Drift Happens: An Empirical Study of Neural Architecture Robustness to Temporal Distribution Shift

Robin Holzinger* **Riccardo Colletti***

Department of Electrical Engineering and Computer Sciences,
University of California, Berkeley, USA

Extended version. Accepted at QCDS 2026; the proceedings version will appear in Springer LNCS.

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.

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.

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 [39] 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 multilayer perceptrons, convolutional

*Both authors contributed equally.

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, 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 [3; 14; 23], with limited attention to drift that unfolds sequentially over time, particularly in non-generated, in-the-wild settings [39].

Following established terminology in concept drift research [11], 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 $P(Y \mid X)$ 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 $P(Y \mid X)$, 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 [12] or online platforms.

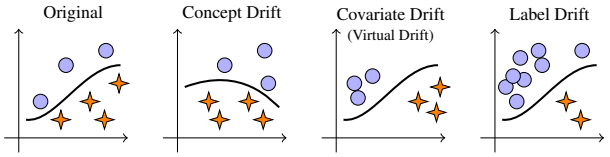


Figure 1: Drift types in a binary classification setting. Circles and stars indicate the label classes; the curve represents the decision boundary. *Concept drift* induces a change in the decision boundary, *Covariate drift* (virtual drift) changes the input distribution, and *Label drift* alters the relative class frequency [19].

In practice, these forms of drift rarely occur in isolation and often manifest in combination [2]. 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 [3] and out-of-distribution generalization [14], formalizing shift as a transition between a small number of discrete source and target domains. The introduction of large-scale benchmarks such as `WILDS` [23], which focuses on naturally occurring distribution shifts without explicit temporal indexing, and more recently `Wild-Time` [39], 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 [29; 1]. Recent empirical analyses have characterized how concept drift manifests in practice, including its locality and temporal progression across large-scale data streams [2]. 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 [26; 27; 34; 4; 19].

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 equivari-

ance, recurrent networks capture sequential structure [18; 5], and Transformer-based encoders rely on self-attention with minimal structural priors [35]. Likewise, large-scale pretraining in vision and language [8; 24] 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 [26; 27], continual-learning pipelines [34; 4; 19], or accuracy-aware data maintenance [38], 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.

3.1.1. Yearbook: A Century of Portraits

The `Yearbook` dataset [13] 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 [39] uses a pre-/post-1970 split and reports marked out-of-distribution degradation, and Modyn [4] observes accuracy decay as the train–test time gap grows. We reproduce this trend (Fig. 2), consistent with covariate and concept drift.

3.1.2. Amazon Reviews 2023: E-commerce

The `Amazon Reviews` dataset [20] 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.

3.1.3. arXiv: Scientific Discourse

The `arXiv` dataset [7] 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 [19]. Wild-Time [39] 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.

3.2.1. 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 [15] that ease optimization at greater depth. At the other extreme, the *vision Transformer* (`ViT`) drops convolution for self-attention over patches [9], 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` [28] and `DINOv3-S` [32] are pretrained on large curated image collections; `CLIP-B32` [30] and `SigLIP-B` [40] use contrastive image-text pretraining, the latter with a sigmoid loss; `ConvNeXt-S` [25], `ResNet50-IN` [15], and `ViT-S16-IN21k` [9] are trained with supervised `ImageNet` labels, the last on `ImageNet-21k`; and `MAE-B` [16] and `EVA02-B` [10] 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.

3.2.2. 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`) [21] 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` [5], a single-layer bidirectional `LSTM` [18], and a two-layer bidirectional `LSTM` with attention. The *Transformer* encoders replace recurrence with self-attention [35] 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` [8], `RoBERTa` [24], `DeBERTa-v3` [17], `ELECTRA` [6], `MPNet` [33], `ModernBERT` [37], and `DistilBERT` [31] on both text tasks, plus `MiniLM-L6` [36] 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 $\mathcal{T} = \{T_1, \ldots, T_K\}$, 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:

$$M_{ij} = \text{perf}(f_i, \mathcal{D}_j) \tag{1}$$

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 [22] 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 $\{f_1, \ldots, f_K\}$ 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`, each paper may belong to multiple subject categories. We keep only seven leaf categories as the label space, retaining papers that carry at least one of them. The label distribution remains skewed, with negative examples substantially outnumbering positives for each category. We therefore use a weighted binary cross-entropy with logits, applied independently to each of the $C = 7$ categories:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C}\big[w_c\, y_{ic}\log(\sigma(z_{ic})) + (1-y_{ic})\log(1-\sigma(z_{ic}))\big], \tag{2}$$

where z_{ic} denotes the logit for class c, $\sigma(\cdot)$ is the sigmoid function, and $y_{ic} \in \{0, 1\}$ is the binary label. The class weights w_c are defined as

$$w_c = \frac{|\{i : y_{ic} = 0\}|}{|\{i : y_{ic} = 1\}|}, \tag{3}$$

i.e., the ratio of negative to positive examples for each category. This reweighting compensates for label imbalance by amplifying the contribution of rare positive labels, preventing the model from achieving deceptively high accuracy by predicting the all-zero vector.

For *regression* on `Amazon Reviews`, we train models to predict the star rating using a weighted mean squared error:

$$\mathcal{L}_{\text{WMSE}} = \frac{1}{n}\sum_{i=1}^{n} w_{y_i}(y_i - \hat{y}_i)^2, \tag{4}$$

where $y_i \in \{1, 2, 3, 4, 5\}$ is the true rating and $\hat{y}_i$ is the prediction. The rating-specific weights w_r are defined as

$$w_r = \frac{n}{n_r}, \tag{5}$$

with n the total number of samples and n_r the number of samples with rating r. Because the empirical rating distribution is heavily skewed toward high scores, an unweighted `MSE` would be dominated by the majority class and largely ignore rare low-rating events. The inverse-frequency weighting counteracts this imbalance, ensuring that errors on minority ratings remain visible in the objective and that temporal degradation in performance cannot be explained solely by changes in the prevalence of positive reviews.

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 the cells that are filled, since a train-test pair with no completed run leaves its cell empty. The *in-distribution* score is the average of the diagonal,

$$\mathrm{ID}(M) = \mathrm{mean}_i\, M_{ii}, \tag{6}$$

a model's performance on the same period it was trained through. The *future* score averages the cells with $j > i$, where the evaluation period falls after the training cutoff,

$$\mathrm{Fut}(M) = \mathrm{mean}_{j>i}\, M_{ij}. \tag{7}$$

The *decay* is the difference between the two, oriented so a larger value always means worse temporal robustness, since accuracy and $\overline{\mathrm{AUC}}$ are better when high while $\mathrm{MSE}_{\mathrm{bal}}$ is better when low,

$$\mathrm{Dec}(M) = \begin{cases} \mathrm{ID}(M) - \mathrm{Fut}(M) & (\text{accuracy, } \overline{\mathrm{AUC}}), \\ \mathrm{Fut}(M) - \mathrm{ID}(M) & (\mathrm{MSE}_{\mathrm{bal}}). \end{cases} \tag{8}$$

so a positive decay is a loss of performance on future periods and a negative one, which is 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 ($\mathrm{mean}_{j>i}\, M_{ij}$) 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 $\mathcal{C}$ of models into the *cohort-mean matrix*

$$\bar{M}_{ij} = \frac{1}{|\mathcal{C}|} \sum_{m \in \mathcal{C}} M_{ij}^{(m)}, \tag{9}$$

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.

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 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).

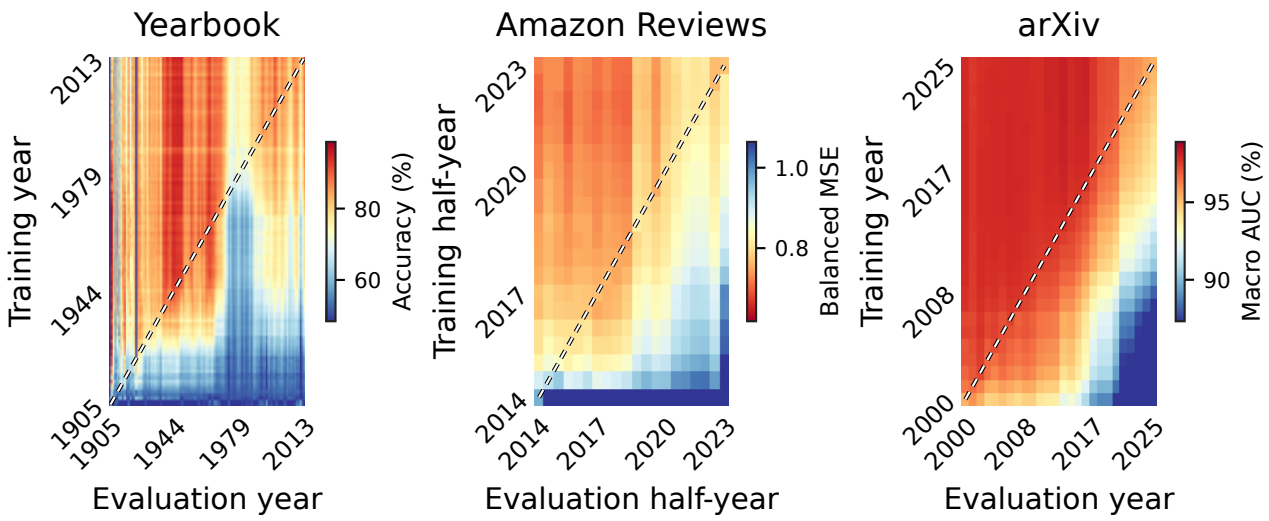


Figure 2: Cohort-mean drift matrices, one panel per domain. Each cell (i, j) is the mean performance across all models of that dataset when trained on data through period i (row) and evaluated on period j (column): accuracy for `Yearbook`, balanced MSE for `Amazon Reviews` (lower is better), and macro AUC for `arXiv`. The dashed diagonal marks in-distribution evaluation; the lower-right region of each panel is forward generalization to unseen future periods. Vertical white bands mark periods with insufficient samples.

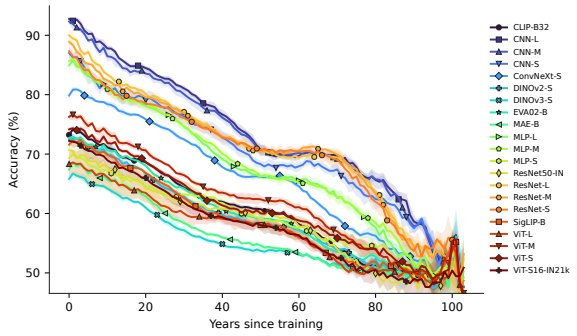


Figure 4: Forgetting curves on `Yearbook`. Each curve is one model's mean accuracy as the gap between its training year and the evaluation year grows, averaged over training years and seeds. A zero gap is in-distribution, and the slope is the rate of forgetting (Section C.7).

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.

5.1.1. 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.

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.

5.2.1. 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

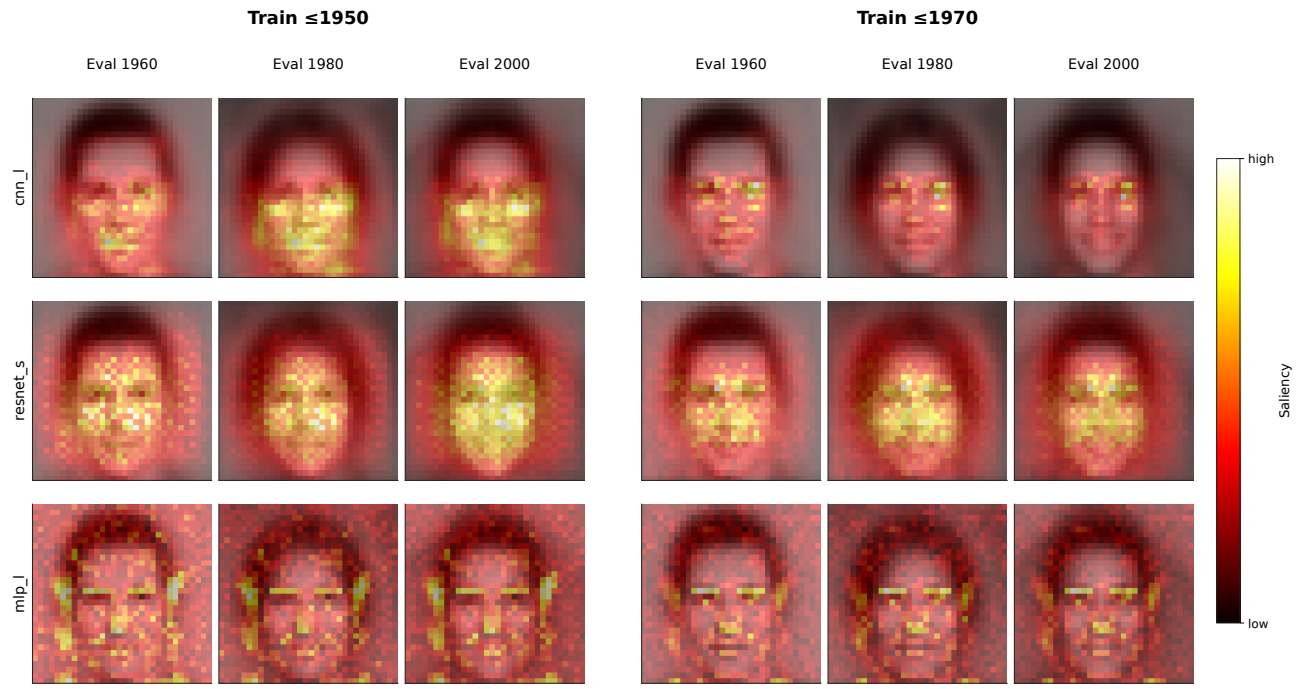


Figure 3: Gradient saliency maps on `Yearbook` for CNN-L, ResNet-S, and MLP-L, each trained through 1950 (left) and 1970 (right) and evaluated on later years. Each panel averages gradient saliency over the selected portraits from the indicated evaluation year rather than a single example.

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.

5.2.2. 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 [4].

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 [27]. 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.

References

[1] Samuel Ackerman, Orna Raz, Marcel Zalmanovici, and Aviad Zlotnick. Automatically detecting data drift in machine learning classifiers, 2021. URL https://arxiv.org/abs/2111.05672.

[2] Gabriel J. Aguiar and Alberto Cano. A comprehensive analysis of concept drift locality in data streams, 2023. URL https://arxiv.org/abs/2311.06396.

[3] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, 2010.

[4] Maximilian Böther, Ties Robroek, Viktor Gsteiger, Robin Holzinger, Xianzhe Ma, Pınar Tözün, and Ana Klimovic. Modyn: Data-centric machine learning pipeline orchestration. *Proc. ACM Manag. Data*, 3 (1), February 2025. doi: 10.1145/3709705. URL https://doi.org/10.1145/3709705.

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger

Schwenk, and Yoshua Bengio. 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)*, pages 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179.

[6] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations (ICLR)*, 2020.

[7] Cornell University. arxiv dataset, 2024. URL https://www.kaggle.com/dsv/7548853.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1423.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[10] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. EVA-02: A visual representation for neon genesis. *Image and Vision Computing*, 149:105171, 2024.

[11] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):1–37, 2014. doi: 10.1145/2523813.

[12] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. In *IEEE International Conference on Computer Vision Workshops*, pages 1–7, 2015.

[13] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, Crystal Lee, Philipp Krahenbuhl, and Alexei A. Efros. A century of portraits: A visual historical record of american high school yearbooks, 2019.

[14] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[17] Pengcheng He, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *International Conference on Learning Representations (ICLR)*, 2023.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

[19] Robin Holzinger. An analysis of drift- and cost-aware ml retraining triggering policies in modyn. Bachelor's thesis, Technical University of Munich, September 2024. Supervisors: Prof. Dr. Viktor Leis., Jana Vatter, Prof. Dr. Ana Klimović, Maximilian Böther.

[20] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

[21] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1181.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[23] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke

Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[25] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[26] Ananth Mahadevan and Michael Mathioudakis. Cost-effective retraining of machine learning models, 2023.

[27] Ananth Mahadevan and Michael Mathioudakis. Cost-aware retraining for machine learning. *Knowledge-Based Systems*, 293:111610, 2024. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2024.111610. URL https://www.sciencedirect.com/science/article/pii/S0950705124002454.

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024.

[29] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[31] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[32] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

[33] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. MPNet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[34] Huangshi Tian, Minchen Yu, and Wei Wang. Continuum: A platform for cost-aware, low-latency continual learning. In *Proceedings of the ACM Symposium on Cloud Computing*, SoCC '18, page 26–40, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450360111. doi: 10.1145/3267809.3267817. URL https://doi.org/10.1145/3267809.3267817.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[36] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[37] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*, 2024.

[38] Sarah Wooders, Xiangxi Mo, Amit Narang, Kevin Lin, Ion Stoica, Joseph M. Hellerstein, Natacha Crooks, and Joseph E. Gonzalez. Ralf: Accuracy-aware scheduling for feature store maintenance. *Proc. VLDB Endow.*, 17(3):563–576, nov 2023. ISSN 2150-8097. doi: 10.14778/3632093.3632116. URL https://doi.org/10.14778/3632093.3632116.

[39] Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei Koh, and Chelsea Finn. Wild-time: A benchmark of in-the-wild distribution shift over time. In *Advances in Neural Information Processing Systems*, 2022.

[40] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Appendix

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,

$$\mathrm{MSE}_{\mathrm{bal}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \mathrm{MSE}_r, \qquad \mathrm{MSE}_r = \frac{1}{|\mathcal{S}_r|} \sum_{i \in \mathcal{S}_r} (y_i - \hat{y}_i)^2,$$

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,

$$\overline{\text{AUC}} = \frac{1}{C}\sum_{c=1}^{C} \text{AUC}_c, \qquad \text{AUC}_c = \int_0^1 \text{TPR}_c(t)\, d\text{FPR}_c(t),$$

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.

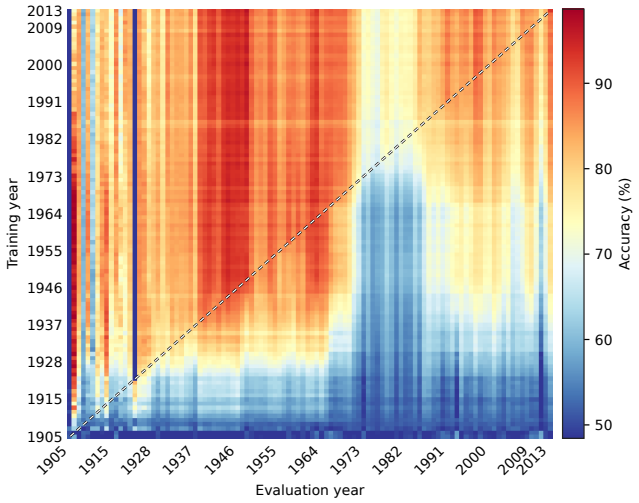


Figure 5: Cohort-mean Accuracy matrix $\bar{M}$ over the Yearbook models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j.

C.1. Model Roster

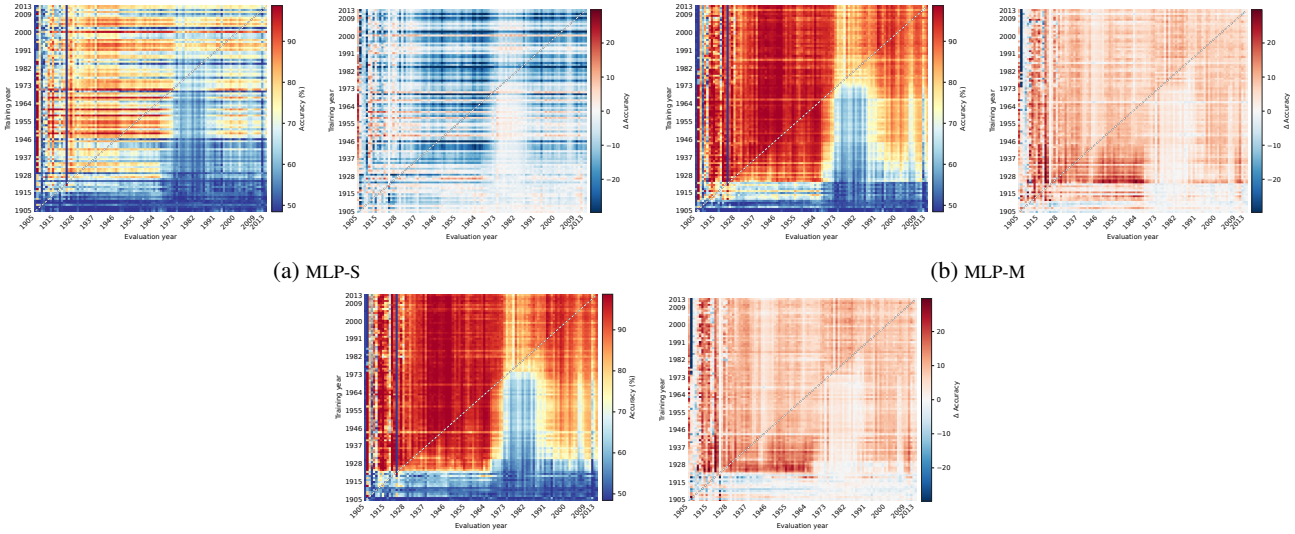
Table 3: Yearbook: models trained from scratch.

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



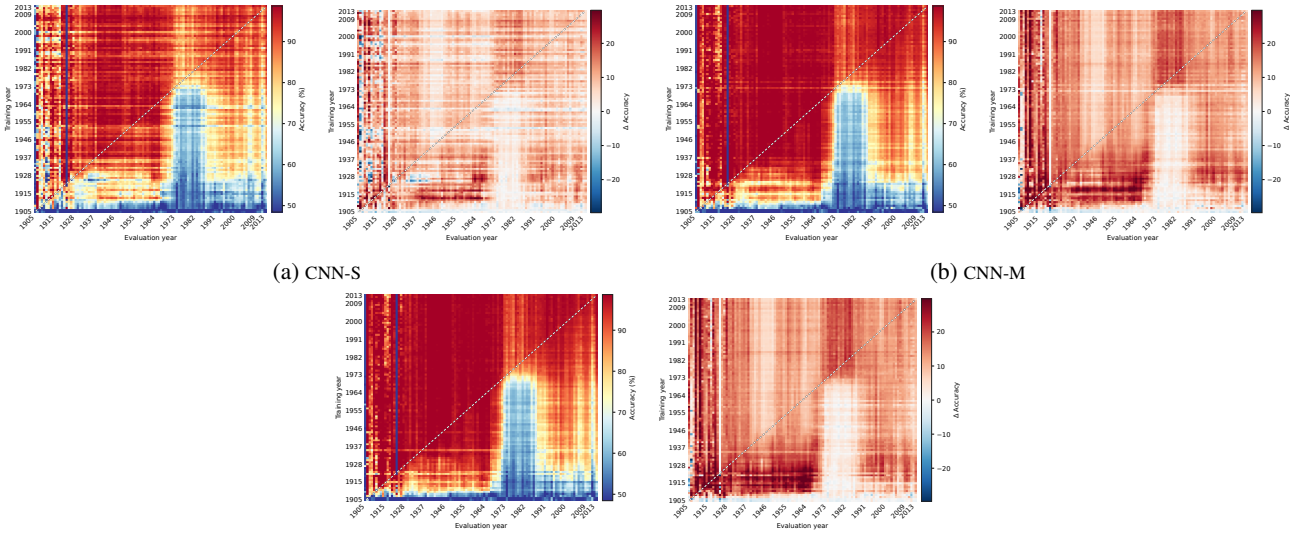
(a) MLP-S

(b) MLP-M

(c) MLP-L

Figure 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



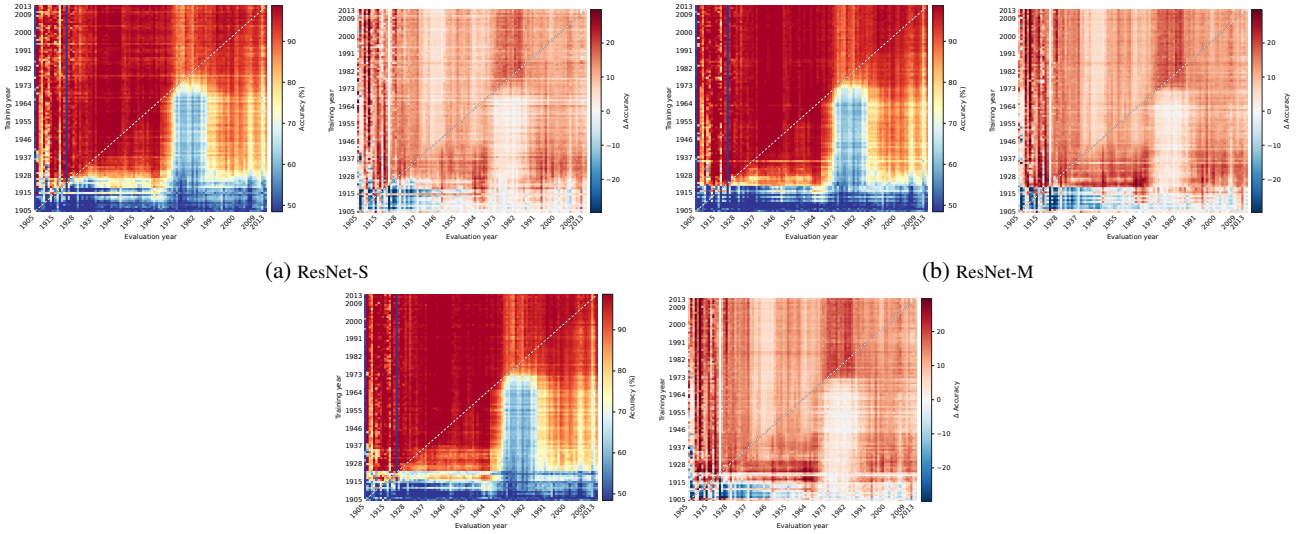
(a) CNN-S

(b) CNN-M

(c) CNN-L

Figure 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

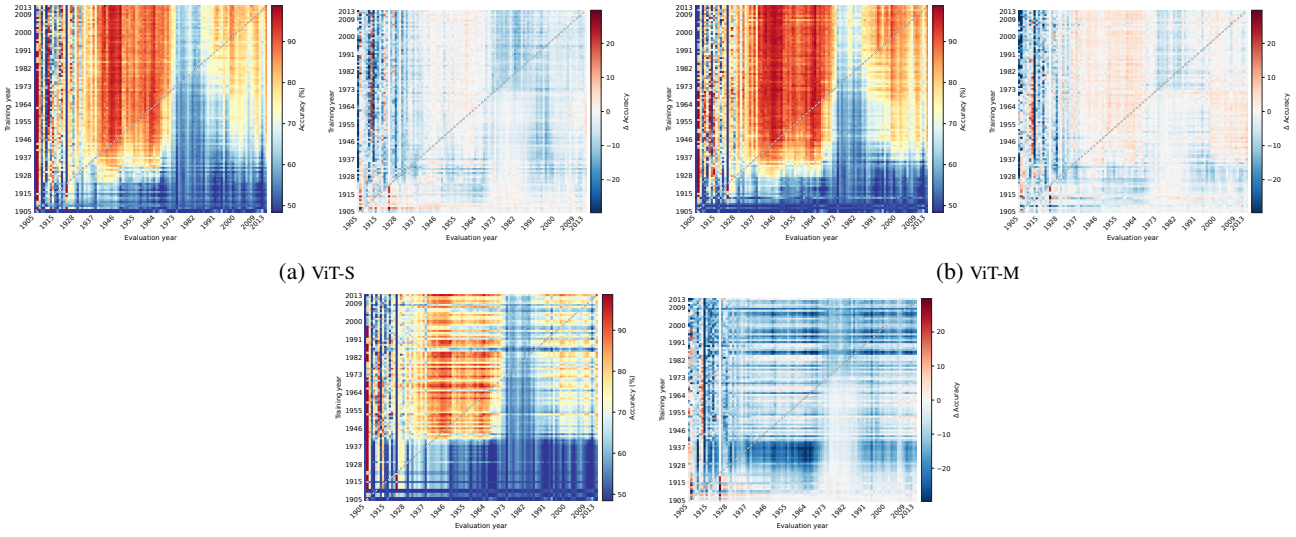


(a) ResNet-S (b) ResNet-M

(c) ResNet-L

Figure 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

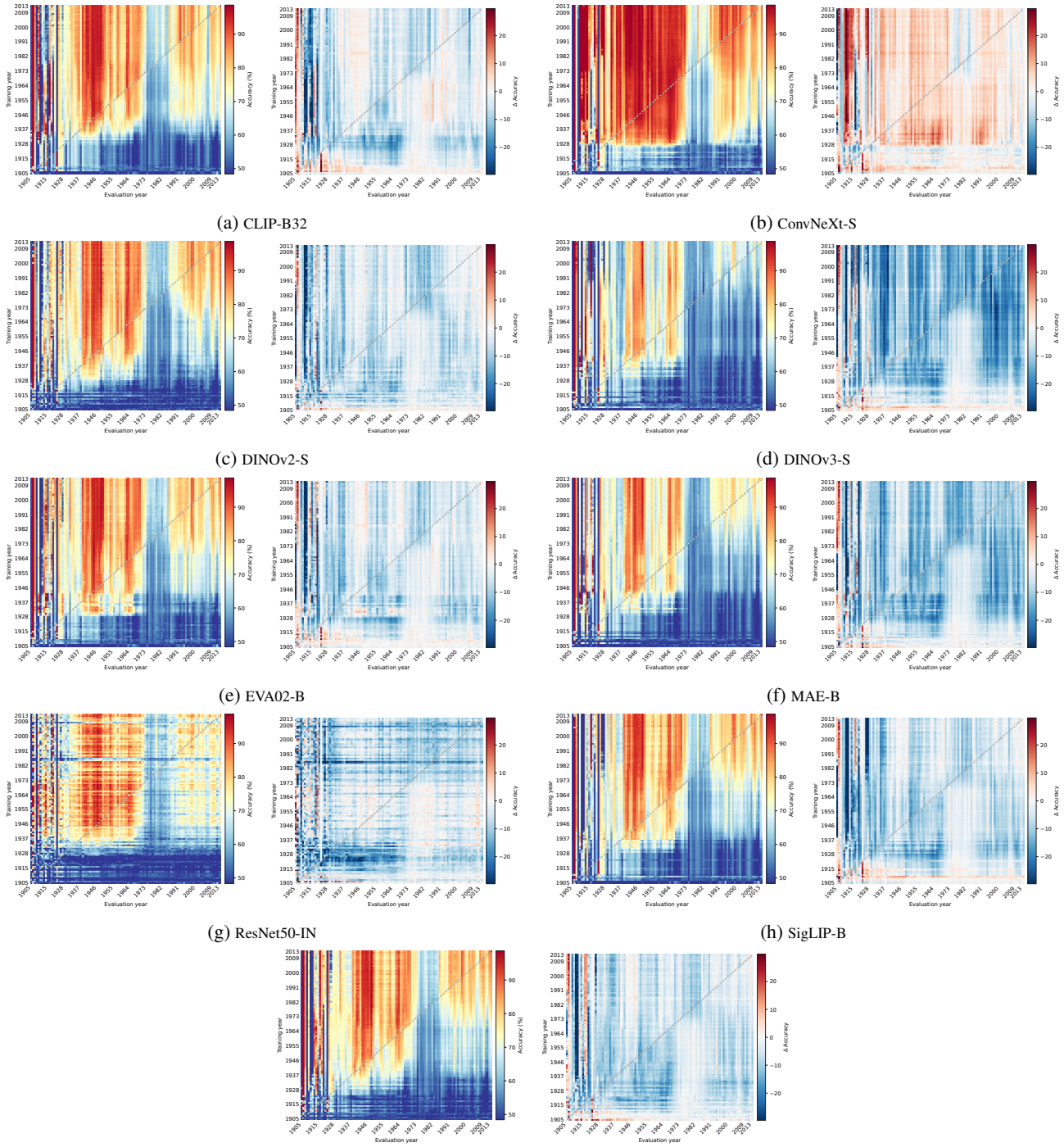


(a) ViT-S (b) ViT-M

(c) ViT-L

Figure 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



(a) CLIP-B32

(b) ConvNeXt-S

(c) DINOv2-S

(d) DINOv3-S

(e) EVA02-B

(f) MAE-B

(g) ResNet50-IN

(h) SigLIP-B

(i) ViT-S16-IN21k

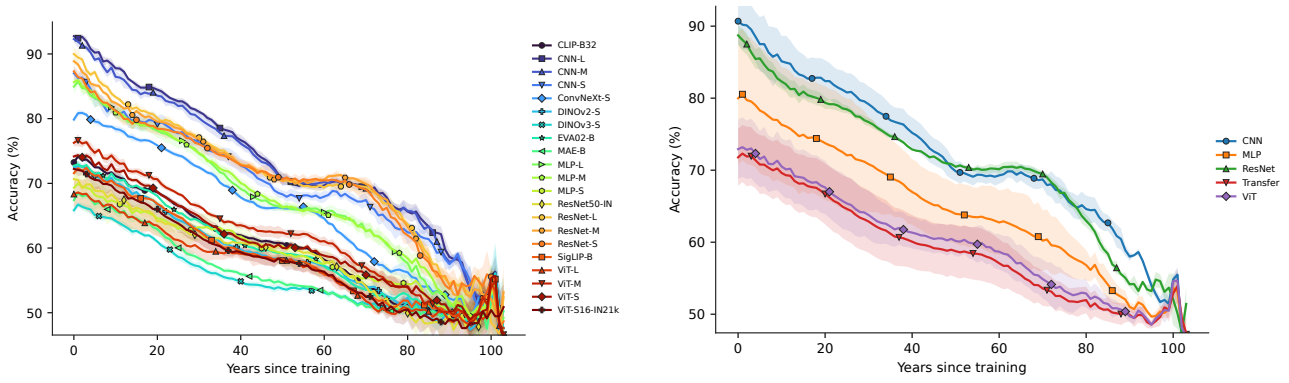
Figure 10: Transfer models: Accuracy drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

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,

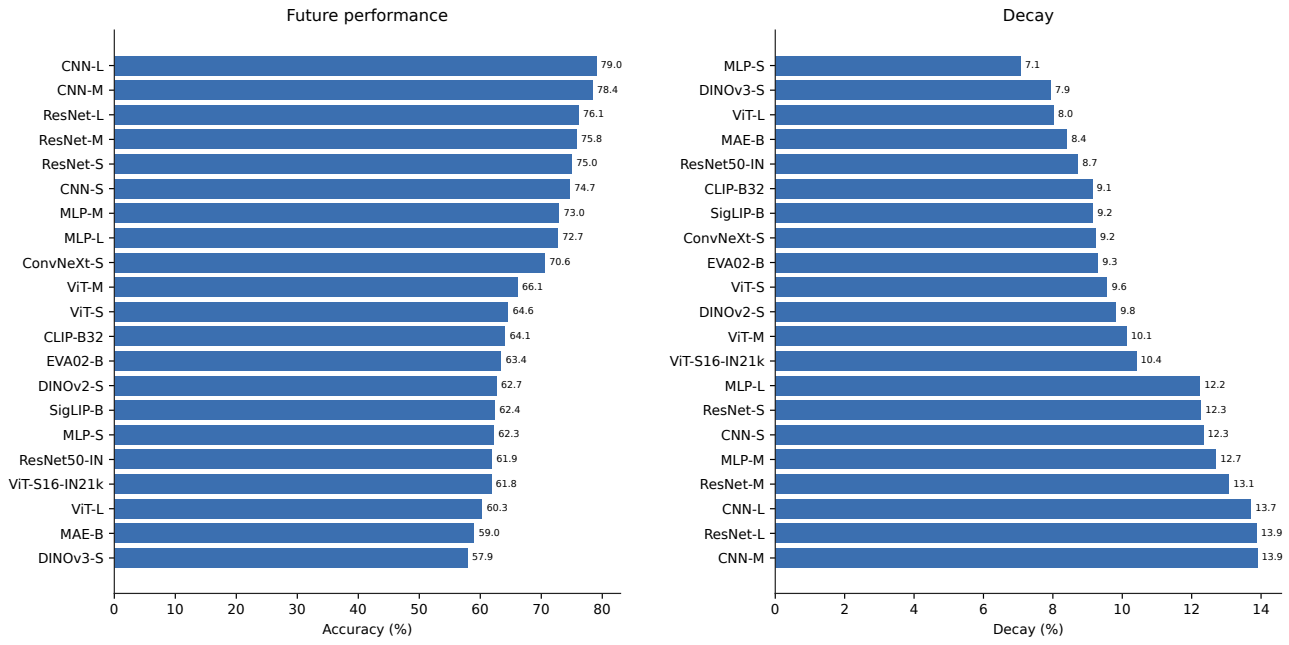
$$F(\ell) = \text{mean}_i \, M_{i,\, i+\ell}.$$

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.



(a) Per model (b) Per model family

Figure 11: Forgetting curves: each model (left) and averaged within each family (right).



(a) By future performance (b) By decay

Figure 12: Models ranked by mean future performance and by temporal decay.

C.8. Result Tables

Table 5: Temporal robustness on Yearbook.

Model	Future (%)	Decay (%)
MLP-S	62.3	7.1
DINOv3-S	57.9	7.9
ViT-L	60.3	8.0
MAE-B	59.0	8.4
ResNet50-IN	61.9	8.7
CLIP-B32	64.1	9.1
SigLIP-B	62.4	9.2
ConvNeXt-S	70.6	9.2
EVA02-B	63.4	9.3
ViT-S	64.6	9.6
DINOv2-S	62.7	9.8
ViT-M	66.1	10.1
ViT-S16-IN21k	61.8	10.4
MLP-L	72.7	12.2
ResNet-S	75.0	12.3
CNN-S	74.7	12.3
MLP-M	73.0	12.7
ResNet-M	75.8	13.1
CNN-L	79.0	13.7
ResNet-L	76.1	13.9
CNN-M	78.4	13.9

Table 6: Yearbook: models trained up to 1905, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	ViT-S16-IN21k	50.0	50.7	-0.7
2	ResNet-M	60.0	50.3	9.7
3	ResNet-L	70.0	50.1	19.9
4	MLP-L	40.0	49.8	-9.8
5	MAE-B	40.0	49.5	-9.5
6	ResNet-S	70.0	49.3	20.7
7	DINOv2-S	50.0	48.5	1.5
8	DINOv3-S	50.0	48.3	1.7
9	ViT-L	50.0	46.8	3.2
10	MLP-S	40.0	46.3	-6.3
11	ConvNeXt-S	50.0	46.1	3.9
12	ViT-M	50.0	45.7	4.3
13	CNN-L	50.0	45.4	4.6
14	ViT-S	50.0	45.1	4.9
15	ResNet50-IN	50.0	45.1	4.9
16	EVA02-B	50.0	44.9	5.1
17	MLP-M	50.0	44.9	5.1
18	CNN-S	50.0	44.9	5.1
19	CLIP-B32	50.0	44.7	5.3
20	CNN-M	60.0	44.7	15.3
21	SigLIP-B	50.0	44.7	5.3

Table 7: Yearbook: models trained up to 1944, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	ResNet-L	98.4	82.9	15.5
2	ResNet-M	98.3	81.9	16.4
3	CNN-M	98.8	81.4	17.4
4	ResNet-S	98.0	81.1	16.9
5	CNN-L	97.1	80.2	16.8
6	CNN-S	95.0	79.1	15.9
7	ConvNeXt-S	96.3	78.5	17.8
8	MLP-M	93.4	77.6	15.9
9	MLP-L	87.9	74.6	13.3
10	ViT-M	92.6	71.4	21.2
11	ViT-L	91.0	70.7	20.3
12	ViT-S	89.4	69.9	19.5
13	CLIP-B32	94.3	69.9	24.5
14	EVA02-B	93.1	67.9	25.2
15	DINOv2-S	88.5	67.1	21.5
16	SigLIP-B	88.9	66.8	22.1
17	ViT-S16-IN21k	86.4	65.0	21.4
18	ResNet50-IN	80.2	64.8	15.3
19	MAE-B	86.7	62.6	24.1
20	DINOv3-S	82.7	62.4	20.3
21	MLP-S	65.8	57.9	7.9

Table 8: Yearbook: models trained up to 1978, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	CNN-L	85.9	91.4	-5.4
2	ResNet-M	88.7	90.4	-1.6
3	ResNet-L	87.6	89.3	-1.7
4	CNN-M	86.8	88.8	-2.0
5	MLP-M	79.4	84.5	-5.1
6	ResNet-S	83.4	84.4	-1.0
7	MLP-L	75.5	83.2	-7.7
8	CNN-S	72.7	81.1	-8.5
9	ConvNeXt-S	80.6	78.6	1.9
10	ViT-M	72.4	76.8	-4.4
11	CLIP-B32	69.9	74.8	-5.0
12	EVA02-B	60.6	73.3	-12.8
13	ViT-L	66.8	72.4	-5.6
14	DINOv2-S	64.5	72.3	-7.8
15	ViT-S	64.2	71.8	-7.6
16	ViT-S16-IN21k	66.5	70.7	-4.3
17	ResNet50-IN	72.7	70.7	2.0
18	SigLIP-B	64.8	69.9	-5.1
19	MLP-S	67.9	69.4	-1.5
20	MAE-B	56.1	65.0	-8.9
21	DINOv3-S	60.3	59.9	0.4

Table 9: Yearbook: models trained up to 2012, ordered by future performance.

Rank	Model	Accuracy (%)	Future (%)	Decay (%)
1	CNN-L	93.2	98.2	-5.0
2	CNN-M	93.2	97.7	-4.6
3	ResNet-L	88.9	97.5	-8.5
4	ResNet-M	92.6	97.0	-4.3
5	ResNet-S	86.8	95.4	-8.5
6	CNN-S	83.7	93.4	-9.7
7	MLP-L	87.4	91.9	-4.6
8	MLP-M	86.3	91.7	-5.4
9	ConvNeXt-S	73.7	91.2	-17.5
10	DINOv2-S	82.6	90.0	-7.3
11	ViT-M	77.9	88.1	-10.2
12	ViT-S	80.5	88.0	-7.5
13	SigLIP-B	85.8	87.5	-1.7
14	EVA02-B	82.1	87.4	-5.3
15	CLIP-B32	74.7	86.6	-11.9
16	ViT-S16-IN21k	87.9	86.5	1.4
17	ResNet50-IN	76.8	83.8	-7.0
18	DINOv3-S	71.1	77.9	-6.9
19	MAE-B	78.4	77.4	1.0
20	MLP-S	70.0	76.1	-6.1
21	ViT-L	66.8	74.4	-7.5

Table 10: Yearbook: future performance and decay by model family.

	1905		1944		1978		2012	
Family	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.

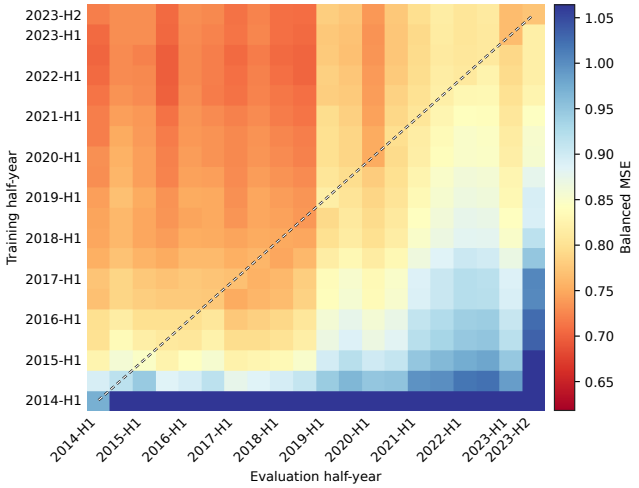


Figure 13: Cohort-mean Balanced MSE matrix $\bar{M}$ over the Amazon Reviews models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j.

D.1. Model Roster

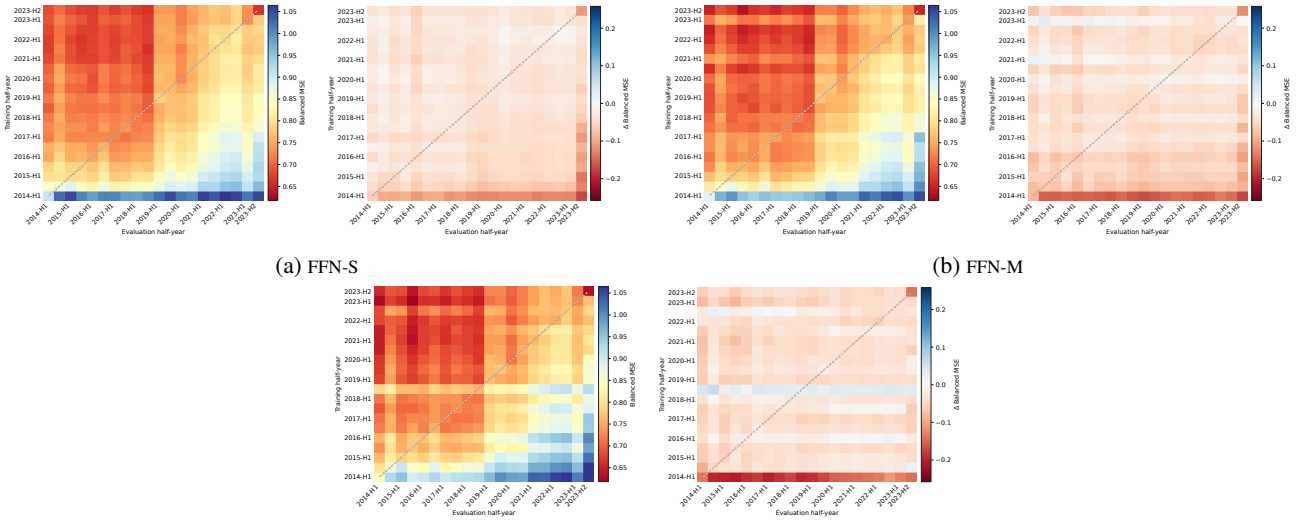
Table 11: Amazon Reviews: models trained from scratch.

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 pretrained 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



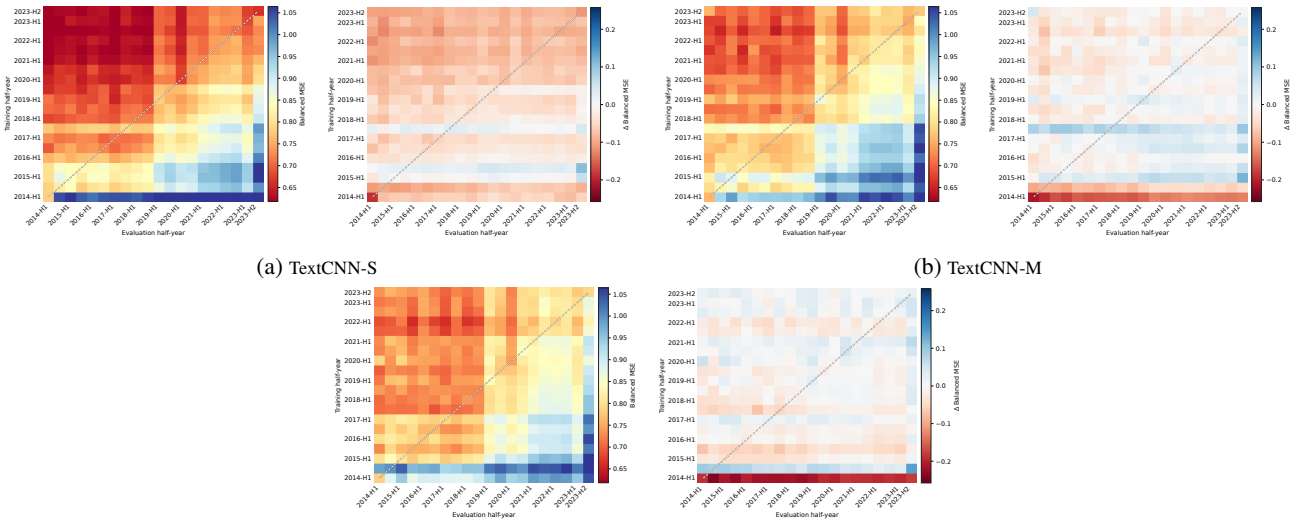
(a) FFN-S

(b) FFN-M

(c) FFN-L

Figure 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



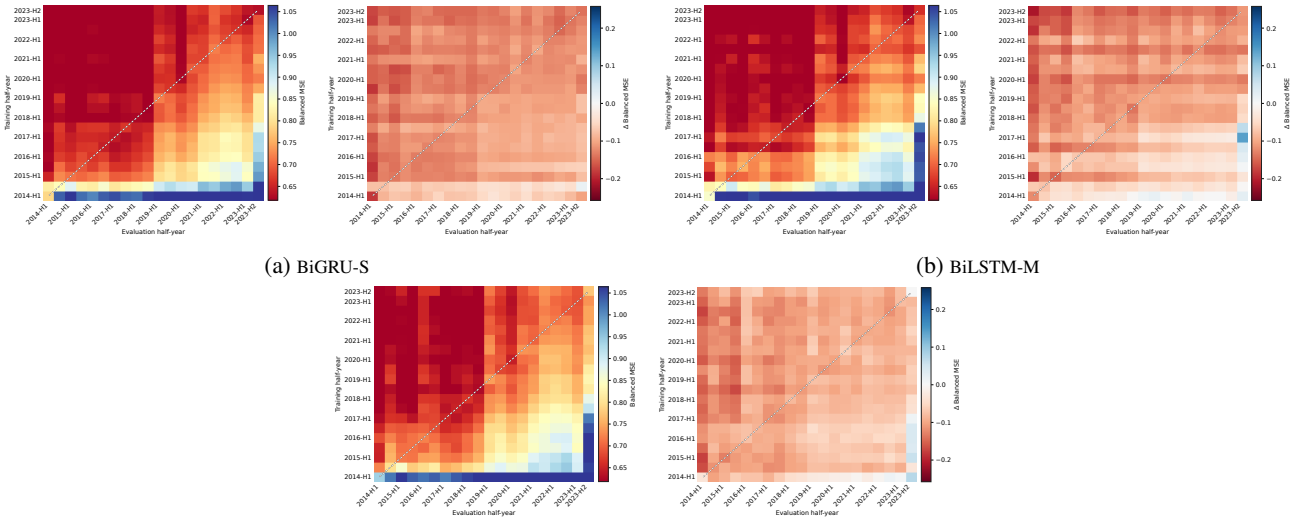
(a) TextCNN-S

(b) TextCNN-M

(c) TextCNN-L

Figure 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

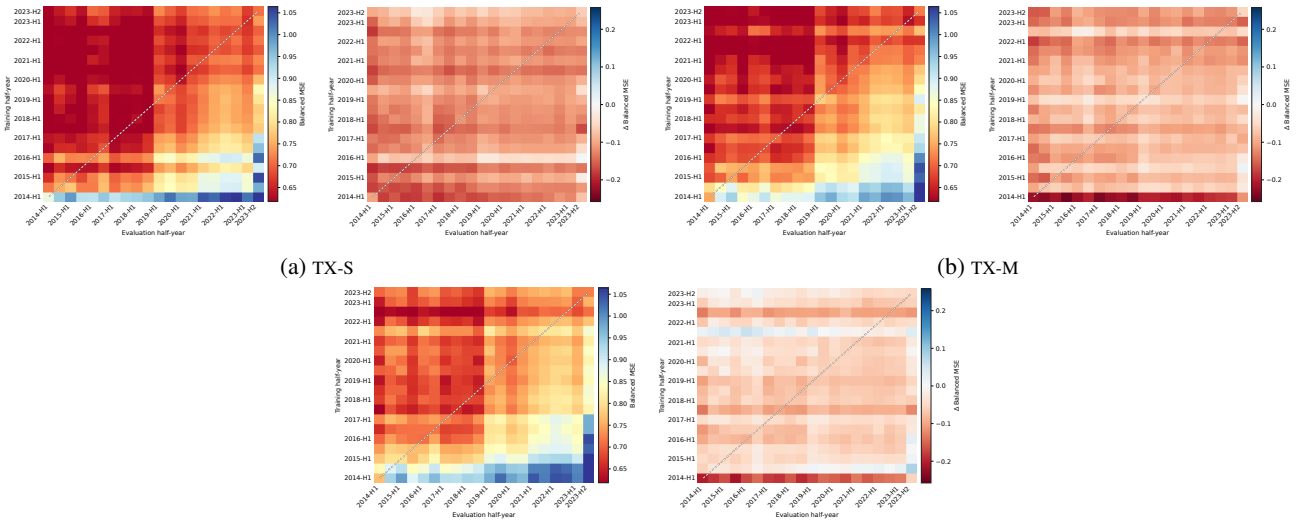


(a) BiGRU-S (b) BiLSTM-M

(c) BiLSTM-Attn-L

Figure 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

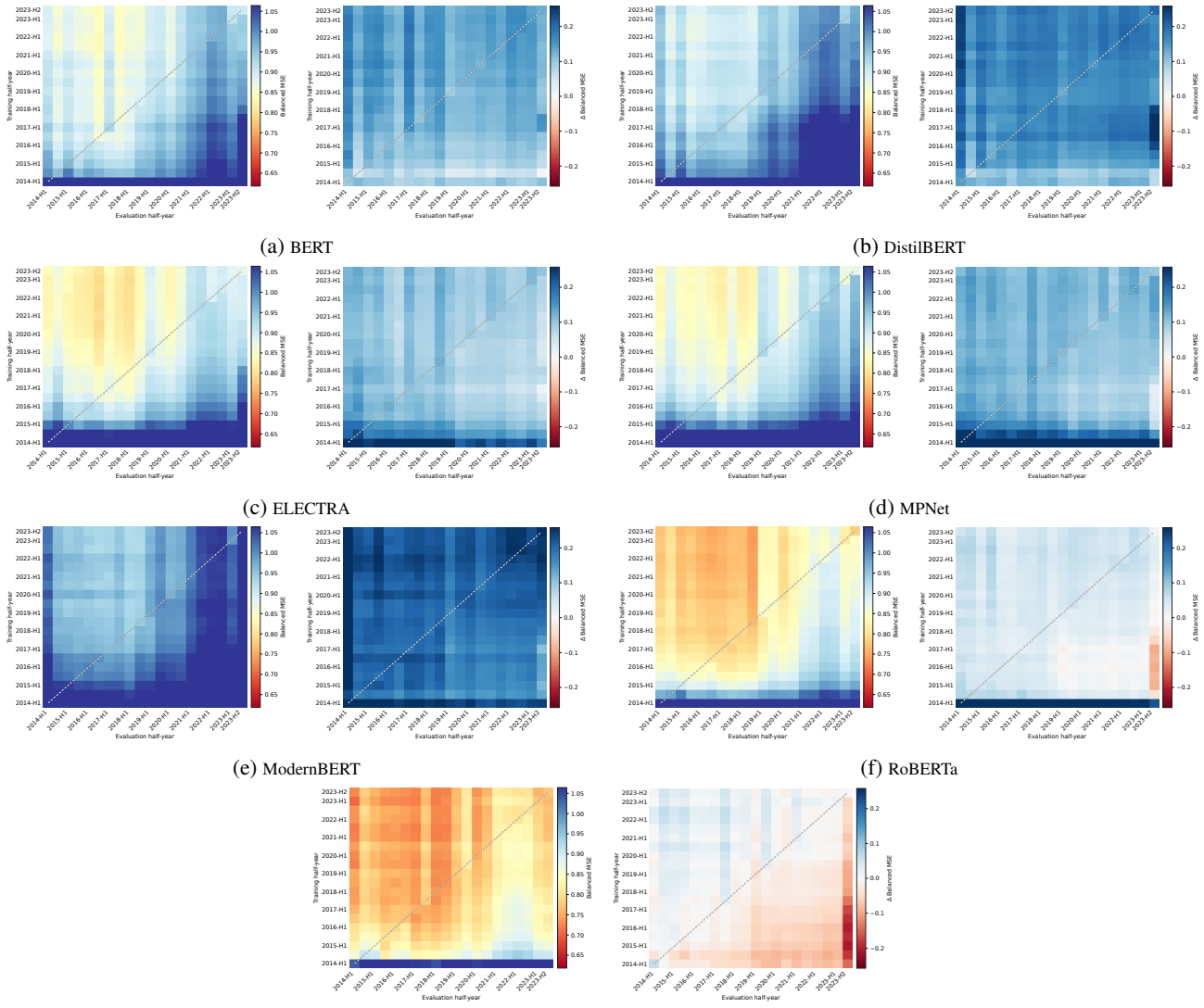


(a) TX-S (b) TX-M

(c) TX-L

Figure 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



(a) BERT (b) DistilBERT

(c) ELECTRA (d) MPNet

(e) ModernBERT (f) RoBERTa

(g) DeBERTa-v3

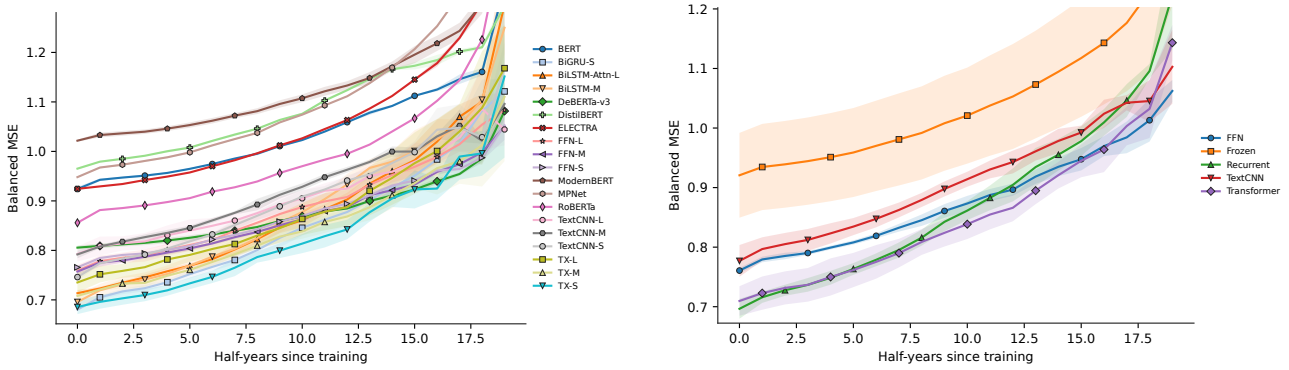
Figure 18: Frozen models: Balanced MSE drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

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,

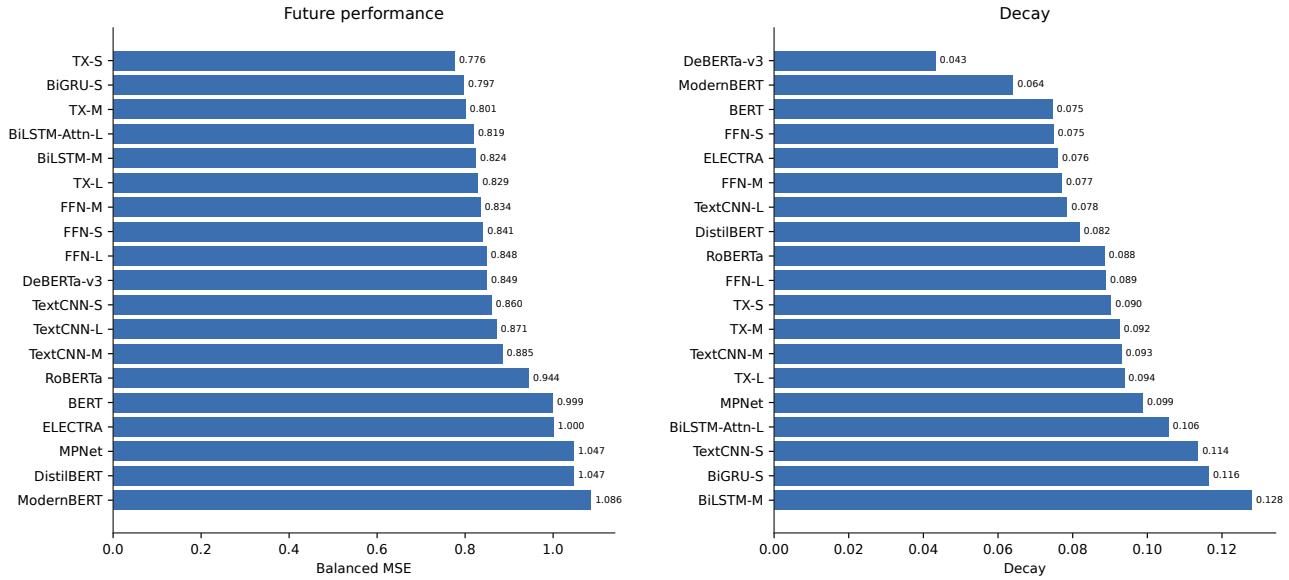
$$F(\ell) = \text{mean}_i \, M_{i,\, i+\ell}.$$

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.



(a) Per model (b) Per model family

Figure 19: Forgetting curves: each model (left) and averaged within each family (right).



(a) By future performance (b) By decay

Figure 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
DeBERTa-v3	0.849	0.043
ModernBERT	1.086	0.064
BERT	0.999	0.075
FFN-S	0.841	0.075
ELECTRA	1.000	0.076
FFN-M	0.834	0.077
TextCNN-L	0.871	0.078
DistilBERT	1.047	0.082
RoBERTa	0.944	0.088
FFN-L	0.848	0.089

Model	Future	Decay
TX-S	0.776	0.090
TX-M	0.801	0.092
TextCNN-M	0.885	0.093
TX-L	0.829	0.094
MPNet	1.047	0.099
BiLSTM-Attn-L	0.819	0.106
TextCNN-S	0.860	0.114
BiGRU-S	0.797	0.116
BiLSTM-M	0.824	0.128

Table 14: Amazon Reviews: models trained up to 2014-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay
1	TX-M	0.748	0.933	0.185
2	TextCNN-L	0.782	0.935	0.153
3	FFN-L	0.836	0.972	0.136
4	TX-L	0.767	0.981	0.215
5	FFN-M	0.885	0.986	0.100
6	TX-S	0.868	0.987	0.119
7	TextCNN-M	0.761	0.993	0.231
8	FFN-S	0.907	1.023	0.116
9	BiGRU-S	0.790	1.073	0.283
10	TextCNN-S	0.782	1.078	0.296
11	DeBERTa-v3	1.035	1.083	0.048
12	BiLSTM-Attn-L	0.938	1.105	0.167
13	BiLSTM-M	0.850	1.127	0.277
14	BERT	1.067	1.232	0.165
15	DistilBERT	1.113	1.250	0.137
16	ELECTRA	1.332	1.382	0.050
17	ModernBERT	1.273	1.386	0.113
18	RoBERTa	1.227	1.468	0.241
19	MPNet	1.492	1.657	0.165

Table 15: Amazon Reviews: models trained up to 2017-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay
1	TX-S	0.625	0.748	0.123
2	BiGRU-S	0.644	0.765	0.121
3	TX-M	0.701	0.785	0.084
4	BiLSTM-Attn-L	0.654	0.789	0.136
5	FFN-L	0.724	0.810	0.086
6	FFN-S	0.723	0.812	0.089
7	DeBERTa-v3	0.758	0.814	0.056
8	TextCNN-S	0.711	0.817	0.106
9	TX-L	0.722	0.818	0.096
10	BiLSTM-M	0.681	0.828	0.147
11	FFN-M	0.742	0.829	0.087
12	RoBERTa	0.792	0.860	0.068
13	TextCNN-L	0.789	0.876	0.087
14	TextCNN-M	0.773	0.885	0.113
15	ELECTRA	0.873	0.926	0.052
16	MPNet	0.868	0.940	0.072
17	BERT	0.929	0.984	0.055
18	ModernBERT	0.958	1.028	0.070
19	DistilBERT	0.942	1.047	0.104

Table 16: Amazon Reviews: models trained up to 2020-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay
1	BiLSTM-M	0.614	0.706	0.092
2	BiGRU-S	0.627	0.715	0.088
3	TX-S	0.637	0.721	0.084
4	BiLSTM-Attn-L	0.647	0.749	0.102
5	TX-M	0.675	0.751	0.076
6	TextCNN-S	0.672	0.767	0.094
7	TX-L	0.693	0.773	0.080
8	FFN-L	0.713	0.793	0.080
9	FFN-S	0.729	0.795	0.066
10	FFN-M	0.744	0.817	0.074
11	DeBERTa-v3	0.753	0.820	0.067
12	TextCNN-M	0.748	0.833	0.085
13	TextCNN-L	0.743	0.834	0.091
14	RoBERTa	0.800	0.870	0.070
15	ELECTRA	0.844	0.911	0.067
16	MPNet	0.860	0.938	0.078
17	BERT	0.876	0.959	0.084
18	DistilBERT	0.926	0.998	0.072
19	ModernBERT	0.969	1.041	0.072

Table 17: Amazon Reviews: models trained up to 2023-H1, ordered by future performance.

Rank	Model	Balanced MSE	Future	Decay
1	TX-M	0.634	0.665	0.031
2	BiGRU-S	0.627	0.691	0.064
3	BiLSTM-M	0.638	0.703	0.065
4	BiLSTM-Attn-L	0.687	0.735	0.048
5	TX-S	0.663	0.740	0.077
6	TextCNN-S	0.682	0.747	0.066
7	TX-L	0.699	0.759	0.060
8	DeBERTa-v3	0.776	0.764	-0.013
9	FFN-L	0.728	0.770	0.042
10	FFN-S	0.731	0.777	0.046
11	FFN-M	0.742	0.821	0.079
12	TextCNN-M	0.751	0.838	0.087
13	RoBERTa	0.812	0.847	0.035
14	TextCNN-L	0.794	0.869	0.075
15	ELECTRA	0.882	0.894	0.012
16	BERT	0.911	0.939	0.029
17	MPNet	0.863	0.956	0.093
18	DistilBERT	0.937	0.980	0.043
19	ModernBERT	0.992	1.069	0.076

Table 18: Amazon Reviews: future performance and decay by model family.

	2014-H1		2017-H1		2020-H1		2023-H1	
Family	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`.

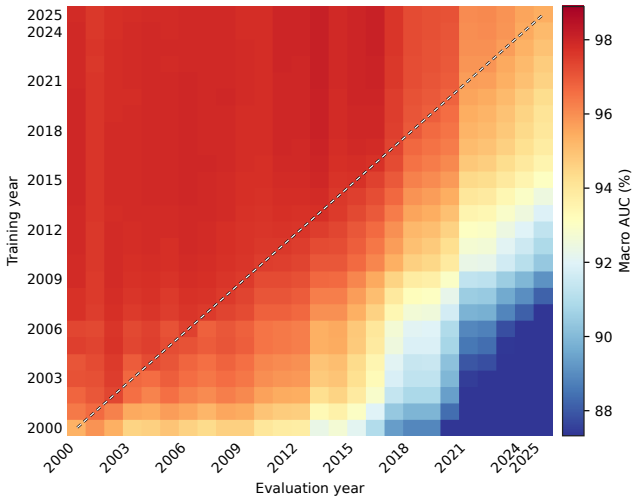


Figure 21: Cohort-mean Macro AUC matrix $\bar{M}$ over the arXiv models. Cell (i, j) is the mean across those models of the score from training through slice i and evaluating on slice j.

E.1. Model Roster

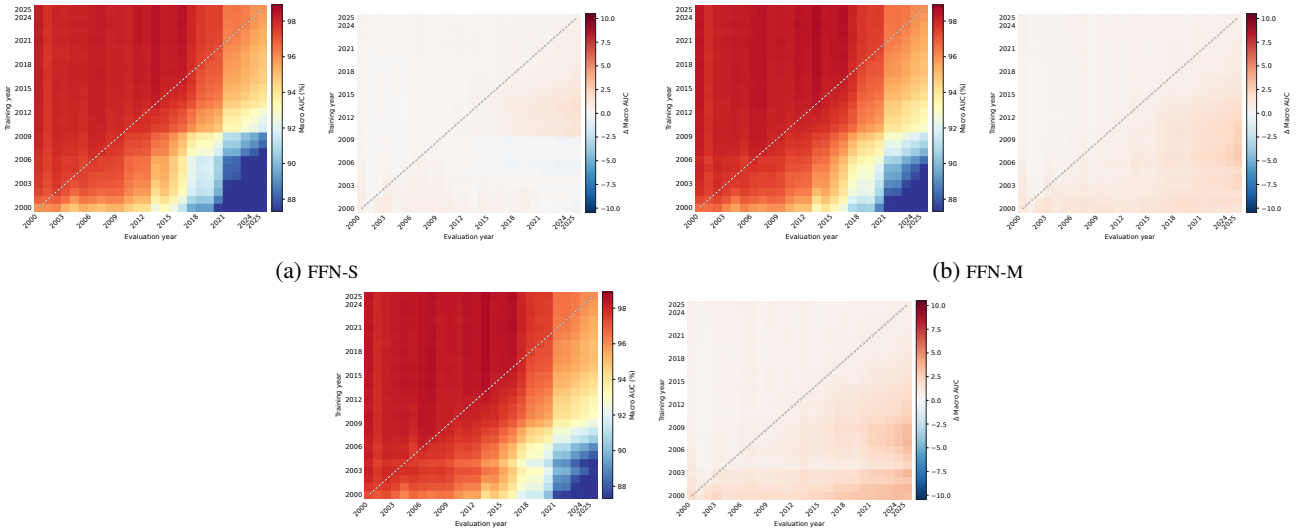
Table 19: arXiv: models trained from scratch.

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

Table 20: arXiv: frozen pretrained encoders, with trainable head and total parameters.

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



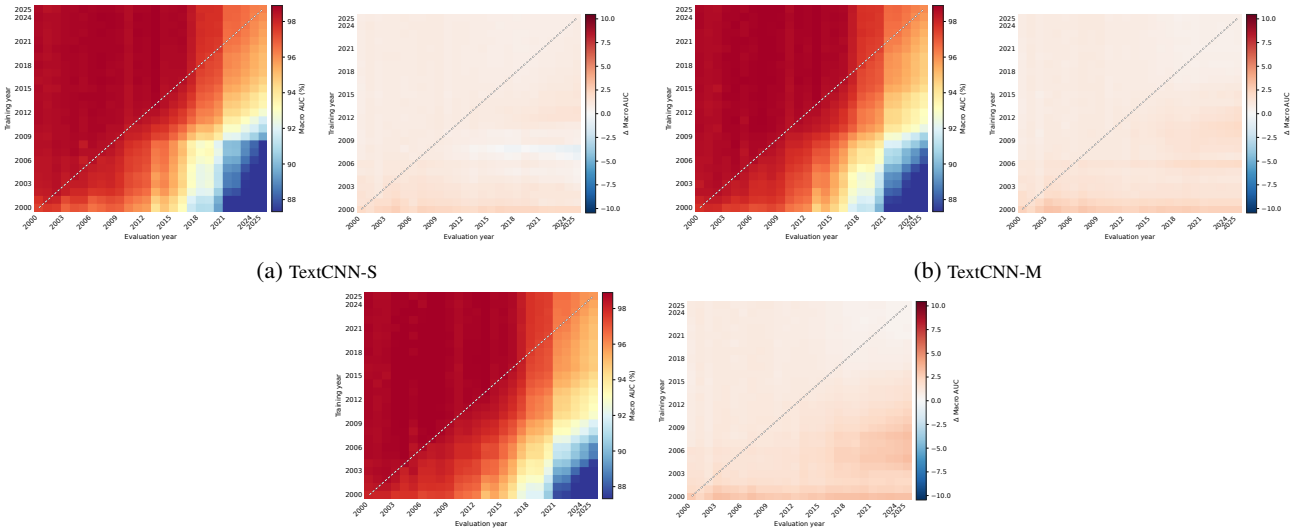
(a) FFN-S

(b) FFN-M

(c) FFN-L

Figure 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



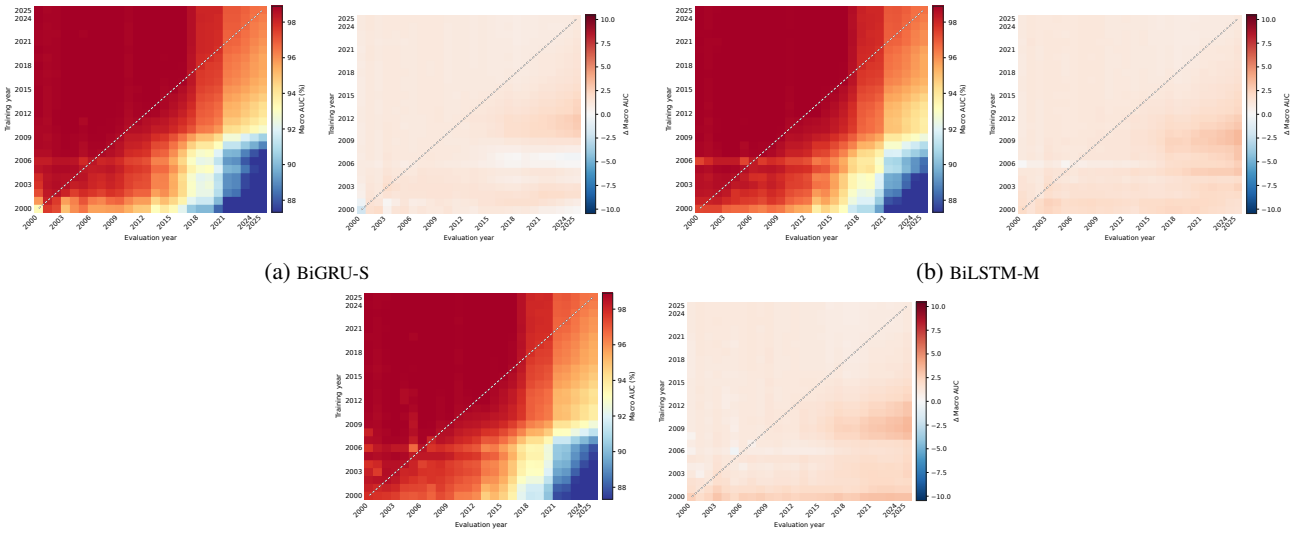
(a) TextCNN-S

(b) TextCNN-M

(c) TextCNN-L

Figure 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



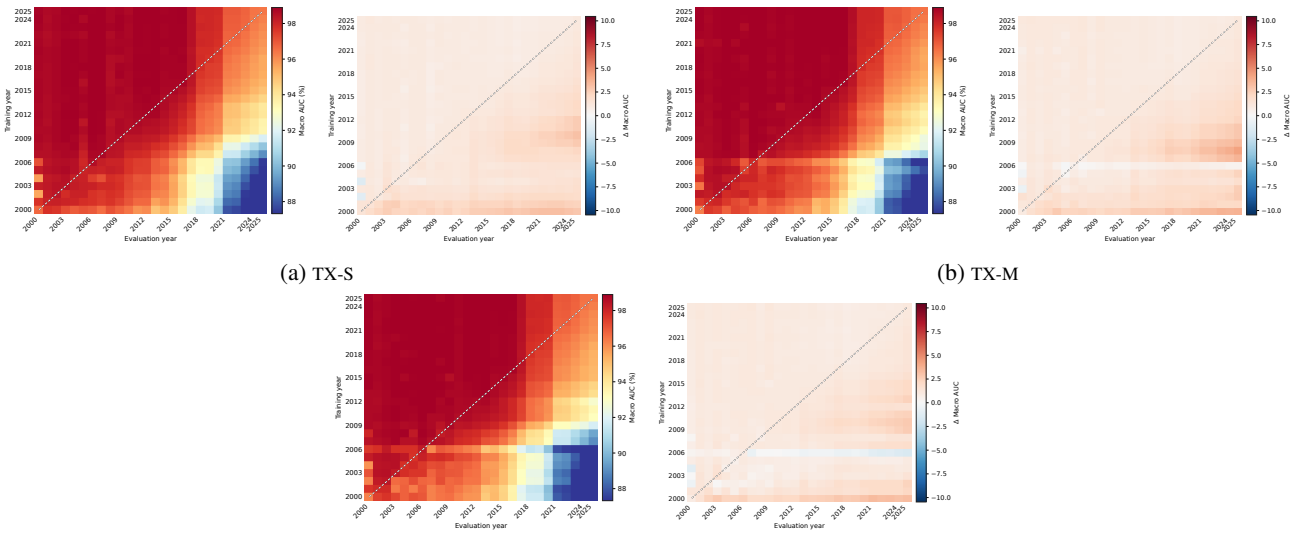
(a) BiGRU-S

(b) BiLSTM-M

(c) BiLSTM-Attn-L

Figure 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



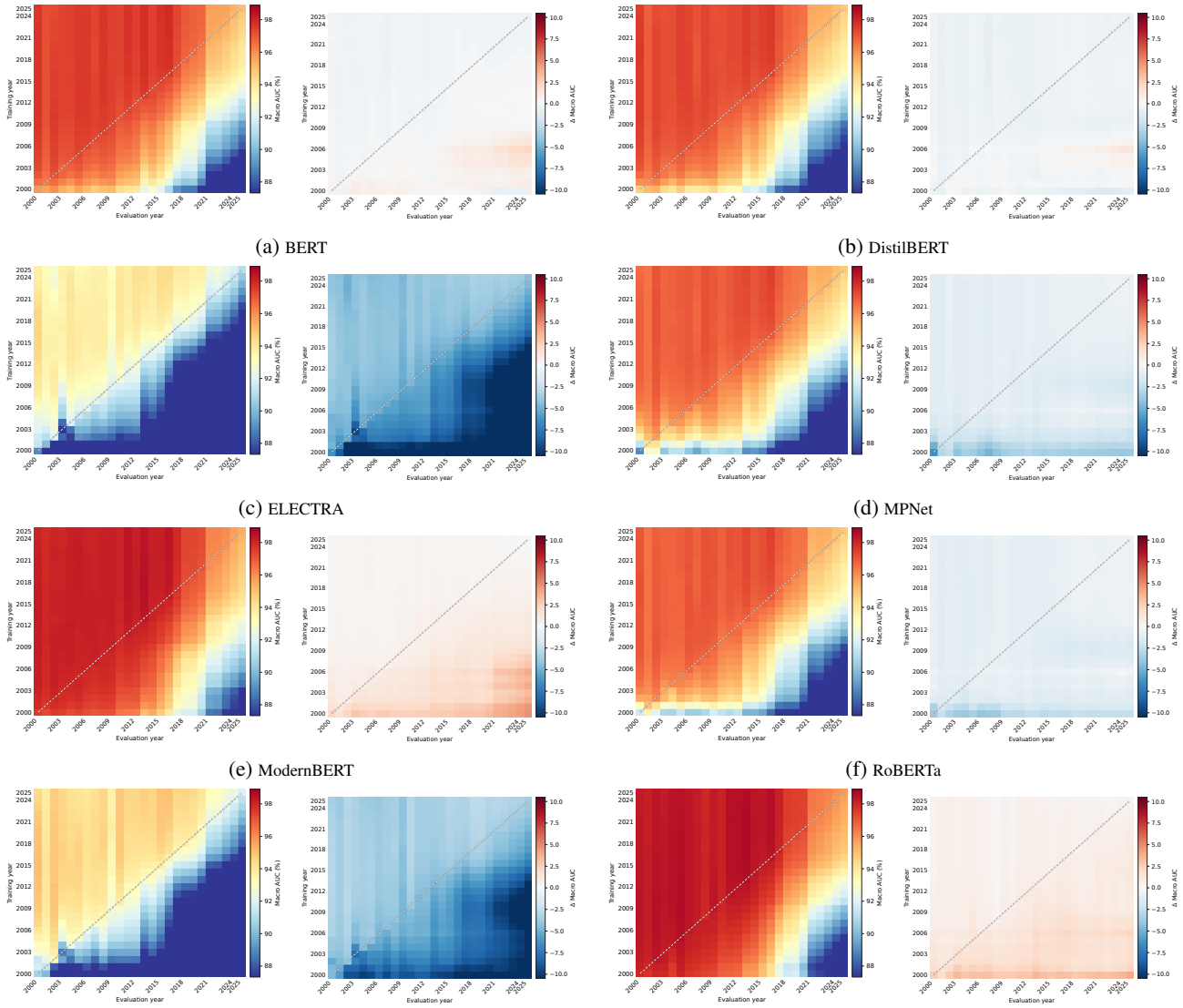
(a) TX-S

(b) TX-M

(c) TX-L

Figure 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



(a) BERT (b) DistilBERT

(c) ELECTRA (d) MPNet

(e) ModernBERT (f) RoBERTa

(g) DeBERTa-v3 (h) MiniLM-L6

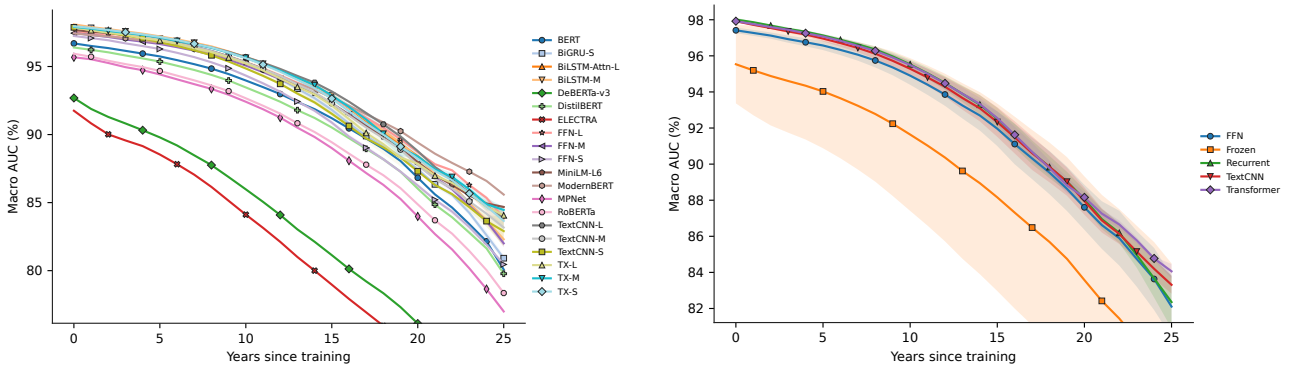
Figure 26: Frozen models: Macro AUC drift matrix $M^{(m)}$ and deviation from the cohort mean $\Delta^{(m)} = M^{(m)} - \bar{M}$ for each model, shown on a sequential and a zero-centred diverging scale, respectively.

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,

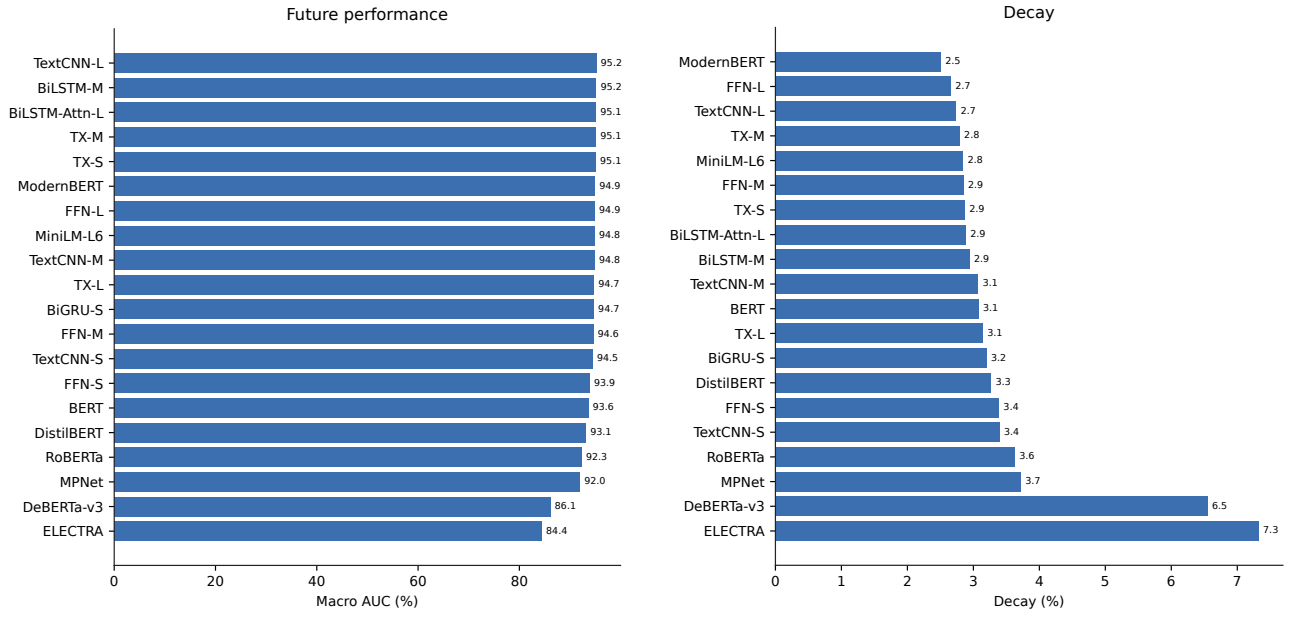
$$F(\ell) = \text{mean}_i \, M_{i,\, i+\ell}.$$

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.



(a) Per model (b) Per model family

Figure 27: Forgetting curves: each model (left) and averaged within each family (right).



(a) By future performance (b) By decay

Figure 28: Models ranked by mean future performance and by temporal decay.

E.8. Result Tables

Table 21: Temporal robustness on arXiv.

Model	Future (%)	Decay (%)
ModernBERT	94.9	2.5
FFN-L	94.9	2.7
TextCNN-L	95.2	2.7
TX-M	95.1	2.8
MiniLM-L6	94.8	2.8
FFN-M	94.6	2.9
TX-S	95.1	2.9
BiLSTM-Attn-L	95.1	2.9
BiLSTM-M	95.2	2.9
TextCNN-M	94.8	3.1
BERT	93.6	3.1
TX-L	94.7	3.1
BiGRU-S	94.7	3.2
DistilBERT	93.1	3.3
FFN-S	93.9	3.4
TextCNN-S	94.5	3.4
RoBERTa	92.3	3.6
MPNet	92.0	3.7
DeBERTa-v3	86.1	6.5
ELECTRA	84.4	7.3

Table 22: arXiv: models trained up to 2000, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	MiniLM-L6	97.9	93.9	4.0
2	ModernBERT	97.5	93.9	3.7
3	TextCNN-L	98.1	93.8	4.3
4	TX-M	96.2	93.6	2.7
5	BiLSTM-Attn-L	97.8	93.5	4.3
6	TX-S	96.6	93.5	3.2
7	TX-L	96.2	93.4	2.8
8	TextCNN-M	97.7	93.4	4.3
9	FFN-L	97.2	93.2	4.0
10	TextCNN-S	97.4	93.0	4.4
11	BiLSTM-M	97.3	92.8	4.5
12	FFN-M	96.6	92.3	4.3
13	BiGRU-S	93.3	91.9	1.4
14	FFN-S	95.9	91.3	4.6
15	BERT	95.8	90.9	4.9
16	DistilBERT	94.6	90.2	4.4
17	RoBERTa	91.5	88.0	3.5
18	MPNet	88.9	87.3	1.6
19	DeBERTa-v3	90.7	81.2	9.5
20	ELECTRA	88.9	77.3	11.7

Table 23: arXiv: models trained up to 2008, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	TX-M	98.6	96.0	2.6
2	TextCNN-L	98.7	95.7	3.0
3	BiLSTM-Attn-L	98.5	95.6	2.9
4	BiLSTM-M	98.8	95.6	3.2
5	TX-S	98.4	95.3	3.1
6	FFN-L	98.0	95.2	2.8
7	ModernBERT	98.0	95.0	3.0
8	MiniLM-L6	97.9	94.9	3.0
9	TextCNN-M	98.6	94.9	3.7
10	FFN-M	97.9	94.9	3.0
11	TX-L	98.2	94.8	3.4
12	BiGRU-S	98.6	94.5	4.2
13	BERT	97.4	93.8	3.6
14	FFN-S	97.6	93.7	4.0
15	TextCNN-S	98.5	93.6	4.9
16	DistilBERT	97.1	93.3	3.9
17	RoBERTa	96.8	92.5	4.3
18	MPNet	96.7	92.5	4.3
19	DeBERTa-v3	93.4	86.1	7.3
20	ELECTRA	92.5	84.6	7.9

Table 24: arXiv: models trained up to 2016, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	BiLSTM-M	98.8	96.6	2.2
2	BiGRU-S	98.8	96.6	2.2
3	TX-S	98.7	96.5	2.2
4	TX-M	98.7	96.5	2.2
5	TX-L	98.7	96.5	2.2
6	BiLSTM-Attn-L	98.7	96.4	2.3
7	TextCNN-L	98.6	96.3	2.3
8	TextCNN-S	98.5	96.2	2.3
9	TextCNN-M	98.5	96.2	2.4
10	FFN-L	98.3	96.2	2.2
11	FFN-M	98.2	96.1	2.2
12	MiniLM-L6	98.4	96.0	2.4
13	FFN-S	98.1	96.0	2.2
14	ModernBERT	98.2	95.8	2.4
15	BERT	97.6	95.1	2.5
16	DistilBERT	97.3	94.8	2.5
17	RoBERTa	97.0	94.6	2.4
18	MPNet	97.0	94.4	2.5
19	DeBERTa-v3	93.6	90.0	3.7
20	ELECTRA	92.6	88.3	4.3

Table 25: arXiv: models trained up to 2024, ordered by future performance.

Rank	Model	Macro AUC (%)	Future (%)	Decay (%)
1	TX-M	96.5	96.3	0.1
2	BiLSTM-M	96.5	96.3	0.2
3	TX-L	96.4	96.3	0.2
4	BiLSTM-Attn-L	96.4	96.3	0.2
5	BiGRU-S	96.4	96.3	0.2
6	TX-S	96.4	96.2	0.2
7	TextCNN-S	96.2	96.0	0.2
8	FFN-L	96.0	95.8	0.2
9	FFN-M	96.0	95.8	0.2
10	TextCNN-M	96.0	95.8	0.3
11	FFN-S	95.9	95.7	0.2
12	MiniLM-L6	95.9	95.6	0.3
13	TextCNN-L	95.8	95.5	0.3
14	ModernBERT	95.6	95.2	0.4
15	BERT	95.1	94.7	0.4
16	DistilBERT	94.9	94.5	0.4
17	RoBERTa	94.9	94.5	0.4
18	MPNet	94.9	94.4	0.4
19	DeBERTa-v3	92.1	91.2	0.8
20	ELECTRA	91.5	90.4	1.1

Table 26: arXiv: future performance and decay by model family.

	2000		2008		2016		2024	
Family	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