv2-S	82.6	90.0	-7.3
11	CLIP-B32	69.9	74.8	-5.0	11	ViT-M	77.9	88.1	-10.2
12	EVA02-B	60.6	73.3	-12.8	12	ViT-S	80.5	88.0	-7.5
13	ViT-L	66.8	72.4	-5.6	13	SigLIP-B	85.8	87.5	-1.7
14	DINOv2-S	64.5	72.3	-7.8	14	EVA02-B	82.1	87.4	-5.3
15	ViT-S	64.2	71.8	-7.6	15	CLIP-B32	74.7	86.6	-11.9
16	ViT-S16-IN21k	66.5	70.7	-4.3	16	ViT-S16-IN21k	87.9	86.5	1.4
17	ResNet50-IN	72.7	70.7	2.0	17	ResNet50-IN	76.8	83.8	-7.0
18	SigLIP-B	64.8	69.9	-5.1	18	DINOv3-S	71.1	77.9	-6.9
19	MLP-S	67.9	69.4	-1.5	19	MAE-B	78.4	77.4	1.0
20	MAE-B	56.1	65.0	-8.9	20	MLP-S	70.0	76.1	-6.1
21	DINOv3-S	60.3	59.9	0.4	21	ViT-L	66.8	74.4	-7.5

Table 10: Yearbook: future performance and decay by model family.

Family	1905		1944		1978		2012	
	Future	Decay	Future	Decay	Future	Decay	Future	Decay
CNN	45.0	8.3	80.3	16.7	87.1	-5.3	96.4	-6.4
MLP	47.0	-3.7	70.0	12.4	79.0	-4.8	86.6	-5.3
ResNet	49.9	16.8	82.0	16.2	88.0	-1.4	96.6	-7.1
Transfer	46.9	1.9	67.2	21.4	70.6	-4.4	85.4	-6.1
ViT	45.9	4.1	70.7	20.3	73.7	-5.9	83.5	-8.4

D Amazon Reviews – Drift Matrices

The cohort-mean and per-model deviation matrices shown here, and the in-distribution, future, and decay quantities tabulated below, are defined in Section 4.5.

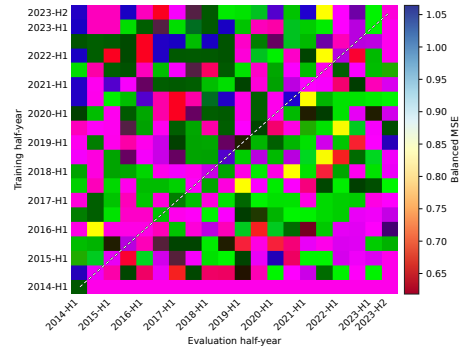


Fig. 13: Cohort-mean Balanced MSE matrix $\bar{M}$ over the Amazon Reviews models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j .

D.1 Model Roster

Table 11: Amazon Reviews: models trained from scratch.

Model	Family	Trainable	Total
TX-S	Transformer	83k	124.7M
TextCNN-S	TextCNN	88k	124.7M
FFN-S	FFN	99k	124.7M
BiGRU-S	Recurrent	99k	124.7M
FFN-M	FFN	394k	125.0M
TextCNN-M	TextCNN	461k	125.1M
TX-M	Transformer	493k	125.1M
BiLSTM-M	Recurrent	536k	125.2M
FFN-L	FFN	1.6M	126.2M
TX-L	Transformer	1.9M	126.5M
TextCNN-L	TextCNN	1.9M	126.6M
BiLSTM-Attn-L	Recurrent	2.2M	126.8M

Table 12: Amazon Reviews: frozen pre-trained encoders, with trainable head and total parameters.

Model	Family	Trainable	Total
DistilBERT	Frozen	769	66.4M
ELECTRA	Frozen	769	108.9M
BERT	Frozen	769	109.5M
MPNet	Frozen	769	109.5M
RoBERTa	Frozen	769	124.6M
ModernBERT	Frozen	769	149.0M
DeBERTa-v3	Frozen	769	183.8M

D.2 FFN

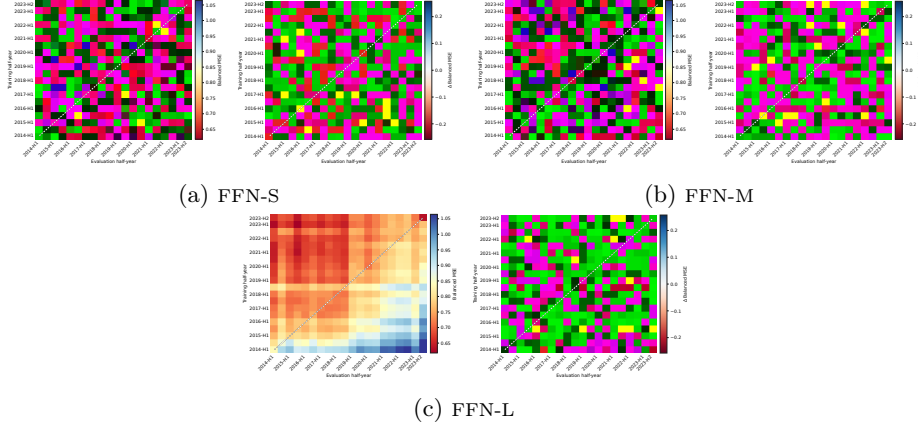


Fig. 14: FFN models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

D.3 TextCNN

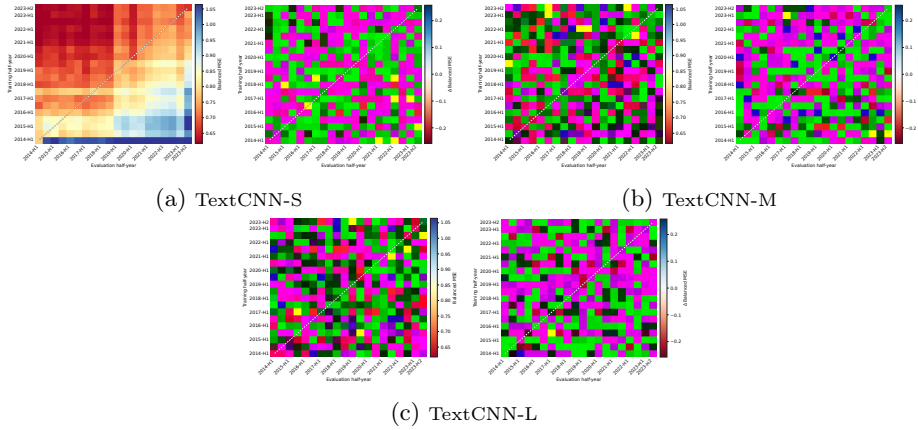


Fig. 15: TextCNN models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

D.4 Recurrent

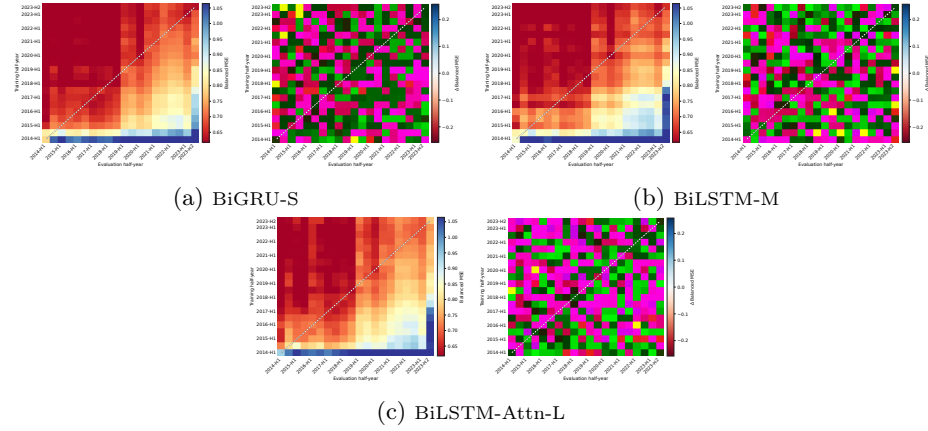


Fig. 16: Recurrent models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

D.5 Transformer

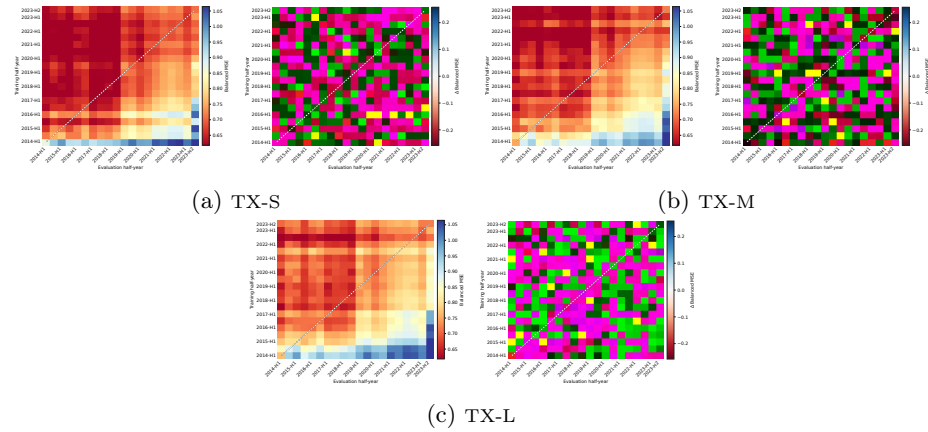


Fig. 17: Transformer models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

D.6 Frozen

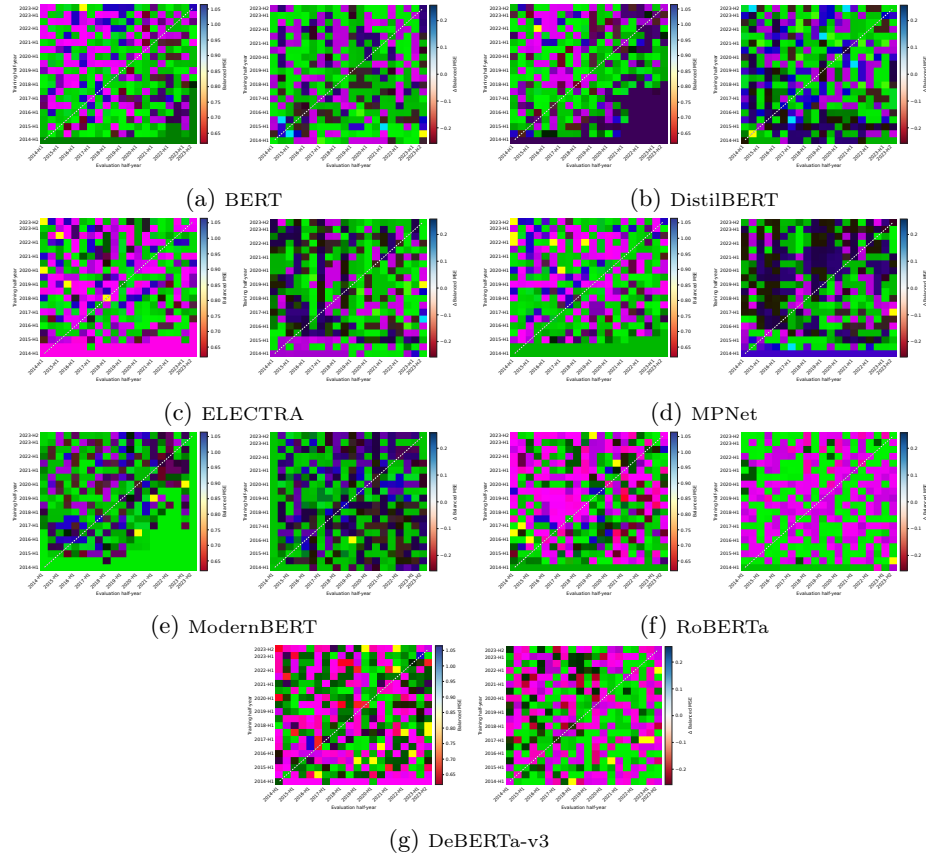


Fig. 18: Frozen models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

D.7 Forgetting and Rankings

To see how quickly each model forgets, we summarize its drift matrix as a forgetting curve. The curve plots the Balanced MSE against the lag $\ell = j - i$, the number of slices between the training cutoff i and the evaluation slice j . At each lag we average over all training cutoffs,

$$F(\ell) = \text{mean}_i M_{i, i+\ell}.$$

The result is the Balanced MSE at a fixed temporal distance, independent of which period a model was trained on. This separates the effect of temporal distance from the difficulty of any single slice.

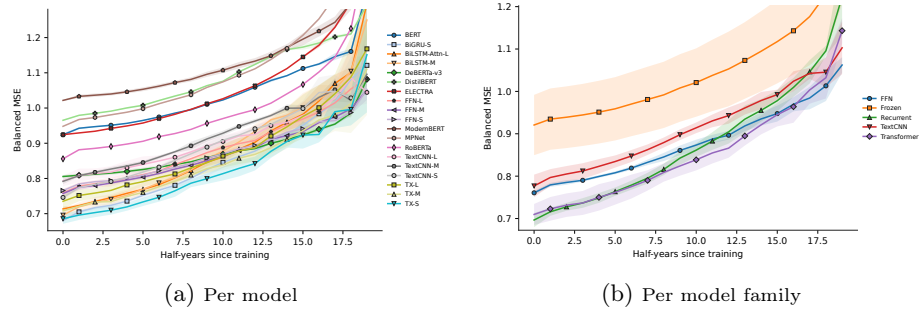


Fig. 19: Forgetting curves: each model (left) and averaged within each family (right).

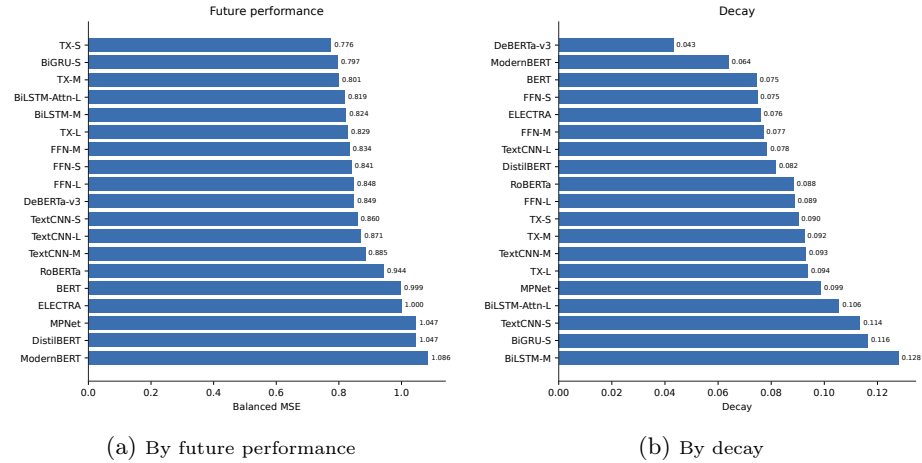


Fig. 20: Models ranked by mean future performance and by temporal decay.

D.8 Result Tables

Table 13: Temporal robustness on Amazon Reviews.

Model	Future Decay		Model	Future Decay	
DeBERTa-v3	0.849	0.043	TX-S	0.776	0.090
ModernBERT	1.086	0.064	TX-M	0.801	0.092
BERT	0.999	0.075	TextCNN-M	0.885	0.093
FFN-S	0.841	0.075	TX-L	0.829	0.094
ELECTRA	1.000	0.076	MPNet	1.047	0.099
FFN-M	0.834	0.077	BiLSTM-Attn-L	0.819	0.106
TextCNN-L	0.871	0.078	TextCNN-S	0.860	0.114
DistilBERT	1.047	0.082	BiGRU-S	0.797	0.116
RoBERTa	0.944	0.088	BiLSTM-M	0.824	0.128
FFN-L	0.848	0.089			

Table 14: Amazon Reviews: models trained up to 2014-H1, ordered by future performance.

Table 15: Amazon Reviews: models trained up to 2017-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay	Rank	Model	Balanced MSE	Future	Decay
1	TX-M	0.748	0.933	0.185	1	TX-S	0.625	0.748	0.123
2	TextCNN-L	0.782	0.935	0.153	2	BiGRU-S	0.644	0.765	0.121
3	FFN-L	0.836	0.972	0.136	3	TX-M	0.701	0.785	0.084
4	TX-L	0.767	0.981	0.215	4	BiLSTM-Attn-L	0.654	0.789	0.136
5	FFN-M	0.885	0.986	0.100	5	FFN-L	0.724	0.810	0.086
6	TX-S	0.868	0.987	0.119	6	FFN-S	0.723	0.812	0.089
7	TextCNN-M	0.761	0.993	0.231	7	DeBERTa-v3	0.758	0.814	0.056
8	FFN-S	0.907	1.023	0.116	8	TextCNN-S	0.711	0.817	0.106
9	BiGRU-S	0.790	1.073	0.283	9	TX-L	0.722	0.818	0.096
10	TextCNN-S	0.782	1.078	0.296	10	BiLSTM-M	0.681	0.828	0.147
11	DeBERTa-v3	1.035	1.083	0.048	11	FFN-M	0.742	0.829	0.087
12	BiLSTM-Attn-L	0.938	1.105	0.167	12	RoBERTa	0.792	0.860	0.068
13	BiLSTM-M	0.850	1.127	0.277	13	TextCNN-L	0.789	0.876	0.087
14	BERT	1.067	1.232	0.165	14	TextCNN-M	0.773	0.885	0.113
15	DistilBERT	1.113	1.250	0.137	15	ELECTRA	0.873	0.926	0.052
16	ELECTRA	1.332	1.382	0.050	16	MPNet	0.868	0.940	0.072
17	ModernBERT	1.273	1.386	0.113	17	BERT	0.929	0.984	0.055
18	RoBERTa	1.227	1.468	0.241	18	ModernBERT	0.958	1.028	0.070
19	MPNet	1.492	1.657	0.165	19	DistilBERT	0.942	1.047	0.104

Table 16: Amazon Reviews: models trained up to 2020-H1, ordered by future performance. Table 17: Amazon Reviews: models trained up to 2023-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay	Rank	Model	Balanced MSE	Future	Decay
1	BiLSTM-M	0.614	0.706	0.092	1	TX-M	0.634	0.665	0.031
2	BiGRU-S	0.627	0.715	0.088	2	BiGRU-S	0.627	0.691	0.064
3	TX-S	0.637	0.721	0.084	3	BiLSTM-M	0.638	0.703	0.065
4	BiLSTM-Attn-L	0.647	0.749	0.102	4	BiLSTM-Attn-L	0.687	0.735	0.048
5	TX-M	0.675	0.751	0.076	5	TX-S	0.663	0.740	0.077
6	TextCNN-S	0.672	0.767	0.094	6	TextCNN-S	0.682	0.747	0.066
7	TX-L	0.693	0.773	0.080	7	TX-L	0.699	0.759	0.060
8	FFN-L	0.713	0.793	0.080	8	DeBERTa-v3	0.776	0.764	-0.013
9	FFN-S	0.729	0.795	0.066	9	FFN-L	0.728	0.770	0.042
10	FFN-M	0.744	0.817	0.074	10	FFN-S	0.731	0.777	0.046
11	DeBERTa-v3	0.753	0.820	0.067	11	FFN-M	0.742	0.821	0.079
12	TextCNN-M	0.748	0.833	0.085	12	TextCNN-M	0.751	0.838	0.087
13	TextCNN-L	0.743	0.834	0.091	13	RoBERTa	0.812	0.847	0.035
14	RoBERTa	0.800	0.870	0.070	14	TextCNN-L	0.794	0.869	0.075
15	ELECTRA	0.844	0.911	0.067	15	ELECTRA	0.882	0.894	0.012
16	MPNet	0.860	0.938	0.078	16	BERT	0.911	0.939	0.029
17	BERT	0.876	0.959	0.084	17	MPNet	0.863	0.956	0.093
18	DistilBERT	0.926	0.998	0.072	18	DistilBERT	0.937	0.980	0.043
19	ModernBERT	0.969	1.041	0.072	19	ModernBERT	0.992	1.069	0.076

Table 18: Amazon Reviews: future performance and decay by model family.

Family	2014-H1		2017-H1		2020-H1		2023-H1	
	Future	Decay	Future	Decay	Future	Decay	Future	Decay
FFN	0.993	0.117	0.817	0.087	0.802	0.073	0.789	0.056
Frozen	1.351	0.131	0.943	0.068	0.934	0.073	0.921	0.039
Recurrent	1.101	0.242	0.794	0.134	0.723	0.094	0.710	0.059
TextCNN	1.002	0.227	0.859	0.102	0.811	0.090	0.818	0.076
Transformer	0.967	0.173	0.784	0.101	0.748	0.080	0.721	0.056

E arXiv – Drift Matrices

The cohort-mean and per-model deviation matrices shown here, and the in-distribution, future, and decay quantities tabulated below, are defined in Section 4.5. The label space comprises the leaf categories `cs.LG`, `hep-ph`, `cs.CV`, `cs.AI`, `hep-th`, `quant-ph`, and `gr-qc`.

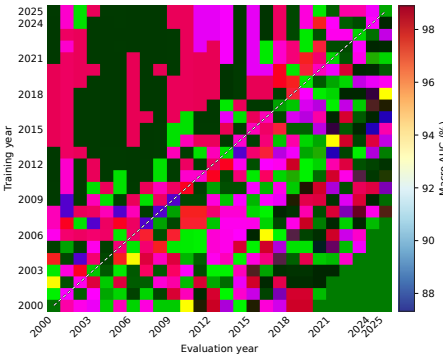


Fig. 21: Cohort-mean Macro AUC matrix $\bar{M}$ over the arXiv models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j .

E.1 Model Roster

Table 19: arXiv: models trained from scratch. Table 20: arXiv: frozen pretrained encoders, with trainable head and total parameters.

Model	Family	Trainable	Total
TX-S	Transformer	83k	124.7M
TextCNN-S	TextCNN	89k	124.7M
FFN-S	FFN	99k	124.7M
BiGRU-S	Recurrent	100k	124.7M
FFN-M	FFN	397k	125.0M
TextCNN-M	TextCNN	462k	125.1M
TX-M	Transformer	493k	125.1M
BiLSTM-M	Recurrent	537k	125.2M
FFN-L	FFN	1.6M	126.2M
TX-L	Transformer	1.9M	126.5M
TextCNN-L	TextCNN	1.9M	126.6M
BiLSTM-Attn-L	Recurrent	2.2M	126.8M

Model	Family	Trainable	Total
MiniLM-L6	Frozen	3k	22.7M
DistilBERT	Frozen	5k	66.4M
ELECTRA	Frozen	5k	108.9M
BERT	Frozen	5k	109.5M
MPNet	Frozen	5k	109.5M
RoBERTa	Frozen	5k	124.7M
ModernBERT	Frozen	5k	149.0M
DeBERTa-v3	Frozen	5k	183.8M

E.2 FFN

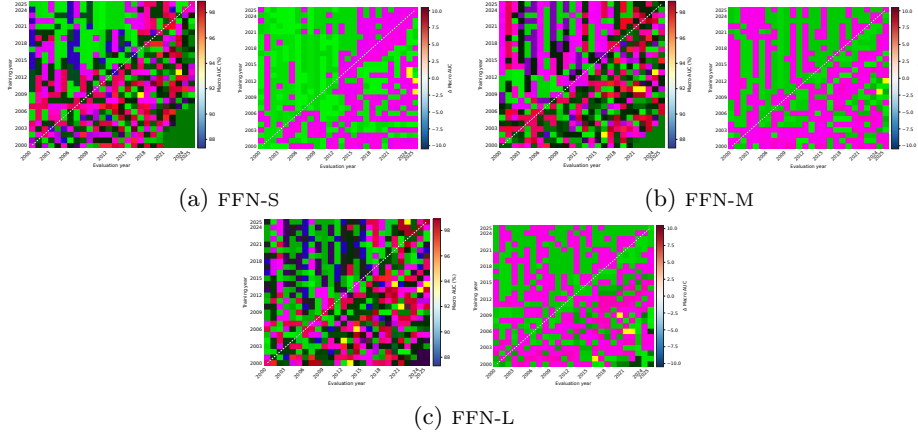


Fig. 22: FFN models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

E.3 TextCNN

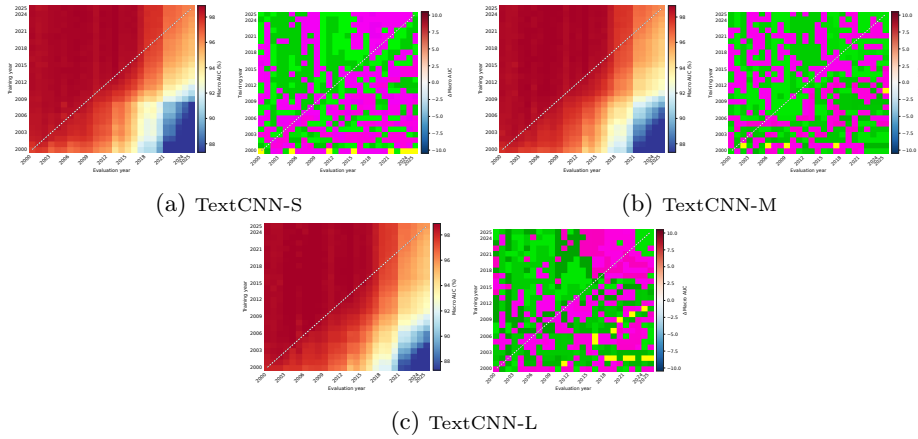


Fig. 23: TextCNN models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

E.4 Recurrent

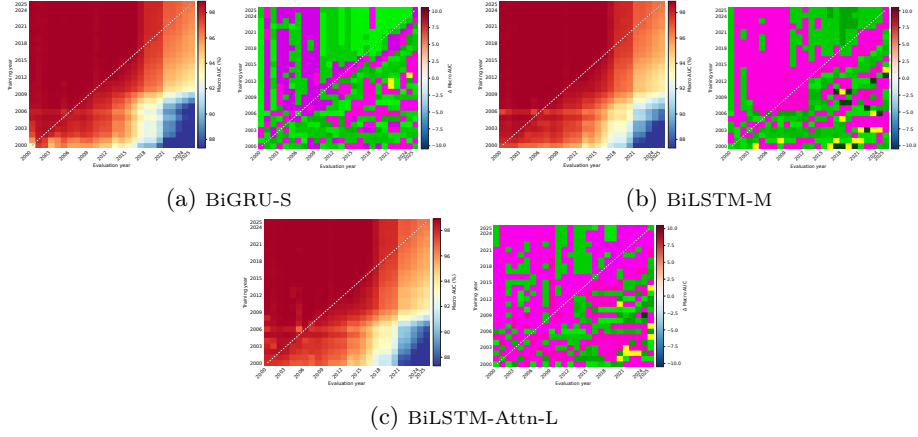


Fig. 24: Recurrent models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

E.5 Transformer

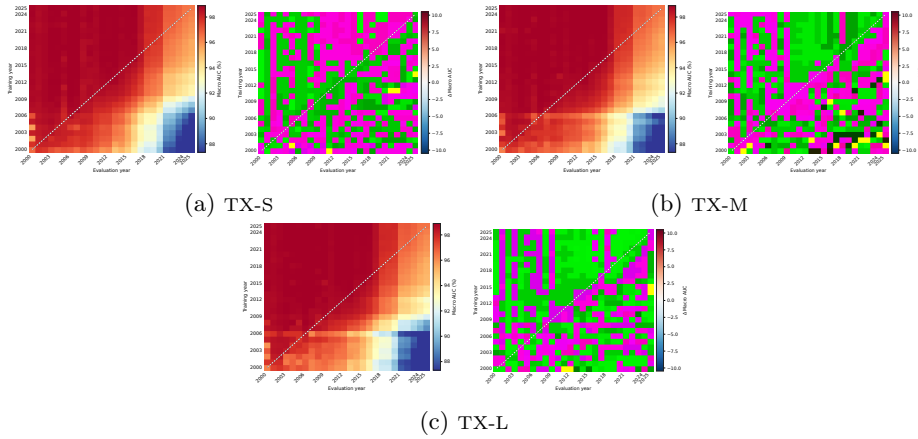


Fig. 25: Transformer models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

E.6 Frozen

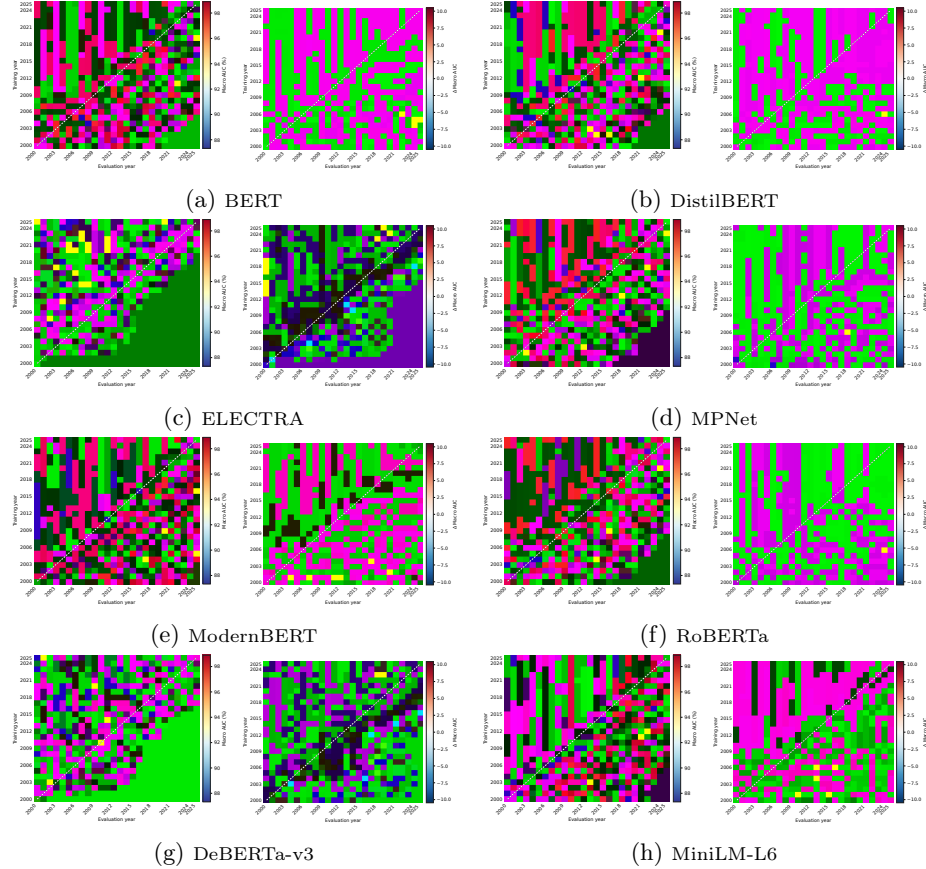


Fig. 26: Frozen models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

E.7 Forgetting and Rankings

To see how quickly each model forgets, we summarize its drift matrix as a forgetting curve. The curve plots the Macro AUC against the lag $\ell = j - i$, the number of slices between the training cutoff i and the evaluation slice j . At each lag we average over all training cutoffs,

$$F(\ell) = \text{mean}_i M_{i, i+\ell}.$$

The result is the Macro AUC at a fixed temporal distance, independent of which period a model was trained on. This separates the effect of temporal distance from the difficulty of any single slice.

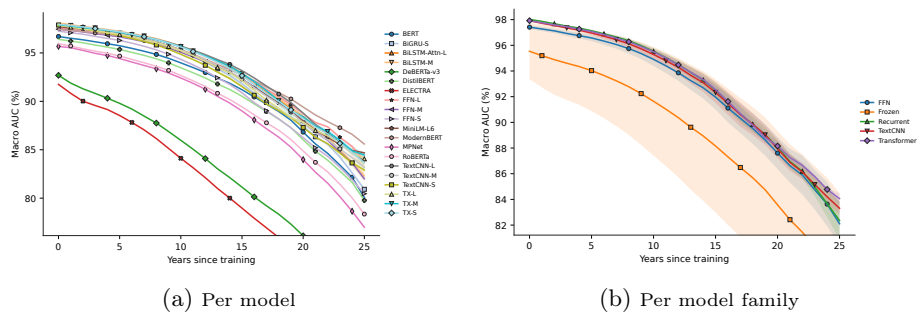


Fig. 27: Forgetting curves: each model (left) and averaged within each family (right).

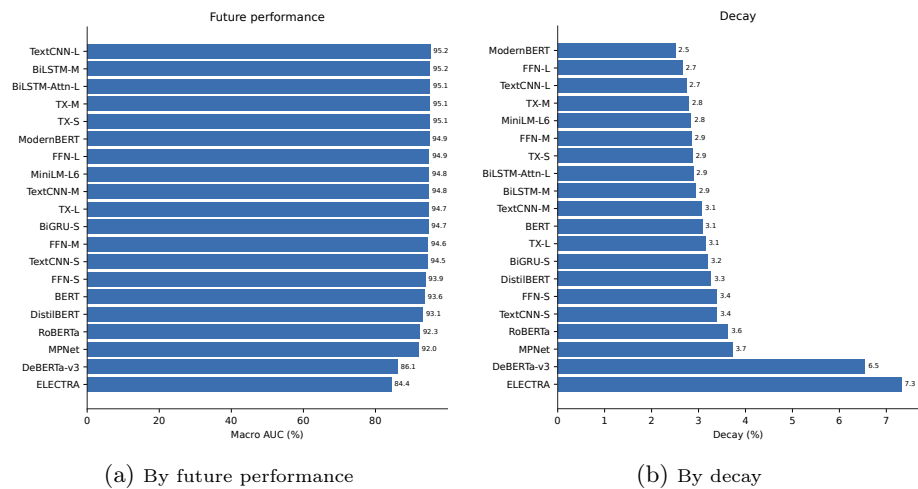


Fig. 28: Models ranked by mean future performance and by temporal decay.

E.8 Result Tables

Table 21: Temporal robustness on arXiv.

Model	Future (%)	Decay (%)	Model	Future (%)	Decay (%)
ModernBERT	94.9	2.5	BERT	93.6	3.1
FFN-L	94.9	2.7	TX-L	94.7	3.1
TextCNN-L	95.2	2.7	BiGRU-S	94.7	3.2
TX-M	95.1	2.8	DistilBERT	93.1	3.3
MiniLM-L6	94.8	2.8	FFN-S	93.9	3.4
FFN-M	94.6	2.9	TextCNN-S	94.5	3.4
TX-S	95.1	2.9	RoBERTa	92.3	3.6
BiLSTM-Attn-L	95.1	2.9	MPNet	92.0	3.7
BiLSTM-M	95.2	2.9	DeBERTa-v3	86.1	6.5
TextCNN-M	94.8	3.1	ELECTRA	84.4	7.3

Table 22: arXiv: models trained up to 2000, ordered by future performance. Table 23: arXiv: models trained up to 2008, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)	Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	MiniLM-L6	97.9	93.9	4.0	1	TX-M	98.6	96.0	2.6
2	ModernBERT	97.5	93.9	3.7	2	TextCNN-L	98.7	95.7	3.0
3	TextCNN-L	98.1	93.8	4.3	3	BiLSTM-Attn-L	98.5	95.6	2.9
4	TX-M	96.2	93.6	2.7	4	BiLSTM-M	98.8	95.6	3.2
5	BiLSTM-Attn-L	97.8	93.5	4.3	5	TX-S	98.4	95.3	3.1
6	TX-S	96.6	93.5	3.2	6	FFN-L	98.0	95.2	2.8
7	TX-L	96.2	93.4	2.8	7	ModernBERT	98.0	95.0	3.0
8	TextCNN-M	97.7	93.4	4.3	8	MiniLM-L6	97.9	94.9	3.0
9	FFN-L	97.2	93.2	4.0	9	TextCNN-M	98.6	94.9	3.7
10	TextCNN-S	97.4	93.0	4.4	10	FFN-M	97.9	94.9	3.0
11	BiLSTM-M	97.3	92.8	4.5	11	TX-L	98.2	94.8	3.4
12	FFN-M	96.6	92.3	4.3	12	BiGRU-S	98.6	94.5	4.2
13	BiGRU-S	93.3	91.9	1.4	13	BERT	97.4	93.8	3.6
14	FFN-S	95.9	91.3	4.6	14	FFN-S	97.6	93.7	4.0
15	BERT	95.8	90.9	4.9	15	TextCNN-S	98.5	93.6	4.9
16	DistilBERT	94.6	90.2	4.4	16	DistilBERT	97.1	93.3	3.9
17	RoBERTa	91.5	88.0	3.5	17	RoBERTa	96.8	92.5	4.3
18	MPNet	88.9	87.3	1.6	18	MPNet	96.7	92.5	4.3
19	DeBERTa-v3	90.7	81.2	9.5	19	DeBERTa-v3	93.4	86.1	7.3
20	ELECTRA	88.9	77.3	11.7	20	ELECTRA	92.5	84.6	7.9

Table 24: arXiv: models trained up to 2016, ordered by future performance. Table 25: arXiv: models trained up to 2024, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)	Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	BiLSTM-M	98.8	96.6	2.2	1	TX-M	96.5	96.3	0.1
2	BiGRU-S	98.8	96.6	2.2	2	BiLSTM-M	96.5	96.3	0.2
3	TX-S	98.7	96.5	2.2	3	TX-L	96.4	96.3	0.2
4	TX-M	98.7	96.5	2.2	4	BiLSTM-Attn-L	96.4	96.3	0.2
5	TX-L	98.7	96.5	2.2	5	BiGRU-S	96.4	96.3	0.2
6	BiLSTM-Attn-L	98.7	96.4	2.3	6	TX-S	96.4	96.2	0.2
7	TextCNN-L	98.6	96.3	2.3	7	TextCNN-S	96.2	96.0	0.2
8	TextCNN-S	98.5	96.2	2.3	8	FFN-L	96.0	95.8	0.2
9	TextCNN-M	98.5	96.2	2.4	9	FFN-M	96.0	95.8	0.2
10	FFN-L	98.3	96.2	2.2	10	TextCNN-M	96.0	95.8	0.3
11	FFN-M	98.2	96.1	2.2	11	FFN-S	95.9	95.7	0.2
12	MiniLM-L6	98.4	96.0	2.4	12	MiniLM-L6	95.9	95.6	0.3
13	FFN-S	98.1	96.0	2.2	13	TextCNN-L	95.8	95.5	0.3
14	ModernBERT	98.2	95.8	2.4	14	ModernBERT	95.6	95.2	0.4
15	BERT	97.6	95.1	2.5	15	BERT	95.1	94.7	0.4
16	DistilBERT	97.3	94.8	2.5	16	DistilBERT	94.9	94.5	0.4
17	RoBERTa	97.0	94.6	2.4	17	RoBERTa	94.9	94.5	0.4
18	MPNet	97.0	94.4	2.5	18	MPNet	94.9	94.4	0.4
19	DeBERTa-v3	93.6	90.0	3.7	19	DeBERTa-v3	92.1	91.2	0.8
20	ELECTRA	92.6	88.3	4.3	20	ELECTRA	91.5	90.4	1.1

Table 26: arXiv: future performance and decay by model family.

Family	2000		2008		2016		2024	
	Future	Decay	Future	Decay	Future	Decay	Future	Decay
FFN	92.2	4.3	94.6	3.3	96.1	2.2	95.8	0.2
Frozen	87.8	5.4	91.6	4.6	93.6	2.8	93.8	0.5
Recurrent	92.7	3.4	95.2	3.4	96.5	2.2	96.3	0.2
TextCNN	93.4	4.3	94.7	3.9	96.2	2.3	95.8	0.3
Transformer	93.5	2.9	95.4	3.0	96.5	2.2	96.3	0.2

Beyond Single-Objective Accuracy: A Longitudinal Evaluation of Tabular Foundation Models under Dataset Shift

João Marcos Tomáz Silva Campos¹, Rafael Martins Gomes², Diogo Tuler Chaves², Mariangela Leal Cherchiglia², Hugo André da Rocha², Leonardo Chaves Dutra da Rocha², Wagner Meira Junior², and Marcos André Gonçalves²

¹ Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, Brazil
{joaomarcoscamos, mgoncalv, meira, rafaelgomes, diogo.tuler}@dcc.ufmg.br, rochahugoandre@gmail.com, mcherchiglia@gmail.com

² Universidade Federal de São João del-Rei (UFSJ), São João del-Rei, Brazil
lcrocha@ufs.j.edu.br

Abstract. Evaluations of classification under dataset shift often rely on static benchmarks and single-objective metrics, obscuring how discrimination, calibration, and risk ranking may fail in different ways over time. We present a 19-year longitudinal benchmark of tabular models under real-world clinical shift, using a Brazilian renal registry with 9.6 million records from 1997–2015. We compare dynamically retrained tree-based ensembles with standard and drift-aware Tabular Foundation Models (TabPFN, TabICL) under uniform temporal protocols. Our results show that robustness is not monolithic: models with strong global survival rankings may suffer from calibration and same-year classification failures, while model rankings change across stable, covariate-shift, and structural/observation-process-shift regimes. We further introduce an exploratory Temporal Robustness Index and combine domain-aware SHAP with medication-record perturbations to diagnose model failures. The analysis reveals that standard models often rely on temporally unstable pharmaceutical proxies, whereas temporally conditioned models emphasize more stable physiological signals. Overall, our study provides a multi-objective and shift-aware auditing framework for evaluating tabular foundation models in dynamic clinical environments.

Keywords: Dataset Shift · Calibration · Temporal Robustness · Tabular Foundation Models · Clinical Forecasting

1 Introduction

Evaluating machine learning models under dataset shift is a central challenge in high-stakes domains. Historically, evaluations of concept drift have relied on single-objective metrics, implicitly assuming that discriminative power,

calibration, and risk ranking degrade uniformly [3]. In real-world longitudinal systems, this assumption rarely holds [2].

While tree-based ensembles [1, 8] have long dominated tabular prediction, recent Tabular Foundation Models (TFMs) such as TabPFN [6] and TabICL [10] introduce a new paradigm leveraging Transformers [13] and in-context learning. Their behavior under long-horizon temporal shift remains largely unexplored.

We investigate whether predictive discrimination, probabilistic calibration, and survival ranking lead to consistent conclusions about model quality under long-horizon dataset shift. To this end, we conduct a 19-year longitudinal evaluation on a national renal registry comprising 9.6 million records from 67,267 patients. Classical ensembles with dynamic sliding-window retraining are compared against standard TFMs and a temporally conditioned drift-aware variant under identical temporal windows. Models are assessed annually using discrimination, calibration, and IPCW-corrected survival ranking, complemented by shift-analysis and diagnostic techniques.

The results show that robustness is not a single property. Discrimination, calibration, and survival ranking diverge under temporal shift, and model rankings change across shift regimes. Standard TFMs perform strongly under stable conditions, whereas gradient boosting is more resilient under structural shift. Temporal conditioning further improves annual discrimination under drift. These findings show that aggregate accuracy can obscure clinically consequential failure modes and that model selection must account for both the evaluation objective and shift regime.

In sum, our contributions are:

- **A long-horizon benchmark for tabular models under shift.** We provide a long-horizon evaluation of classical ensembles, standard TFMs, and drift-aware TFMs under real-world dataset shift.
- **A multi-objective evaluation framework.** We jointly assess discrimination, calibration, and survival ranking, exposing failure modes hidden by single-metric evaluation.
- **Divergent objectives and ranking reversals under shift.** We show that evaluation objectives can lead to different conclusions about model quality and that model rankings reverse across shift regimes.
- **A shift-aware diagnostic methodology.** We combine multiple diagnostic analyses to explain when and why models fail.

2 Background and Related Work

2.1 Classification and Calibration under Dataset Shift

Most supervised learning methods, including tree-based ensembles such as XGBoost and LightGBM [1, 8], assume that training and test data are sampled from a stable distribution $P(X, y)$. In longitudinal clinical registries, this assumption is routinely violated by dataset shift [3, 2], which may arise as covariate shift through changes in $P(X)$, label shift through changes in $P(y)$, or structural

shift through changes in $P(y|X)$. Common mitigation strategies include sliding-window retraining, online adaptation, and periodic model updating [3, 11].

Existing benchmarks under dataset shift frequently summarize performance through a single discriminative metric, despite the fact that classification quality, probabilistic calibration, and longitudinal risk ordering may degrade differently under temporal shift [2]. Longitudinal benchmarks that jointly evaluate these objectives under real-world dataset shift remain comparatively rare. This limitation is particularly consequential in clinical settings, where a model may preserve relative risk ranking while producing unreliable probability estimates or unstable decision boundaries. Our work addresses this gap through a unified longitudinal protocol that jointly evaluates discrimination, calibration, and survival ranking across multiple shift regimes, enabling a more comprehensive assessment of model behavior under long-horizon dataset shift.

2.2 Tabular Foundation Models in Non-Stationary Environments

Tabular Foundation Models (TFMs) [7] extend the foundation-model paradigm to structured data by pretraining on large collections of synthetic tasks and performing prediction through in-context learning. TabPFN [6] approximates Bayesian posterior inference and achieves strong zero-shot performance, while TabICL [10] introduces attention mechanisms designed for larger tabular contexts. Drift-Resilient TabPFN [5] further incorporates explicit temporal conditioning during pretraining to improve adaptation across distributional regimes.

Despite rapid progress in static benchmarks, the behavior of TFMs under prolonged, real-world non-stationarity remains poorly understood. Prior studies have largely emphasized aggregate predictive performance, short-horizon perturbations, or synthetic drift, leaving open whether TFMs preserve calibration, discrimination, and risk ranking over years of evolving clinical and administrative practice. We provide one of the first large-scale longitudinal comparisons of standard and drift-aware TFMs against dynamically retrained classical ensembles under nearly two decades of observed shift.

2.3 Survival-Aware and Mechanistic Evaluation under Shift

Right-censoring complicates longitudinal evaluation because standard classification metrics may misrepresent patient-level risk trajectories [12]. Survival-aware measures such as the IPCW-weighted Concordance Index provide a principled alternative for assessing global risk ordering under censoring [12]. At the same time, post hoc attribution methods such as SHAP [9] can reveal whether predictions depend on stable physiological variables or temporally fragile proxies.

Our study combines survival-aware evaluation, distributional diagnostics, domain-aware attribution, temporal robustness analysis, and controlled feature perturbation. This integration moves beyond documenting performance degradation by identifying *which* objectives fail, *under which* shift regimes, and *why* particular model families become vulnerable.

3 Methodology

3.1 Dataset and Problem Framing

Our study uses a longitudinal cohort derived from Brazil’s national healthcare systems through deterministic and probabilistic record linkage [4]. The cohort comprises 67,267 patients and approximately 9.6 million clinical events recorded between 1997 and 2015. It covers more than 6,200 event types, including renal replacement therapies, chronic disease diagnoses, and medication dispensations, and exhibits pronounced temporal non-stationarity driven by changes in clinical protocols, administrative policies, and data-recording practices. The prediction target is all-cause mortality.

Preliminary experiments showed that the registry contains extreme sequence-length heterogeneity and that, under the severe reduction in recorded exposure observed in 2015, sequential architectures became highly sensitive to zero-padding patterns. These patterns reflect changes in the observation process rather than clinical state and may therefore act as spurious predictors. The patient-year representation mitigates this problem by encoding each annual history as a fixed-length vector of event frequencies, providing a more stable input space for comparing classical ensembles and Tabular Foundation Models under the same temporal conditions.

To satisfy TFM context constraints while retaining clinically relevant information, we select the 500 highest-ranked features according to LightGBM information gain computed exclusively on the 1997–2006 training period (Fig. 1). This procedure avoids using future test years during feature selection, although we acknowledge that model-based selection may favor predictors that are effective for tree-based learning. The resulting features are clinically coherent: hemodialysis, erythropoietin, and renal replacement therapies dominate, together with c2MED-prefixed medication records. These medication variables are highly predictive under relatively stable conditions, but, as later analyses indicate, are also sensitive to administrative and documentation changes.

All sampling is performed at the patient level. Because TFMs are constrained by finite context windows, they receive at most approximately 3,000 patient-year instances per class in each training window. Classical models use the complete balanced training set and therefore benefit from a larger effective sample size. We retain this asymmetry because it reflects the practical operating constraints of current TFMs; accordingly, comparisons should be interpreted as evaluations of deployable model configurations rather than as strictly sample-matched capacity comparisons. A Fixed- K ablation further verifies that the main TFM trends are stable across alternative sampling configurations.

3.2 Evaluation Protocols

A common limitation of temporal-shift benchmarks is the use of a training set frozen at a single historical period. In practice, a model deployed in 2012 would be more plausibly updated with recent data than trained solely on records from the late 1990s. We therefore adopt two complementary temporal protocols, applied uniformly to all model families.

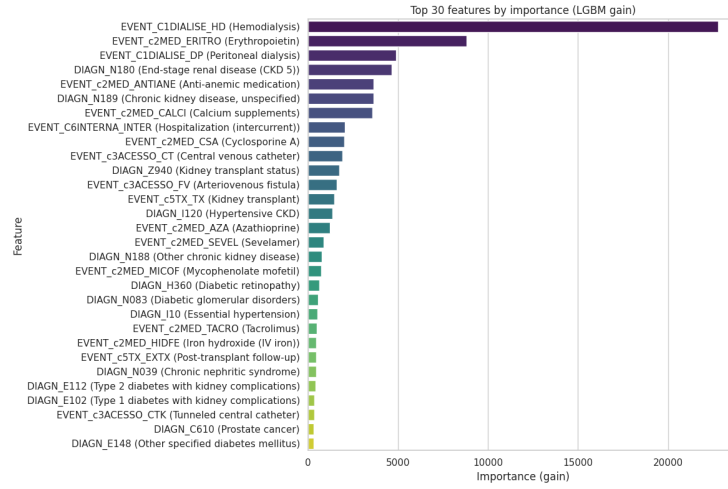


Fig. 1. Top 30 features by LightGBM information gain on the 1997–2006 training split. Core renal procedures dominate; **c2MED**-prefixed medication features emerge as high-importance predictors that are effective under relatively stable conditions but susceptible to administrative and documentation changes.

The **sliding-window protocol** constitutes our primary evaluation. For each test year $Y_{\text{test}} \in \{2007, \dots, 2015\}$ and window size $W \in \{3, 5, 7, 10\}$, classical baselines are retrained from scratch using records from $[Y_{\text{test}} - W, Y_{\text{test}} - 1]$. TabPFN and TabICL receive the same temporal window as their in-context learning set, without gradient-based task-specific training. Drift-Resilient TabPFN additionally receives a temporal-domain tensor $d_i = \text{year_to_domain}[\text{year} \times i]$, thereby enabling explicit conditioning on the temporal regime. We also evaluate an expanding-window variant trained on all available records from 1997 to $Y * \text{test} - 1$.

The **one-year-ahead stream protocol** serves as a robustness analysis of continuous deployment. For each $Y \in 2000, \dots, 2015$, every model is updated using all data available from 1997 to $Y - 1$ and then evaluated on year Y . The earliest years, 1998–1999, are excluded because the registry had not yet accumulated enough observations ($n < 50$) for balanced training. Agreement between this protocol and the main sliding- or expanding-window analyses provides evidence that the reported degradation patterns are not artifacts of a single update strategy.

Across both protocols, training data are balanced through patient-level sampling, with $N \leq 3,000$ instances per class for TFMs. Test sets preserve the naturally imbalanced mortality prevalence, which ranges from 4.2% to 26%, to approximate deployment conditions. Patients in a test pool never appear in the corresponding training window, preventing patient-level leakage across temporal partitions.

3.3 Multi-Objective Evaluation

Our primary evaluation treats robustness as a multi-objective property rather than as a single performance score. For each test year, we assess three related but non-equivalent objectives.

Discrimination is measured using Macro-F1, positive-class F1, and PR-AUC. These metrics evaluate annual mortality classification under severe class imbalance. The classification target is defined as the occurrence of any DEATH event within the same calendar year, and all post-death rows are removed to prevent leakage.

Calibration is assessed using Expected Calibration Error (ECE), which measures agreement between predicted probabilities and observed annual mortality frequencies. Because ECE is sensitive to sample size, prevalence, and binning, we interpret it as a comparative diagnostic of probability reliability across years and model families rather than as an absolute measure of calibration quality.

Global risk ranking is evaluated using the IPCW-adjusted Concordance Index [12] under a 2-year landmark survival horizon. The IPCW weights are computed as $W_i = \min(1/\hat{G}(T_i), \tau_{\max})$, with clipping at the 95th percentile to limit variance. This survival objective is intentionally broader than the annual classification target: it evaluates longitudinal ordering over a two-year horizon rather than same-year mortality decisions. The two tasks are therefore related but not identical, and divergence between them is interpreted as evidence of objective-specific robustness rather than as direct disagreement between interchangeable metrics.

Patient-level bootstrap confidence intervals for annual F1 estimates are computed using 1,000 resamples clustered by patient ID and are reported in the supplementary material. Year-over-year distributional change is quantified using Jensen–Shannon divergence over binary feature distributions.

3.4 Exploratory Shift Characterization and Failure Diagnostics

Beyond the primary metrics, we employ a set of exploratory diagnostics to characterize the observed shifts and investigate potential failure mechanisms. These analyses are intended to support interpretation rather than to define additional confirmatory endpoints.

The **Temporal Robustness Index** is defined as $\text{TRI}_y = (F1_y/F1_{\text{ref}})/(\text{JS}_y + \varepsilon)$, with $\varepsilon = 10^{-4}$. TRI contextualizes each model’s performance retention relative to the magnitude of measured distributional change. Because the denominator can amplify differences when JS divergence is small, TRI is used exclusively as an intra-model diagnostic and not as a calibrated cross-model ranking statistic.

We further organize test years into an exploratory **drift-regime taxonomy**. Years with low JS divergence and relatively stable mortality prevalence are described as *Stable*. Years with elevated feature divergence but limited prevalence change are characterized as *Type I Covariate Shift*. Years with simultaneous

changes in feature sparsity, prevalence, censoring, and observed predictive relationships are characterized as *Type II Structural/Observation-Process Shift*. These labels summarize empirical patterns and domain evidence; they should not be interpreted as formal identification of the underlying causal distribution components. In particular, the 2015 regime reflects administrative truncation of the observation process, which changes the empirically available relationship between recorded features and outcomes without necessarily implying a change in the underlying biological mechanism.

To investigate model reliance on temporally stable or unstable predictors, **domain-aware SHAP** keeps temporal embeddings $E_{\text{dom}}(d_i)$ fixed during permutation, so that attributions reflect feature contributions within the appropriate temporal context [9]. Finally, a **medication perturbation sweep** sets the 11 c2MED features to zero for increasing fractions $p \in 0\%, \dots, 100\%$ of 2015 test patients. This intervention does not establish a causal clinical effect of medication use; rather, it tests whether model predictions depend on medication-recording patterns that became unreliable under the 2015 administrative shift.

4 Results and Dataset Shift Analysis

4.1 Divergence of Discrimination, Calibration, and Ranking

We evaluated all models year by year over the chronological test period (2007–2015) using the dynamic sliding-window protocol. Our central hypothesis is that robustness under long-horizon dataset shift is objective-dependent: discrimination, calibration, and global risk ranking may degrade differently and produce different model orderings. Table 1 supports this hypothesis across the evaluated architectures.

Predictive discrimination and survival ranking diverge substantially. TabICL achieves the strongest global risk ordering (IPCW C-index: 0.542), but this advantage does not transfer to same-year mortality classification in 2015, where its positive-class F1 falls to 0.109. In contrast, Drift-Resilient TabPFN achieves the best average annual F1 (0.316), PR-AUC (0.249), and 2015 F1 (0.212), although it does not lead the survival-ranking objective.

This divergence is practically relevant. Annual classification evaluates the same-year decision quality, whereas IPCW C-index evaluates patient ordering over a 2-year horizon. A model may therefore rank patients well over time while still providing unreliable probabilities or weak decision boundaries for immediate clinical action.

4.2 Characterizing Shift Regimes: Covariate (2012) and Structural/Observation-Process Shift (2015)

We contextualize degradation trajectories using JS divergence between binary feature distributions, interpreted alongside prevalence, sparsity, censoring, and domain evidence. Two distinct regimes emerge, but these labels summarize observed patterns rather than formally identifying changes in $P(X)$ or $P(y|X)$.

Table 1. Summary of longitudinal classification and survival-ranking performance (2007–2015). Annual metrics are macro-averaged using the fixed $W = 10$ window. Lower ECE is better. IPCW C-index uses a 2-year landmark horizon; 2014–2015 are excluded because the required survival estimates are not identifiable under extreme censoring and absence of recorded events (Section 4.4). Patient-level bootstrap 95% confidence intervals for F1-macro are reported in the supplementary material.

Model	Annual Classification (avg. 2007–2015)				Survival Ranking		
	Avg. F1 _{pos}	Avg. PR-AUC	Avg. ECE	2015 F1 _{pos}	C-index	IPCW	C-index
XGBoost	0.238	0.164	0.180	0.164	0.527		0.445
LightGBM	0.247	0.173	0.177	0.178	0.536		0.425
Random Forest	0.197	0.135	0.089	0.109	0.516		0.377
TabPFN	0.221	0.155	0.215	0.103	0.521		0.420
TabICL	0.222	0.165	0.263	0.109	0.586		0.542
Drift-Resilient TabPFN	0.316	0.249	0.240	0.212	0.575		0.433

Table 2. Temporal Robustness Index (TRI) across shift regimes. Higher values indicate greater retention of within-model performance relative to measured distributional change. TRI is not used as a calibrated cross-model ranking statistic. $\varepsilon = 10^{-4}$.

Model	Stable (2007)	Covariate Shift (2012)	Structural Shift (2015)
XGBoost	1262.1	601.5	210.4
TabICL	1262.1	620.3	205.8
TabPFN	1262.1	743.7	294.6
Drift-Resilient TabPFN	1254.3	770.3	410.2

The 2012 Administrative Covariate Shift. In 2012, all models show a sudden discriminative decline. Domain evidence links this period to SUS billing changes that temporarily altered medication-recording frequencies, including Sevelamer and Erythropoietin. JS divergence reaches approximately 0.042, while mortality prevalence remains comparatively stable, suggesting a Type I covariate shift affecting recorded inputs more than outcome frequency.

The 2015 Structural/Observation-Process Shift. The 2015 cohort exhibits a more severe regime. Administrative truncation reduces exposure time, increases feature sparsity, and lowers recorded mortality prevalence from approximately 10% to 4.2%. These changes alter the empirical relationship between recorded features and observable outcomes without necessarily implying a biological change. We therefore describe 2015 as a Type II structural/observation-process shift. Under this regime, standard TFMs show relative Macro-F1 reductions above 10%, classical ensembles decline by approximately 9.4%, and Drift-Resilient TabPFN declines by approximately 3%.

Table 2 reports the Temporal Robustness Index. TRI is interpreted only within each model, since the ratio can become unstable when JS divergence is small. Under this interpretation, Drift-Resilient TabPFN retains the highest TRI in both 2012 and 2015, suggesting slower performance degradation relative to the observed distributional change.

Table 3. Mean PR-AUC stratified by exploratory drift regime across all window configurations ($W \in 3, 5, 7, 10$, fixed and expanding modes). **Stable:** low JS divergence and relatively stable prevalence. **Type I:** 2012 administrative covariate shift. **Type II:** 2015 structural/observation-process shift.

Model	Stable	Type I (Covariate)	Type II (Structural)
XGBoost	0.198	0.208	0.138
LightGBM	0.179	0.184	0.101
Random Forest	0.159	0.169	0.084
TabPFN	0.196	0.209	0.131
TabICL	0.209	0.206	0.129

Model Rankings Depend on Shift Regime. Table 3 shows that aggregate rankings conceal regime-specific behavior. TabICL leads under stable conditions, all architectures perform similarly in the 2012 Type I regime, and XGBoost leads in the 2015 Type II regime. Although numerical differences should be read alongside supplementary uncertainty analyses, the pattern indicates that model suitability depends on the shift regime and cannot be inferred from aggregate performance alone.

4.3 Calibration under Long-Horizon Shift

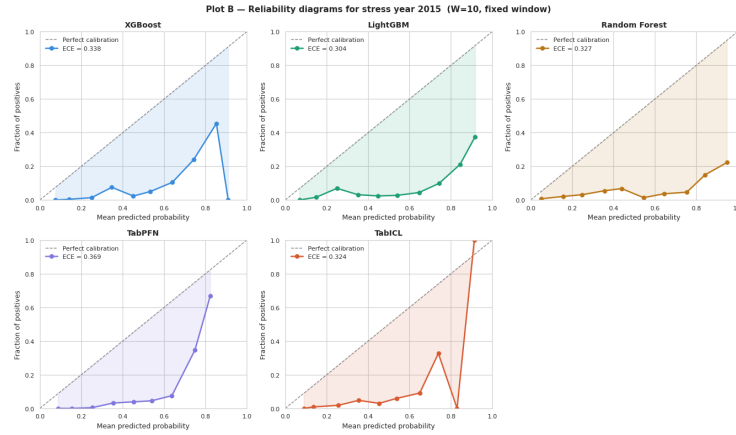


Fig. 2. Reliability diagrams for the 2015 stress year ($W = 10$, fixed window). All models exhibit calibration degradation, with standard TFMs showing pronounced departures from the diagonal despite competitive ranking performance. ECE values are annotated in each panel.

Table 4. Landmark survival analysis for Drift-Resilient TabPFN using a 2-year horizon. IPCW C-index is not estimable for 2014 because all observations are censored, or for 2015 because no events are recorded within the horizon.

Landmark	N	Events	Censored	C-index	IPCW C-index
2007	1,908	240	1,668	0.555	0.544
2008	2,409	284	2,125	0.550	0.565
2009	2,522	307	2,215	0.506	0.547
2010	2,619	311	2,308	0.503	0.511
2011	2,536	241	2,295	0.563	0.547
2012	2,553	256	2,297	0.594	0.597
2013	2,596	218	2,378	0.627	0.628
2014	2,615	71	2,544	0.592	– (all censored)
2015	2,310	0	2,310	–	– (no events)

Figure 2 shows reliability diagrams for the 2015 stress year under the fixed $W = 10$ window. Calibration deteriorates across all model families, but with different failure patterns.

Classical ensembles exhibit overconfidence in higher-probability bins, while standard TFMs are also substantially miscalibrated. TabPFN concentrates probability mass in high-confidence bins without corresponding empirical mortality, and TabICL departs from the diagonal across several bins. Because ECE is sensitive to prevalence, binning, and sample size, we interpret it comparatively rather than as an absolute calibration measure.

Together with Table 1, these results highlight the central multi-objective finding: TabICL achieves the strongest IPCW C-index, yet its 2015 probability estimates remain poorly calibrated. Survival ranking and calibration therefore capture distinct deployment properties and should be audited jointly.

4.4 Survival Analysis and IPCW Correction

Table 4 reports the 2-year landmark survival analysis for Drift-Resilient TabPFN. IPCW correction is estimable for 2007–2013, but not for 2014, when all observations are censored within the horizon, or for 2015, when no deaths are recorded after administrative truncation. These failures reflect limitations of the observed follow-up process and further illustrate the severity of the late-period observation-process shift.

Across estimable years, IPCW C-index ranges from 0.511 to 0.628. The mean absolute difference between corrected and uncorrected C-index is 0.017, indicating that censoring correction changes ranking estimates only modestly before the extreme late-period truncation. Censoring, therefore, contributes to instability but does not fully account for the divergence between annual classification and survival ranking observed over 2007–2013.

4.5 Diagnosing Proxy Reliance through Medication Perturbation

To complement the SHAP analysis, we perturb medication records in the 2015 stress year. For each model trained with the fixed $W = 10$ window, the 11 c2MED features are set to zero for increasing fractions of test patients. This intervention does not estimate the causal effect of medication exposure; rather, it evaluates whether predictions depend on medication-recording patterns that became unreliable after administrative truncation.

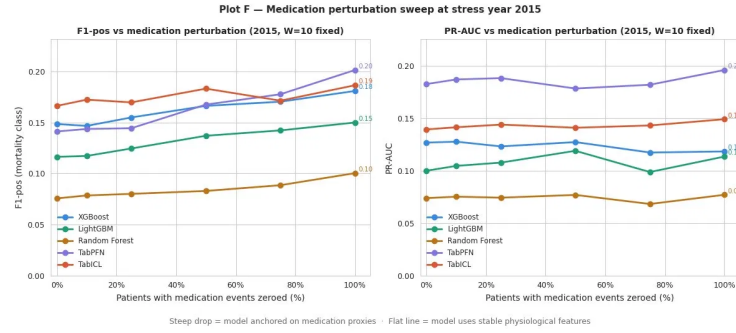


Fig. 3. Medication-record perturbation at the 2015 stress year. Left: positive-class F1 as a function of the fraction of patients with all c2MED features zeroed. Right: PR-AUC under the same perturbation. Standard models improve as medication information is removed, whereas the separately evaluated Drift-Resilient model declines, suggesting different dependence on medication-recording patterns.

Figure 3 shows that all five standard models improve as medication features are progressively removed. XGBoost F1-pos rises from 0.149 to 0.181, LightGBM from 0.116 to 0.150, TabPFN from 0.141 to 0.201, and TabICL from 0.166 to 0.186. This monotonic trend suggests that medication-recording associations learned from earlier years became less reliable under the 2015 observation-process shift and that removing these features partially restores classification performance.

This observation is consistent with the SHAP analysis in Section 5, which identifies medication variables among the most influential predictors for TabICL and standard TabPFN. Table 5 summarizes representative perturbation levels from the full sweep shown in Figure 3. Across all five standard models, positive-class F1 increases as medication records are progressively removed. In contrast, Drift-Resilient TabPFN exhibits a modest decline from 0.279 to 0.269 under the available 50% perturbation experiment. Although the Drift-Resilient evaluation was conducted separately and is therefore not directly comparable to the complete sweep, the opposite response direction is consistent with a qualitatively different dependence on medication-recording patterns.

Table 5. Medication-record perturbation at the 2015 stress year. Values report positive-class F1. Standard models were evaluated across the full perturbation sweep (Figure 3); representative perturbation levels are shown here. Drift-Resilient TabPFN was evaluated separately at the available 50% perturbation level.

Model	0%	50%	100%
XGBoost	0.149	0.166	0.181
LightGBM	0.116	0.137	0.150
Random Forest	0.076	0.083	0.100
TabPFN	0.141	0.168	0.201
TabICL	0.166	0.183	0.186
Drift-Resilient TabPFN	0.279	0.269	<i>not evaluated</i>

4.6 Robustness Check: One-Year-Ahead Stream Protocol

We compare the one-year-ahead stream protocol with the $W = 10$ expanding-window evaluation to test whether conclusions depend on the update policy. Figure 4 shows that the two trajectories are closely aligned after 2007.

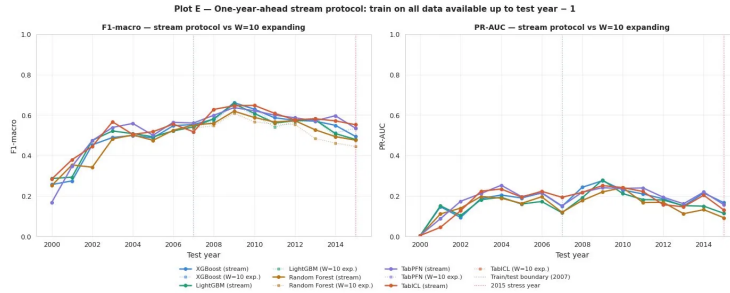


Fig. 4. One-year-ahead stream protocol (solid lines) versus $W = 10$ expanding window (dotted lines). Left: F1-macro; right: PR-AUC. The train/test boundary and the 2015 stress year are marked. Similar post-2007 trajectories indicate that the observed degradation is not specific to a single temporal update strategy.

Three patterns are consistent across protocols: stream and expanding-window trajectories are nearly indistinguishable after 2007, performance peaks around 2008–2009 and declines toward 2015, and the aggregate results in Table 6 do not materially change model ordering. These findings reduce the likelihood that the 2015 degradation is an artifact of a frozen training period or window specification.

Table 6. One-year-ahead stream results averaged over 2007–2015. Values are broadly consistent with the expanding-window analysis, indicating that continuous assimilation of historical data does not materially change the main degradation patterns.

Model	Avg. F1-macro	Avg. F1-pos	Avg. PR-AUC	Avg. ECE
XGBoost	0.577	0.256	0.202	0.236
TabPFN	0.589	0.263	0.202	0.205
TabICL	0.592	0.273	0.195	0.207
LightGBM	0.565	0.241	0.175	0.189
Random Forest	0.550	0.225	0.159	0.203

4.7 Practical Implications: Drift Regime and Objective Shape Model Selection

The results suggest that model selection under temporal shift should jointly consider the anticipated drift regime and the deployment objective. Under stable conditions and the 2012 Type I regime, TFMs and gradient boosting methods achieve broadly comparable discrimination. Under the 2015 Type II structural/observation-process regime, sliding-window gradient boosting is comparatively resilient among standard models, while Drift-Resilient TabPFN achieves the strongest annual discrimination overall. For longitudinal risk ranking, TabICL provides the highest IPCW C-index, but this does not imply reliable calibration or same-year classification.

Taken together, these findings support the paper’s central claim that robustness is not monolithic. Model superiority depends on the evaluated property, temporal regime, and operational decision. Multi-objective and shift-aware auditing is therefore necessary to avoid selecting models that appear robust under aggregate evaluation but fail on clinically relevant dimensions.

5 Why Models Fail: Domain-Aware Feature Attribution

To investigate the mechanisms behind the performance and calibration patterns observed under temporal shift, we apply a domain-aware XAI pipeline to audit model-level feature attributions. We extract global SHAP values across the longitudinal test period (2007–2015) and interpret them alongside the shift-regime and perturbation analyses reported in Section 4. These analyses are exploratory: they do not identify causal clinical effects, but help assess whether models rely on temporally stable physiological signals or on proxies that become unreliable as administrative and documentation practices change.

A comparative inspection of SHAP beeswarm plots reveals three attribution patterns.

TabICL and Standard TFMs. TabICL assigns substantial importance to medication-related variables, notably Iron supplementation and Sevelamer. These features can be clinically meaningful, but they are also closely tied to

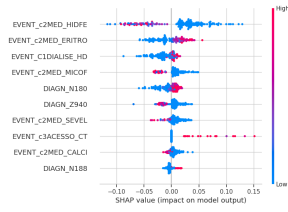


Fig. 5. TabICL assigns high importance to medication-record variables such as Iron and Sevelamer.

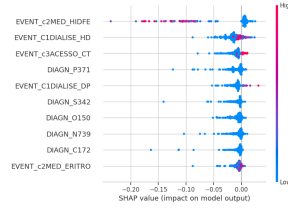


Fig. 6. Standard TabPFN distributes importance across a broader set of background variables.

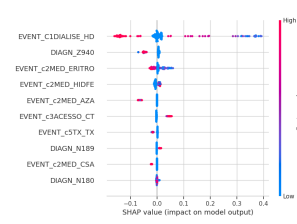


Fig. 7. Drift-Resilient TabPFN emphasizes core dialysis-related signals.

prescription protocols, reimbursement rules, and documentation practices. As a result, they may behave as temporally fragile proxies when administrative recording changes. This interpretation is consistent with the 2015 calibration degradation and with the medication-record perturbation experiment, in which removing c2MED features improves the performance of standard models. Standard TabPFN shows a different pattern: importance is more diffusely spread across a long tail of background variables. We interpret this *signal diffusion* as a potential source of instability, since small temporal changes across many weak predictors may collectively affect decision boundaries under shift.

Drift-Resilient TabPFN. In contrast, the temporally conditioned model places greater weight on core dialysis-related features, especially *EVENT_C1DIALISE_HD*. This pattern suggests that explicit temporal conditioning may reduce reliance on medication-record proxies and favor more persistent markers of disease severity. Clinically, hemodialysis has a dual interpretation: repeated sessions reflect ongoing treatment and care access, while dependence on renal replacement therapy also indicates severe baseline renal disease. By anchoring more strongly on such stable physiological signals, Drift-Resilient TabPFN provides a plausible explanation for its stronger annual discrimination under the 2015 stress regime.

5.1 Clinical Coherence of Learned Representations

The SHAP interaction analysis further suggests that the temporally conditioned model captures clinically plausible relationships. Higher dialysis frequency and erythropoietin administration appear as protective or stabilizing factors, consistent with treatment adherence and anemia management. Kidney transplantation is associated with extended survival, reflecting restoration of renal endocrine and metabolic functions. Conversely, dialysis catheter use is associated with a higher risk, consistent with complications related to central venous access, including infection and cardiovascular burden.

These patterns should be interpreted as evidence of clinical coherence rather than causal effect estimation. Their value lies in showing that the drift-aware

model tends to emphasize variables whose meanings are more stable across time, whereas standard models place greater weight on medication-record patterns that become unreliable under the 2015 observation-process shift. The perturbation experiment in Section 4.5 provides a complementary interventional stress test of this proxy-reliance hypothesis.

5.2 Predictive Uncertainty Under Shift

We also compute predictive entropy over the test period as an exploratory proxy for model uncertainty. For Drift-Resilient TabPFN, entropy increases toward 2015, suggesting that the model partially represents the late-period distributional change as increased uncertainty rather than only as overconfident error. This behavior is desirable in deployment settings because rising uncertainty may indicate cases or periods that require human review, recalibration, or model updates.

This uncertainty pattern is consistent with the broader diagnostic evidence: as the observation process becomes increasingly misaligned with historical training windows, the model retains comparatively stronger annual discrimination, but signals reduced confidence. We therefore interpret entropy not as a complete uncertainty quantification method, but as a deployment-relevant warning signal that can complement calibration, drift monitoring, and feature-attribution diagnostics under severe temporal shift.

6 Conclusion and Future Work

As Tabular Foundation Models move toward real-world deployment, evaluating robustness under dataset shift becomes essential, especially in clinical settings where protocols, documentation practices, and patient populations evolve over time. We presented a 19-year longitudinal evaluation of classical ensembles, standard TFMs, and a drift-aware TFM on a national renal registry, using uniform temporal protocols and a multi-objective auditing framework spanning discrimination, calibration, and survival ranking.

Our results show that robustness is not monolithic. Models that perform well in global risk ranking may still fail in annual discrimination or probability calibration, and model rankings vary across stable, covariate-shift, and structural/observation-process-shift regimes. The analyses further suggest that standard models are vulnerable when they rely on medication-record proxies whose meaning changes over time, whereas temporal conditioning can encourage reliance on more stable physiological signals. These findings highlight the need to evaluate tabular models not only by aggregate accuracy, but also by how they behave across objectives, temporal regimes, and clinically meaningful failure modes.

Several limitations remain. The proposed drift taxonomy and TRI are exploratory and should be validated on additional longitudinal datasets. Our evidence comes from a single national registry and should be extended to

multi-center health records and non-clinical domains. Future work should also integrate calibration-aware training, uncertainty-guided abstention, and adaptive updating strategies informed by shift-regime diagnostics. Overall, our study provides a practical framework for auditing foundation models under long-horizon dataset shift and for guiding safer deployment in dynamic tabular environments.

References

1. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794 (2016). <https://doi.org/10.1145/2939672.2939785>
2. Finlayson, S.G., Subbaswamy, A., Singh, K., Bowers, J., Kupke, A., Zittrain, J., Kohane, I.S., Koh, P.W.: A unifying view on dataset shift in medical AI. *Nature Communications* **12**(1), 2776 (2021). <https://doi.org/10.1038/s41467-021-22617-6>
3. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM Computing Surveys* **46**(4), 1–37 (2014). <https://doi.org/10.1145/2523813>
4. Guerra Junior, A.A., Pereira, R.G., Gurgel, E.I., Leal Cherchiglia, M., Dias, L.V., Ávila, J.D., Santos, N., Reis, A., de Assis Acurcio, F., Meira Junior, W.: Building the national database of health centred on the individual: Administrative and epidemiological record linkage—brazil, 2000–2015. *International Journal of Population Data Science* **3**(1), 446 (2018). <https://doi.org/10.23889/ijpds.v3i1.446>
5. Helli, K., Schnurr, D., Hollmann, N., Müller, S., Hutter, F.: Drift-resilient TabPFN: In-context learning temporal distribution shifts on tabular data. In: *Advances in Neural Information Processing Systems*. vol. 37, pp. 98742–98781 (2024)
6. Hollmann, N., Müller, S., Eggensperger, K., Hutter, F.: Accurate predictions on small data with a tabular foundation model. *Nature* **637**(8045), 319–326 (2025). <https://doi.org/10.1038/s41586-024-08328-6>
7. Jiang, J., Liu, S., Cai, H., Zhou, Q., Ye, H.J.: Representation learning for tabular data: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2026). <https://doi.org/10.1109/TPAMI.2026.3657217>
8. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems*. vol. 30 (2017)
9. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: *Advances in Neural Information Processing Systems*. vol. 30 (2017)
10. Qu, J., Holzmüller, D., Varoquaux, G., Morvan, M.L.: TabICL: A tabular foundation model for in-context learning on large data. *arXiv preprint arXiv:2502.05564* (2025)
11. Tsymbal, A.: The problem of concept drift: Definitions and related work. Tech. rep., Department of Computer Science, Trinity College Dublin (2004)
12. Uno, H., Cai, T., Pencina, M.J., D’Agostino, R.B., Wei, L.: On the C-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in Medicine* **30**(10), 1105–1117 (2011). <https://doi.org/10.1002/sim.4154>
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: *Advances in Neural Information Processing Systems*. vol. 30 (2017)