Learning to Defer under Expert Drift

Ziyao Mou*
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
zmou1@jh.edu

Andrea Wynn
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
awynn13@jh.edu

Eric Nalisnick
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
nalisnick@jh.edu

Abstract

The *learning to defer* (L2D) framework enables safety in machine learning and AI systems by allocating difficult or critical decisions to a human expert. However, prior work on L2D assumes fixed expert reliability, overlooking temporal fluctuations in human performance due to factors such as fatigue and cognitive biases. Humans commonly exhibit such substantial, systematic performance shifts across a wide range of domains, from high-stakes settings such as radiology, aviation, and judicial decision-making, to everyday tasks such as crowdsourced image annotation. We propose *Expert Drift Adapted Learning-to-Defer* (EDA-L2D), a novel expert-aware L2D framework that explicitly models these temporal shifts by conditioning the deferral head on recent histories of expert outcomes via a sequence model. This history-aware approach enables adaptive deferral policies that anticipate reliability fluctuations and better allocate decisions between human and machine over time. We provide comprehensive experiments across 3 diverse tasks - image classification, medical diagnosis, and hate speech detection - showing that for a time-dependent expert, our approach consistently achieves higher performance and better deferral rates than prior L2D approaches.

1 Introduction

Hybrid intelligent (HI) systems combine human and machine intelligence to achieve goals unattainable by either alone, emphasizing complementarity over replacement [1, 2, 3]. One popular HI paradigm is *learning to defer* (L2D): the system learns not only to predict a label but also to decide when to hand off the decision to a human expert. The original formulation of L2D shows that accounting for the expert's strengths and biases can improve overall accuracy and fairness relative to a static rejection framework [4].

However, existing L2D systems assume that expert behavior is fixed or stationary. In practice, non-stationarity in human decision-making has been documented across a wide range of high-stakes domains. In medical imaging, radiologist diagnostic accuracy declines measurably over overnight shifts due to fatigue [5, 6]. In aviation and autonomous driving, operator performance fluctuates with cognitive load and attentional underload [7, 8, 9]. In judicial settings, Danziger et al. [10] analyzed 1,112 parole rulings by eight judges and found that the probability of a favorable ruling dropped

*Ziyao Mou is currently affiliated with the Department of Computer Science at Tufts University, Medford MA.

Preprint.

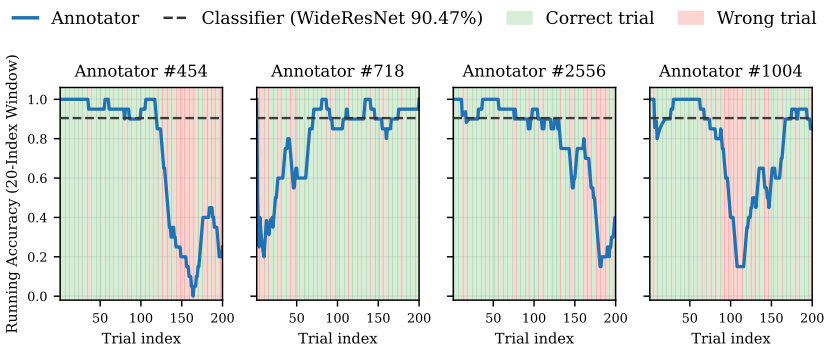


Figure 1: Per-annotator running accuracy (20-index window) over 200 trials, shown for four representative CIFAR-10-H annotators. Accuracy is computed along each annotator's presentation order on non-attention-check trials. Green and red background shading indicate correct and incorrect individual responses, respectively.

from ∼65% at the start of a session to nearly 0% before a break, resetting afterward—a pattern that could not be explained by case characteristics alone. These findings collectively demonstrate that expert decisions exhibit significant within-session non-stationarity, driven by fatigue, cognitive biases [11, 12, 13], and contextual factors.

We show that this phenomenon persists even in ostensibly simple annotation tasks. As shown in Figure 1, four representative annotators from the CIFAR-10-H dataset [14] illustrate the diversity of accuracy trajectories within a single session: some annotators maintain consistently high accuracy throughout, while others exhibit prolonged accuracy drops or abrupt performance shifts. In all four cases, errors tend to cluster temporally suggesting that performance degradation is not random but structured in time. Even on a straightforward visual task, factors such as availability bias and lapses in sustained attention can cause substantial accuracy drift. This motivates L2D systems that can track and adapt to time-varying expert reliability.

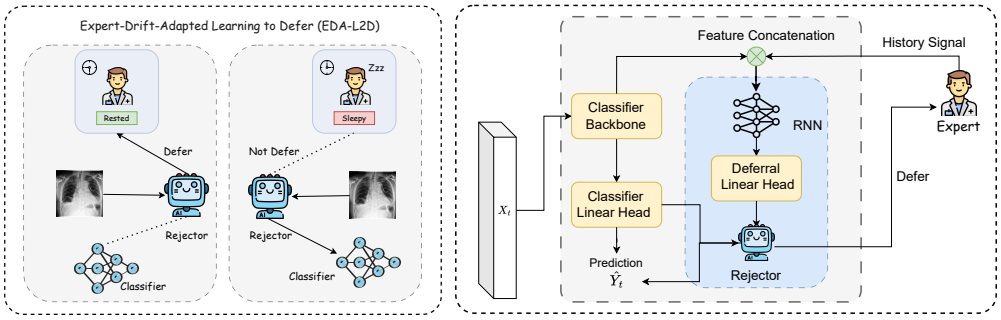
We address this challenge by proposing a novel framework for L2D: *Expert-Drift-Adapted Learning to Defer* (EDA-L2D). Our framework explicitly models non-stationarity in expert performance. We modify L2D's rejector sub-component to condition on short-horizon histories of expert decisions. This allows the system to be aware of the expert's current behavior, and modify its deferral policy accordingly. We illustrate our approach Fig. 2a. Concretely, we integrate a sequence model into the deferral head to to explicitly model expert reliability over time. We validate our approach across diverse tasks including medical image recognition, hate speech detection, and image classification. We demonstrate consistent gains in system accuracy over existing, time-agnostic L2D approaches. In summary, our contributions are as follows:

- We propose a novel *Expert-Drift-Adapted Learning to Defer* (EDA-L2D) that explicitly models variance in human expert performance over time, allowing the system to flexibly adapt to realistic human experts. We propose our approach in §3, including a new time-aware learning objective (§3.2) and model formulation (§3.3) supported by Bayes consistency proofs (Appendix A.2 and A.3).

- We validate the approach with simulated, time-dependent experts on real-world tasks that require HI solutions (§5.1, §5.2, §5.3), and show that this time-variance is present in real human-annotated data.

2 Background

We first describe the traditional L2D setting [15], in which the expert's abilities are assumed to be static and time-invariant.



(a) Learning to Defer with a Fatigued Expert (b) EDA-L2D System Architecture

Figure 2: (a) An example use case of our EDA-L2D framework in a setting with expert performance drift: a radiologist whose performance degrades over a workday. Our model monitors past outcomes to infer current expert reliability in real time, deferring to the expert when reliable and assigning cases to the model when the expert's performance declines. (b) Overview of our expert drift adapted L2D framework. A streaming sequence of inputs is encoded by the backbone into features. At each timestep, the feature vector is concatenated with the expert's previous prediction and passed to an RNN module. The classifier head outputs class logits and the defer head outputs a deferral score; a deferral gate then assigns the instance to either the expert or the classifier. Because the expert's competence drifts over time, this design makes the L2D system time-aware and adaptive to temporal shifts.

2.1 Data & models

We will exclusively consider the problem of multi-class classification with a feature space $\mathcal{X}$ and the label space $\mathcal{Y} = \{1, \ldots, K\}$. Given some input $x \in \mathcal{X}$, we define $\mathbb{P}(\mathrm{y} \mid x)$ as the unknown label-generating distribution and $\mathbb{P}(\mathrm{m} \mid x)$ as the unknown distribution over expert predictions $m \in \mathcal{Y}$. L2D is comprised of two sub-components: a *classifier* $h : \mathcal{X} \to \mathcal{Y}$ and a *rejector* $r : \mathcal{X} \to \{0, 1\}$, which decides whether to make a prediction using the classifier ($r(x) = 0$) or by deferring to the expert ($r(x) = 1$).

2.2 Classifier-rejector loss function

To train our L2D model, we need to fit both the rejector and classifier models. The classifier incurs a loss of zero (correct) or one (incorrect) when it makes a prediction. Similarly, the human expert incurs the same 0-1 loss when making a prediction (i.e. $r(\boldsymbol{x}) = 1$). Combining these two 0-1 losses using the rejector model results in the combined classifier-rejector loss:

$$L_{0-1}(h, r) = \mathbb{E}_{\mathbf{x},\mathrm{y},\mathrm{m}} \left[(1 - r(\mathbf{x})) \, \mathbb{I}[h(\mathbf{x}) \neq \mathrm{y}] \; + \; r(\mathbf{x}) \, \mathbb{I}[\mathrm{m} \neq \mathrm{y}] \right] \tag{1}$$

where $\mathbb{I}$ denotes an indicator function checking the prediction against the ground-truth label. When minimizing this loss, the resulting Bayes optimal classifier and rejector satisfy:

$$h^*(\boldsymbol{x}) = \underset{y \in \mathcal{Y}}{\arg\max} \; \mathbb{P}(\mathrm{y} = y | \boldsymbol{x}), \quad r^*(\boldsymbol{x}) = \mathbb{I}\left[\mathbb{P}(\mathrm{m} = \mathrm{y} | \boldsymbol{x}) \geq \max_{y \in \mathcal{Y}} \mathbb{P}(\mathrm{y} = y | \boldsymbol{x}) \right], \tag{2}$$

where $\mathbb{P}(\mathrm{y}|\boldsymbol{x})$ represents the label probability under the data generating process, and $\mathbb{P}(\mathrm{m} = \mathrm{y}|\boldsymbol{x})$ is the probability of the expert making the correct prediction. The expert may have knowledge not available to the classifier, so they could outperform the Bayes optimal classifier.

2.3 Softmax surrogate loss

A consistent surrogate loss for the above L_{0-1} loss can be derived following the formulation from [15]. First, we consider an augmented label space that includes both the label space $\mathcal{Y}$ and the rejection option $\perp$: $\mathcal{Y}^{\perp} := \mathcal{Y} \cup \{\perp\}$. Secondly, for a class $k \in [1, K]$, let $g_k : \mathcal{X} \mapsto \mathbb{R}$, and let $g_{\perp} : \mathcal{X} \mapsto \mathbb{R}$ denote the rejection option. We can combine these $K + 1$ with a loss resembling the

cross-entropy loss for a softmax parameterization:

$$\begin{aligned}\phi_{\text{SM}}(g_1, \ldots, g_K, g_\perp; \boldsymbol{x}, y, m) = &- \log \left(\frac{\exp\{g_y(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{g_{y'}(\boldsymbol{x})\}} \right) \\ &- \mathbb{I}[m = y] \ \log \left(\frac{\exp\{g_\perp(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{g_{y'}(\boldsymbol{x})\}} \right).\end{aligned} \tag{3}$$

Here, the first term maximizes g_k for the true label k. The second term maximizes the rejection function $g_\perp$, but only when the expert's prediction is correct. At test time, the classifier takes the maximum over the classes: $\hat{y} = h(\boldsymbol{x}) = \arg\max_{k \in [1,K]} g_k(\boldsymbol{x})$. Similarly, we formulate the rejection function as $r(\boldsymbol{x}) = \mathbb{I}[g_\perp(\boldsymbol{x}) \geq \max_k g_k(\boldsymbol{x})]$. The minimizers $g_1^*, \ldots, g_K^*, g_\perp^*$ of ϕ_{SM} also uniquely minimize the 0-1 loss from Equation 1, $L_{0-1}(h, r)$.

3 Expert-drift-adapted Learning to Defer

This section introduces our *Expert-Drift-Adapted Learning to Defer* (EDA-L2D) framework, which extends L2D to account for non-stationarity in expert performance. Accordingly, we derive the corresponding softmax surrogate loss for this setting. We also propose multiple parameterizations, including one that uses a recurrent neural network to incorporate short-horizon temporal context into the deferral policy.

3.1 Setting: non-stationary expert

We consider the same setting as Section 2—L2D for multi-class classification—except now we assume the expert is non-stationary. Thus, instead of a fixed distribution $\mathbb{P}(\text{m} \mid x)$, the expert's predictions are generated by $\mathbb{P}_t(\text{m}_t \mid x)$, with the subscript $t \in \mathcal{T}$ denoting a time index. Note that we are *assuming the classification task itself is stationary*, i.e. $\text{y} \sim \mathbb{P}(\text{y} \mid x)$, which still has no time dependence. To ground the notation in an example, consider a radiology L2D system: forming a diagnosis from a medical image is a static task, yet the paired radiologist changes over time—becoming fatigued over a long shift, with drift that may be non-monotonic, e.g. recovering after a lunch break.

Returning to the technical details, our EDA-L2D system will have a classifier that is defined just as above: $h : \mathcal{X} \to \mathcal{Y}$. Again, the classifier is time-invariant since the prediction task itself is not time dependent. The difference, however, arises in the rejector: $r : \mathcal{X} \times \mathcal{T} \mapsto \{0, 1\}$, meaning the rejector is a function of both the feature space $\mathcal{X}$ and time $\mathcal{T}$. While in the simplest case $\mathcal{T}$ would be a scalar time index, we could also formulate $\mathcal{T}$ as a feature space that describes the current state of the expert (e.g. tabular biomarkers).

3.2 Learning objective and bayes solutions

The learning objective is similar to that in Section 2 (equation 1), except now it takes the form of a summation over time:

$$L_{0-1}^{\mathcal{T}}(h, r) = \sum_{t \in \mathcal{T}} \mathbb{E}_{\mathbf{x}, \text{y}, \text{m}_t} \Big[(1 - r(\mathbf{x}, t)) \, \mathbb{I}[h(\mathbf{x}) \neq \text{y}] + \ r(\mathbf{x}, t) \, \mathbb{I}[\text{m}_t \neq \text{y}] \Big]. \tag{4}$$

As the distribution of y does not depend on time, the classifier's Bayes solution is just as before: $h^*(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(\text{y} = y \mid \boldsymbol{x})$. Yet the Bayes optimal rejector does change, becoming time-dependent like so:

$$r^*(\boldsymbol{x}, t) = \mathbb{I}\left[\mathbb{P}_t(\text{m}_t = \text{y} \mid \boldsymbol{x}) \ \geq \ \max_{y \in \mathcal{Y}} \mathbb{P}(\text{y} = y \mid \boldsymbol{x}) \right].$$

The intuition is the same as above—compare the chance of expert correctness vs classifier confidence for the modal label—yet now the expert correctness term is time dependent, $\mathbb{P}_t(\text{m}_t = \text{y} \mid \boldsymbol{x})$.

3.3 Implementations of the softmax surrogate loss

The derivation for the corresponding softmax surrogate loss follows fairly directly from Mozannar and Sontag [15]'s original derivation. Yet we consider two distinct parameterizations of the underlying

model. The first is a 'per time step' version that treats the L2D problem independently at each time step. We consider this the most naive extension of traditional L2D that still satisfies our motivating setting of expert drift. We then go on to describe how to use a recurrent neural network (RNN) to parameterize the rejector, which allows the expert's previous predictions to help inform the deferral policy. For all implementations, we assume we have access to a training set $\mathcal{D} = \left\{ \{(\boldsymbol{x}_{t,n}, y_{t,n}, m_{t,n})\}_{n=1}^{N} \right\}_{t \in \mathcal{T}}$, which consists of N feature-label-expert-prediction triplets at each of $|\mathcal{T}|$ time steps. Again, we emphasize that only the underlying distribution of $m_{t,n}$ is time-varying, and the time indices on $\boldsymbol{x}_{t,n}, y_{t,n}$ are there merely to 'synchronize' the feature-label pairs with the expert predictions.

Per-time-step implementation and loss The softmax surrogate in equation 3 can be naturally extended to the drifting-expert setting by instantiating a traditional L2D model at each time step. Specifically, we define a model at each time step and train in locally for that time step via the surrogate:

$$\phi_{\text{SM}}^{t}(g_1,...,g_K,g_\perp^t;\boldsymbol{x},y,m_t) = -\log \frac{\exp\{g_y(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{g_{y'}(\boldsymbol{x})\}} - \mathbb{I}[m_t = y] \log \frac{\exp\{g_\perp^t(\boldsymbol{x})\}}{\sum_{y' \in \mathcal{Y}^\perp} \exp\{g_{y'}(\boldsymbol{x})\}}. \tag{5}$$

Notice that $g_\perp^t$ is specific to the current time step, but $g_1, \ldots, g_K$ could be shared across all time steps (since they encode the classifier). As mentioned above, we consider this a naive implementation and will primarily use it to benchmark our second approach, described below. Having $g_\perp^t(\mathbf{x})$ only be a function of $\mathbf{x}$ means that it ignores patterns that could be detected by having information about the expert's behavior at previous time steps. Moreover, this implementation has the clear drawback that it cannot generalize to time steps that were not seen in the training set. For instance, if all training sets had $|\mathcal{T}| = 10$, denoting that the expert worked a 10-hour shift. Then it is not clear how to deploy and use the model when the expert may work for longer than 10 hours.

Loss for RNN-based model We next consider a more sophisticated implementation that uses an RNN-based parameterization to incorporate information about the expert's previous predictions. Here we use a surrogate loss defined across all time steps:

$$\begin{aligned}\phi_{\text{SM}}^{\mathcal{T}}(g_1,...,g_K,g_\perp;\{\boldsymbol{x}_t,y_t,m_t\}_{t\in\mathcal{T}}) = \sum_{t\in\mathcal{T}} &-\log \frac{\exp\{g_{y_t}(\boldsymbol{x}_t)\}}{\exp\{g_\perp(\boldsymbol{x}_T,t)\} + \sum_{y'\in\mathcal{Y}} \exp\{g_{y'}(\boldsymbol{x}_t)\}} \\ &- \mathbb{I}[m_t = y] \log \frac{\exp\{g_\perp(\boldsymbol{x}_t,t)\}}{\exp\{g_\perp(\boldsymbol{x}_T,t)\} + \sum_{y'\in\mathcal{Y}} \exp\{g_{y'}(\boldsymbol{x}_t)\}}.\end{aligned} \tag{6}$$

where here $g_\perp(\boldsymbol{x}_t, t)$ is parameterized with an RNN so that it can leverage information from previous time steps. While we have left the time feature t vague, this will contain the expert's previous prediction m_{t-1} when it is available. We next describe the neural architecture in more detail.

RNN Architecture The RNN first takes as input the feature vector $\mathbf{x}_t$, producing a hidden representation $\boldsymbol{f}_t = \psi_0(\boldsymbol{x}_t)$. For instance, if $\mathbf{x}_t$ is an image, then $\phi(\cdot)$ could be a convolutional NN. The feature representation is then passed into a recurrent module: $\boldsymbol{z}_t = \psi_1(\boldsymbol{z}_{t-1}, \boldsymbol{f}_t, \hat{m}_{t-1})$. During training we use teacher forcing for $\hat{m}_{t-1} \in \{0, 1\}$ (expert correct/incorrect at $t-1$); at test time it is computed from realized routing and the expert's actual outcome. Lastly, the g-functions found in Equation 6 are produced by a final linear transformation: $g_k(\boldsymbol{x}_t) = \boldsymbol{w}_k^\top \boldsymbol{z}_t \quad (k \in 1, \ldots, K])$ and $g_\perp(\mathbf{x}_t) = \boldsymbol{w}_\perp^\top \boldsymbol{z}_t$. The predicted label chosen via $\arg\max_k g_k(\mathbf{x}_t)$. $g_\perp(\boldsymbol{x}_t)$ quantifies the preference for deferral, with the system deferring if $g_\perp(\boldsymbol{x}_t) \geq \max_k g_k(\mathbf{x}_t)$.

The intuition behind this formulation is that the classifier and rejector should not operate solely on the current input but instead incorporate temporal context that reflects the expert's evolving reliability. By conditioning the recurrent state $\boldsymbol{z}_t$ on both the encoded input $\boldsymbol{f}_t$ and the previous correctness indicator $\hat{m}_{t-1}$, the system adaptively tracks fluctuations in expert performance. The classifier head then focuses on predicting the label from this time-dependent state, while the deferral head estimates whether the expert should be trusted at the current step.

While both variants address expert drift, they differ in their temporal expressiveness: the per-step model provides a simple and interpretable instantiation that adapts independently at each time step, whereas the RNN-based model captures dependencies across steps, enabling the system to anticipate trends in expert reliability and generalize to unseen temporal patterns beyond the training horizon.

4 Related work

Learning to Defer (L2D) Early work on classifiers that can choose to reject or abstain instead of making a prediction [16] inspired more recent work on how to model what happens after the classifier abstains [4], with Mozannar and Sontag [15] deriving the first consistent surrogate loss for the problem. This work has been expanded in recent years to address several limitations, including mis-calibration [17, 18], underfitting [19], realizable consistency [20], sample complexity [21], data scarcity [22], and deferral to multiple experts [23]. Two complementary extensions are worth noting. Joshi et al. [24] frame deferral as a sequential decision-making problem via model-based reinforcement learning, addressing tasks where the *prediction target* itself evolves over time—such as managing a patient's diabetes trajectory. Besides, Tailor et al. [25] use meta-learning to adapt to novel experts at test time, addressing population-level variability across different experts. However, these approaches do not consider the complexity of modeling *human* experts—as we do—and could just as well be applied to a black-box API.

Temporal shifts in human performance In many high-stakes domains, human performance shift over time (due to fatigue or other cognitive factors) are well-documented and widely studied. Fatigue is extremely common in high-stakes domains and long-horizon tasks, such as radiology [6], automated driving [7], air traffic control [8], and flying planes [9]. Beyond affecting surface-level performance on these tasks, fatigue physically affects the brain, causing reduced alertness and vigilance that can be measured directly, e.g. using EEG signals [8]. Human behavior can also exhibit cyclical or periodic patterns, such as circadian rhythms [26] or workload-dependent [27], which introduce additional forms of non-stationarity over time.

5 Experiments

We evaluate our proposed EDA-L2D framework through a progression of controlled simulations. We train the two variants of EDA-L2D (per-step and RNN-based) on three benchmarks spanning distinct modalities and annotation regimes: **CIFAR-10** (image classification), **CheXpert** (chest X-ray classification) and **Hate Speech** (text moderation). Our code is publicly available.[2] Our primary baseline is "Naive L2D", meaning that we apply a traditional L2D system that has no awareness of time, collapsing all training data and presenting it to the model without an information about time ordering.

5.1 CIFAR-10

Task and data. We study standard image classification on CIFAR-10 [28]. The dataset contains 60,000 color images from ten object categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck), with 50,000 training images and 10,000 test images. For our experiments, we first arrange training images into sequences and perform a sequence-level split with 90% of the sequences used for training and 10% for validation, ensuring that each sequence appears in exactly one split. The official test set is used for final evaluation.

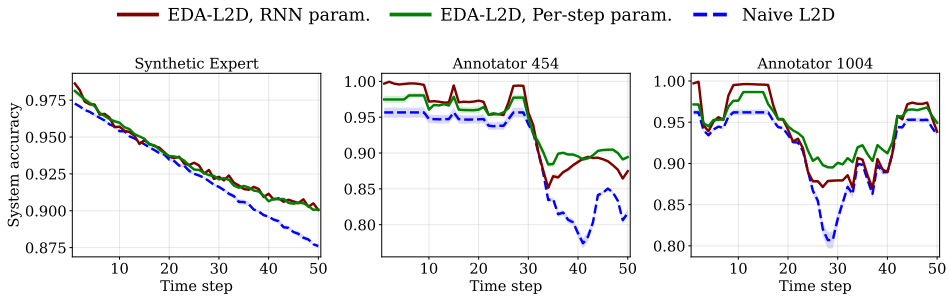
Experts. To evaluate our method under both controlled and realistic conditions, we consider two expert settings.

Synthetic annotator. We use a synthetic annotator whose accuracy linearly decays from 1 to 0.5 over the course of the sequence, simulating a fatiguing annotator.

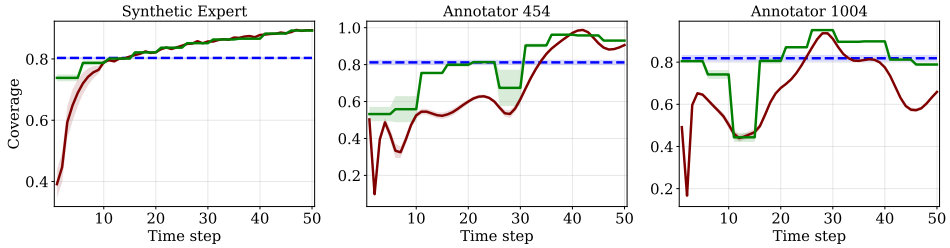
Real human annotator. We select real human annotators from CIFAR-10-H [14] who exhibit large performance drift. For each annotator, we smooth the per-trial accuracy using a rolling window of 20 trials and subsample to $T{=}50$ time steps to obtain the annotator performance curve used in our experiments.

Proposed methods. We instantiate EDA-L2D with a shared WideResNet-$28{\times}4$ backbone. On CIFAR-10, where $K{=}10$, the model outputs 11 logits at each time step: 10 class logits from the classification head and one deferral logit from a recurrent deferral head. At each step, the deferral

[2] `https://github.com/ZiyaoMou/L2D-Expert-Drift`



(a)



(b)

Figure 3: Plots of system accuracy and classifier coverage on CIFAR-10 over T=50 time steps under three annotator performance profiles. The first column corresponds to a synthetic annotator whose accuracy decays linearly from 100% to 50%, while the second and third columns correspond to real annotators from CIFAR-10-H (annotators 454 and 1004). Panels (a) show the system accuracy of the deferral system, and panels (b) show classifier coverage, defined as the proportion of examples handled by the classifier rather than deferred to the annotator. Results are averaged over 10 runs, with error bars denoting the standard error of the mean.

module updates its hidden state from the current backbone features, the expert's outcome at the previous step, and the current position in the sequence.

We consider recurrent deferral modules based on either a single-layer GRU or LSTM, with hyperparameters selected on a held-out validation set. The GRU variant uses hidden dimension 256, while the LSTM variant uses hidden dimension 512 with dropout. In our experiments, validation selection chooses the GRU-based deferral module for the synthetic expert setting and the LSTM-based deferral module for the human-annotator setting. Training details for the Naive L2D baseline, EDA-L2D (per-step), and both recurrent EDA-L2D variants are provided in Appendix B.1.

Results. Fig. 3a: on the synthetic linear expert, RNN and per-step variants beat the naive baseline by 0.59% and 0.86%, respectively, when averaged over all 50 steps. On CIFAR-10-H experts, RNN variants improve over the naive baseline by 3.91% and 1.93% for Annotators 454 and 1004; per-step variants improve by 3.85% and 2.42%, respectively. Fig. 3b shows that the naive baseline maintains nearly constant coverage over time, whereas EDA-L2D reallocates deferral as expert accuracy drifts.

5.2 CheXpert

Task and data. CheXpert[30] is a large chest X-ray dataset with structured labels for 14 pathologies, widely used to benchmark automated radiograph interpretation. We frame the task as *per-task, per-timestep* binary decisions over image sequences of length T: at each timestep $t \in \{1, \ldots, T\}$ and for each observation k, the system must either output a class prediction or defer to an external expert. We use the downsampled CheXpert release with one-vs-rest targets and apply a binary mask to suppress labels marked uncertain or missing. All images are converted to three channels, normalized, and resized to ImageNet-compatible resolution; training augmentation includes random resized crops, horizontal flips, and random rotations up to $15°$.

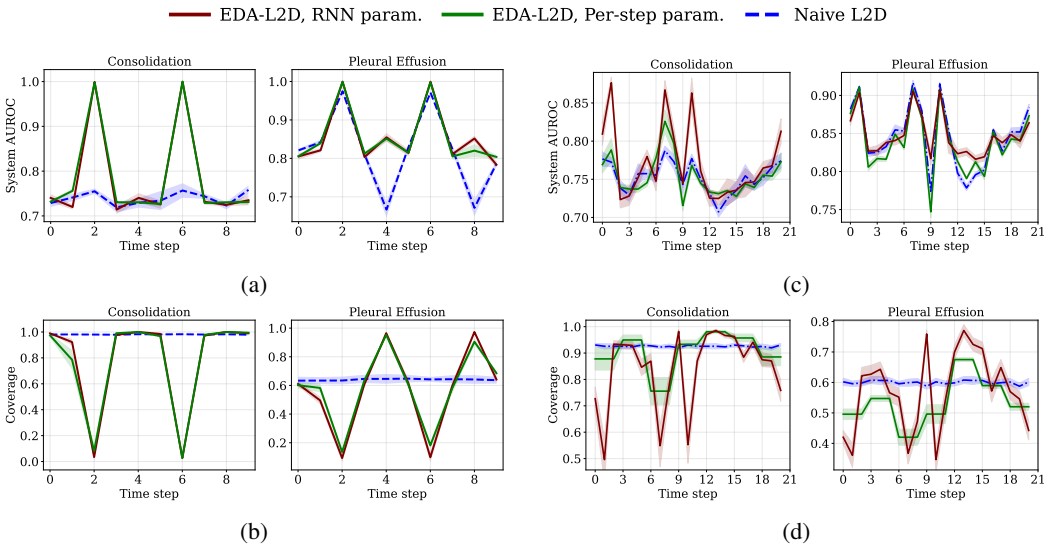


Figure 4: System AUROC (a, c) and classifier coverage (b, d) on CheXpert for Consolidation and Pleural Effusion under the *synthetic expert setting* (a, b) and the *real human expert setting* (c, d); for the latter, we estimate the expert's accuracy trajectory from Figure 3 of Ömer Kasalak et al. [29]. Results are averaged over ten random seeds, with error bars denoting the standard error of the mean.

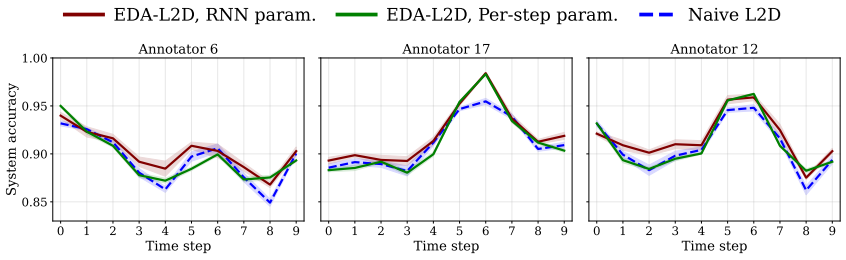
Experts. Studies have shown that radiologist workload varies substantially throughout the day, accompanied by corresponding fluctuations in diagnostic error rates [29]. Motivated by this observation and the reported performance of radiologists on the CheXpert dataset [30], we consider two expert settings.

Synthetic expert. We construct a simple fatiguing expert whose accuracy follows a periodic trajectory over $T{=}10$ discrete time steps, cyclically varying between 50% and 100%. Specifically, we parameterize this trajectory using a mixture of Gaussian basis functions to induce alternating phases of high and low accuracy.

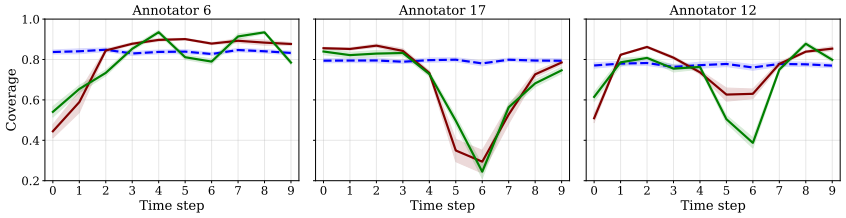
Real human expert. We extract the precision trajectory from Figure 3 of Ömer Kasalak et al. [29], which reports the diagnostic accuracy of the radiologist over a 10.5-hour work period. Assuming that approximately 25 cases are reviewed every 30 minutes, we discretize the curve into $T{=}21$ sequential time steps and linearly rescale the resulting accuracy values to $[0.68, 0.90]$.

Proposed methods. Following Irvin et al. [30], we construct our EDA-L2D(RNN) model on a DenseNet-121 backbone initialised with ImageNet pretraining as a shared feature extractor. At each time step, the extracted features are passed to a per-class classification head producing two logits (negative vs. positive), and to a single-layer LSTM-based deferral module with 1,024 hidden units. The LSTM conditions its deferral decision on the current CNN features and a binary indicator of the expert's correctness at the previous time step, yielding one deferral logit per class. Further architecture and training details are provided in Appendix B.2.

Results. As shown in Fig. 4a, under the synthetic expert setting, both EDA-L2D variants consistently outperform the naive baseline on Consolidation and Pleural Effusion over 10 time steps, with per-step relative AUROC gains of 6.42% and 4.30% and RNN gains of 5.96% and 4.22%, respectively. Fig. 4c shows a similar but weaker trend over 21 time steps: on Consolidation, per-step and LSTM yield relative system-AUROC gains of 0.20% and 2.56%; on Pleural Effusion, LSTM improves by 0.26% while per-step trails the baseline by 0.79% on average. Figs. 4b and 4d mirror the CIFAR-10 pattern: the naive baseline defers at a fixed rate, while both variants dynamically adjust classifier coverage to the periodic expert signal.



(a)



(b)

Figure 5: System accuracy (a) and classifier coverage (b) on the Hate Speech dataset under three real annotator performance curves. Results are averaged over 10 runs with a 60/10/30 train/valid/test split, and error bars indicate the standard error of the mean.

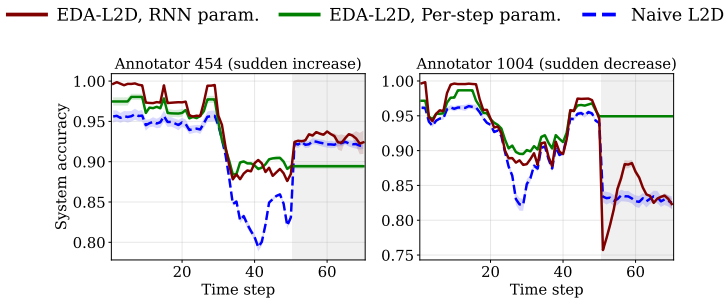
5.3 Hate speech and offensive language detection

Task and data. We study detection of abusive content on Twitter posts using the Davidson et al. [31] corpus of 24,783 English tweets annotated into three mutually exclusive classes: *hate speech*, *offensive but not hate*, and *neither*.

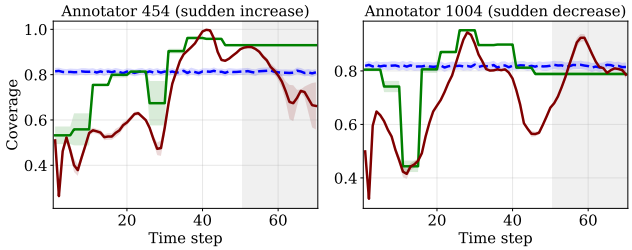
Experts. To our knowledge, no prior work has studied annotator performance drift on the Davidson corpus specifically. The closest reference is Abercrombie et al. [32], who examine intra-annotator stability in hate speech labelling and find that individual annotators' judgements vary substantially over time. We operationalize annotator performance as agreement with the majority vote across all annotators on each item, which we treat as a surrogate gold standard. Using the XHateStability dataset of Abercrombie et al. [32], in which each annotator labels 100 items in a fixed order, we derive a $T{=}10$ expert quality curve by partitioning these annotations into 10 equal bins of 10 items each and computing the running accuracy (15-index window) within each bin. This yields a 10-point curve that captures each annotator's performance trajectory over time.

Proposed methods. We employ a TextCNN encoder to obtain a per-timestep feature representation for each tweet in a length-$T{=}10$ sequence. To model temporal dependencies and expert reliability, we concatenate a binary indicator of the expert's correctness at the previous timestep to the CNN feature and feed the resulting vector into a single-layer LSTM with 256 hidden units. The TextCNN encoder feeds a fully connected classification head, while the LSTM feeds a fully connected deferral head that outputs a single defer logit. We trained models following the recipe in Appendix B.3.

Results. Fig. 5a compares our two EDA-L2D variants to the naive baseline on the hate speech dataset across three annotators. Both variants outperform the baseline for every annotator we inverstigated, with the RNN model strongest overall. Averaged across annotators, accuracy is 90.46% for the naive baseline, 90.63% for per-step conditioning, and 91.30% for the RNN variant. From Fig. 5b we further observe that, while naive L2D maintains a fixed coverage across timesteps, both EDA-L2D variants, particularly the RNN-based model, more effectively leverage the learned temporal structure to adjust coverage at favorable timesteps.



(a)



(b)

Figure 6: System accuracy and classifier coverage on CIFAR-10 over $T = 70$ time steps under two real annotator performance profiles from CIFAR-10-H, evaluated in a step-change scenario. For $t = 0$–49, annotator quality follows the non-monotonic rolling curve seen during training; at $t = 50$ it abruptly shifts (shaded region). The left panel shows a sudden quality *increase* to $p = 0.8$ for Annotator 454, and the right a sudden *decrease* to $p = 0.2$ for Annotator 1004. Bands are ± 1 standard error over 10 runs.

5.4 Generalization evaluation

To further probe the generalization capability of our EDA-L2D variants, we conduct an additional experiment using the same CIFAR-10 setup as in Section 5.1. The annotator accuracy curves for Annotators 454 and 1004 are used both for expert simulation and model training over $T = 50$ time steps. After this in-distribution phase, we introduce a 20-step out-of-distribution period in which the annotator accuracy abruptly shifts to a constant value that contrasts sharply with the final accuracy at $t = 50$: for Annotator 454, quality jumps from $p = 0.25$ to $p = 0.80$; for Annotator 1004, it drops from $p = 0.85$ to $p = 0.20$.

As shown in Fig. 6a and 6b, the naive baseline and EDA-L2D Per-step variant are unable to adjust their behavior in either direction over t = 50. The EDA-L2D RNN variant, by contrast, leverages its sequential conditioning on past annotator outcomes and gradually shifts its coverage toward the new operating point: coverage decreases when quality suddenly improves (left panel) and increases when it degrades (right panel). This adaptation translates into consistently higher system accuracy during $t = 50$–69: for Annotator 454, the RNN variant outperforms Per-step variant by 0.63% and baseline by 0.74%; for Annotator 1004, it outperforms Per-step variant by 1.22% and baseline by 0.53%.

These results suggest that the EDA-L2D RNN variant is more robust to unexpected shifts in annotator quality, adapting its deferral behavior online without any retraining. In real-world deployment, where expert reliability can change unpredictably over time, this flexibility makes the RNN variant a more practical choice than static alternatives.

6 Conclusion, limitation & future work

We propose Expert-Drift-Adapted Learning to Defer (EDA-L2D), a framework that enables improved generalization to real human experts by explicitly modeling temporal drift in their performance, variance that is common and caused by a wide variety of factors such as fatigue and cognitive biases. We achieve this by explicitly incorporating time and prediction history into the L2D model, allowing

the model to learn representations of the expert that vary with each time step and dynamically adapt to the expert's behavior in order to achieve improved deferral decisions and stronger human-AI team performance.

Limitations. Our approach introduces additional data and computational overhead compared to standard L2D, as modeling temporal dynamics requires sequences of expert predictions across time steps. Furthermore, our framework assumes that the expert's drift pattern at test time is consistent with patterns seen during training; in practice, expert behavior may vary in ways that differ substantially from any fixed parametric trajectory, posing challenges for out-of-distribution generalization.

Future Work. Our experiments focus on the single-expert setting, which offers a clean environment for isolating the effect of temporal drift on deferral behavior. The surrogate loss underlying our formulation, however, is not restricted to this case: it naturally extends to the multi-expert setting proposed by Verma et al. [23], and the same modeling principles can accommodate heterogeneous experts with time-varying reliability. Another natural next step is to enrich the model of expert behavior beyond drift alone, capturing cognitive biases such as the availability heuristic [33], gambler's fallacy [34], delay discounting [35], and recency bias [36]; since these biases evolve over time and interact with expert reliability, they are naturally compatible with our temporally structured approach. Pursuing both extensions jointly—multiple experts with richer, evolving bias structure—will require more expressive sequence architectures and introduces additional optimization challenges, which we view as an important and nontrivial direction for future research, enabling a more general treatment of human modeling in L2D systems.

Broader Impact. Beyond these technical extensions, we note that EDA-L2D is intended to improve human-AI collaboration in high-stakes settings such as medical diagnosis and content moderation. Deploying such systems nonetheless requires care: an improperly calibrated deferral policy could over-rely on a fatigued expert, or systematically defer certain types of cases in ways that introduce unfairness. We encourage practitioners to audit deferral behavior across subgroups before deployment.

References

[1] Ece Kamar. Directions in hybrid intelligence: complementing ai systems with human intelligence. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 4070–4073. AAAI Press, 2016. ISBN 9781577357704.

[2] Dominik Dellermann, Philipp Ebel, Matthias Söllner, and Jan Marco Leimeister. Hybrid intelligence. *Business amp; Information Systems Engineering*, 61(5):637–643, March 2019. ISSN 1867-0202. doi: 10.1007/s12599-019-00595-2. URL `http://dx.doi.org/10.1007/s12599-019-00595-2`.

[3] Zeynep Akata, Dan Balliet, Maarten de Rijke, Frank Dignum, Virginia Dignum, Guszti Eiben, Antske Fokkens, Davide Grossi, Koen Hindriks, Holger Hoos, Hayley Hung, Catholijn Jonker, Christof Monz, Mark Neerincx, Frans Oliehoek, Henry Prakken, Stefan Schlobach, Linda van der Gaag, Frank van Harmelen, Herke van Hoof, Birna van Riemsdijk, Aimee van Wynsberghe, Rineke Verbrugge, Bart Verheij, Piek Vossen, and Max Welling. A research agenda for hybrid intelligence: Augmenting human intellect with collaborative, adaptive, responsible, and explainable artificial intelligence. *Computer*, 53(8):18–28, 2020. doi: 10.1109/MC.2020.2996587.

[4] David Madras, Toniann Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer, 2018. URL `https://arxiv.org/abs/1711.06664`.

[5] Tarek N. Hanna, Matthew E. Zygmont, Ryan Peterson, David Theriot, Haris Shekhani, Jamlik-Omari Johnson, and Elizabeth A. Krupinski. The effects of fatigue from overnight shifts on radiology search patterns and diagnostic performance. *Journal of the American College of Radiology*, 2018. doi: 10.1016/j.jacr.2017.12.019.

[6] Silvio G Bruni, Eric Bartlett, and Eugene Yu. Factors involved in discrepant preliminary radiology resident interpretations of neuroradiological imaging studies: a retrospective analysis. *American Journal of Roentgenology*, 198(6):1367–1374, 2012.

[7] Nikol Figalová, Hans-Joachim Bieg, Michael Schulz, Jürgen Pichen, Martin Baumann, Lewis L Chuang, and Olga Pollatos. Fatigue and mental underload further pronounced in l3 conditionally automated driving: Results from an eeg experiment on a test track. In *28th International Conference on Intelligent User Interfaces*, IUI '23, page 64–67. ACM, March 2023. doi: 10.1145/3581754.3584133. URL http://dx.doi.org/10.1145/3581754.3584133.

[8] M. Peukert et al. Subjective and objective fatigue dynamics in air traffic control. *Journal of Sleep Research*, 2023. Showing increase in subjective fatigue and drop in vigilance toward end of evening shifts for air traffic controllers.

[9] Ting Pan, Haibo Wang, Haiqing Si, Haibo Liu, and Mengyue Xu. Research on the identification of pilots' fatigue status based on functional near-infrared spectroscopy. *Aerospace*, 9(3), 2022. ISSN 2226-4310. doi: 10.3390/aerospace9030173. URL https://www.mdpi.com/2226-4310/9/3/173.

[10] Shai Danziger, Jonathan Levav, and Liora Avnaim-Pesso. Extraneous factors in judicial decisions. *Proceedings of the National Academy of Sciences*, 108(17):6889–6892, 2011. doi: 10.1073/pnas.1018033108.

[11] Eric Legler, Darío Cuevas Rivera, Sarah Schwöbel, Ben J. Wagner, and Stefan Kiebel. Cognitive computational model reveals repetition bias in a sequential decision-making task. *Communications Psychology*, 3(1):92, 2025. doi: 10.1038/s44271-025-00271-0.

[12] Anne E. Urai, Jan Willem de Gee, Konstantinos Tsetsos, and Tobias H. Donner. Choice history biases subsequent evidence accumulation. *eLife*, 8:e46331, 2019. doi: 10.7554/eLife.46331.

[13] Didrika S. van de Wouw, Ryan T. McKay, and Nicholas Furl. Biased expectations about future choice options predict sequential economic decisions. *Communications Psychology*, 2(1):119, 2024. doi: 10.1038/s44271-024-00172-8.

[14] Joshua C. Peterson, Ruairidh M. Battleday, Thomas L. Griffiths, and Olga Russakovsky. Human uncertainty makes classification more robust. *CoRR*, abs/1908.07086, 2019. URL http://arxiv.org/abs/1908.07086.

[15] Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert, 2021. URL https://arxiv.org/abs/2006.01862.

[16] C. K. Chow. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, EC-6(4):247–254, 1957. doi: 10.1109/TEC.1957.5222035.

[17] Rajeev Verma and Eric Nalisnick. Calibrated learning to defer with one-vs-all classifiers, 2022. URL https://arxiv.org/abs/2202.03673.

[18] Yuzhou Cao, Hussein Mozannar, Lei Feng, Hongxin Wei, and Bo An. In defense of softmax parametrization for calibrated and consistent learning to defer, 2023. URL https://arxiv.org/abs/2311.01106.

[19] Harikrishna Narasimhan, Wittawat Jitkrittum, Aditya K Menon, Ankit Rawat, and Sanjiv Kumar. Post-hoc estimators for learning to defer to an expert. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 29292–29304. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/bc8f76d9caadd48f77025b1c889d2e2d-Paper-Conference.pdf.

[20] Hussein Mozannar, Hunter Lang, Dennis Wei, Prasanna Sattigeri, Subhro Das, and David Sontag. Who should predict? exact algorithms for learning to defer to humans, 2023. URL https://arxiv.org/abs/2301.06197.

[21] Mohammad-Amin Charusaie, Hussein Mozannar, David Sontag, and Samira Samadi. Sample efficient learning of predictors that complement humans, 2022. URL https://arxiv.org/abs/2207.09584.

[22] Patrick Hemmer, Lukas Thede, Michael Vössing, Johannes Jakubik, and Niklas Kühl. Learning to defer with limited expert predictions, 2023. URL https://arxiv.org/abs/2304.07306.

[23] Rajeev Verma, Daniel Barrejón, and Eric Nalisnick. Learning to Defer to Multiple Experts: Consistent Surrogate Losses, Confidence Calibration, and Conformal Ensembles. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, 2023.

[24] Shalmali Joshi, Sonali Parbhoo, and Finale Doshi-Velez. Learning-to-defer for sequential medical decision-making under uncertainty. *Transactions on Machine Learning Research*, 2023. URL https://arxiv.org/abs/2109.06312.

[25] Dharmesh Tailor, Aditya Patra, Rajeev Verma, Putra Manggala, and Eric Nalisnick. Learning to defer to a population: A meta-learning approach, 2024. URL https://arxiv.org/abs/2403.02683.

[26] Namni Goel, Mathias Basner, Hengyi Rao, and David F. Dinges. Chapter seven - circadian rhythms, sleep deprivation, and human performance. In Martha U. Gillette, editor, *Chronobiology: Biological Timing in Health and Disease*, volume 119 of *Progress in Molecular Biology and Translational Science*, pages 155–190. Academic Press, 2013. doi: https://doi.org/10.1016/B978-0-12-396971-2.00007-5. URL https://www.sciencedirect.com/science/article/pii/B9780123969712000075.

[27] Udo Boehm, Dora Matzke, Matthew Gretton, Spencer Castro, Joel Cooper, Michael Skinner, David Strayer, and Andrew Heathcote. Real-time prediction of short-timescale fluctuations in cognitive workload. *Cognitive Research: Principles and Implications*, 6(1):30, 2021. doi: 10.1186/s41235-021-00289-y. URL https://doi.org/10.1186/s41235-021-00289-y.

[28] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. Technical Report.

[29] Ömer Kasalak, Haider Alnahwi, Romy Toxopeus, Jan P. Pennings, Derya Yakar, and Thomas C. Kwee. Work overload and diagnostic errors in radiology. *European Journal of Radiology*, 167:111032, 2023. ISSN 0720-048X. doi: https://doi.org/10.1016/j.ejrad.2023.111032. URL https://www.sciencedirect.com/science/article/pii/S0720048X23003467.

[30] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison, 2019. URL https://arxiv.org/abs/1901.07031.

[31] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language, 2017. URL https://arxiv.org/abs/1703.04009.

[32] Gavin Abercrombie, Dirk Hovy, and Vinodkumar Prabhakaran. Temporal and second language influence on intra-annotator agreement and stability in hate speech labelling. In Jakob Prange and Annemarie Friedrich, editors, *Proceedings of the 17th Linguistic Annotation Workshop (LAW-XVII)*, pages 96–103, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.law-1.10. URL https://aclanthology.org/2023.law-1.10/.

[33] Amos Tversky and Daniel Kahneman. Availability: A heuristic for judging frequency and probability. *Cognitive Psychology*, 5(2):207–232, 1973. ISSN 0010-0285. doi: https://doi.org/10.1016/0010-0285(73)90033-9. URL https://www.sciencedirect.com/science/article/pii/0010028573900339.

[34] Marko Kovic and Silje Kristiansen. The gambler's fallacy fallacy (fallacy). *Journal of Risk Research*, 22(3):291–302, 2019. doi: 10.1080/13669877.2017.1378248. URL https://doi.org/10.1080/13669877.2017.1378248.

[35] Zeb Kurth-Nelson, Warren Bickel, and A. David Redish. A theoretical account of cognitive effects in delay discounting. *European Journal of Neuroscience*, 35(7):1052–1064, 2012. doi: https://doi.org/10.1111/j.1460-9568.2012.08058.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1460-9568.2012.08058.x.

[36] B.E. Turvey and J.L. Freeman. Jury psychology. In V.S. Ramachandran, editor, *Encyclopedia of Human Behavior (Second Edition)*, pages 495–502. Academic Press, San Diego, second edition edition, 2012. ISBN 978-0-08-096180-4. doi: https://doi.org/10.1016/B978-0-12-375000-6.00216-0. URL https://www.sciencedirect.com/science/article/pii/B9780123750006002160.

A Theoretical results

A.1 Bayes rule for Learning to Defer under expert drift

In this appendix, we provide proofs for the Bayes solutions and consistency of the proposed surrogate losses in the non-stationary expert setting. Throughout, we assume all measurability and regularity conditions hold.

Let $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{M}$ be the input, label, and expert prediction spaces. A classifier $h : \mathcal{X} \to \mathcal{Y}$ and a time-indexed rejector $r : \mathcal{X} \times \mathcal{T} \to \{0, 1\}$ operate alongside a drifting expert whose predictions $m_t \in \mathcal{M}$ follow $\mathbb{P}_t(m_t \mid x, y)$ at each $t \in \mathcal{T}$. With ℓ_{clf} and ℓ_{exp} denoting the respective loss functions and $(x, y, m_t) \sim \mathbb{P}(x, y)\,\mathbb{P}_t(m_t \mid x, y)$, the non-stationary L2D risk is

$$L(h, r) = \sum_{t \in \mathcal{T}} \mathbb{E}_{x,y,m_t}\Big[(1 - r(x, t))\,\ell_{\text{clf}}(h(x), y) + r(x, t)\,\ell_{\text{exp}}(m_t, y)\Big]. \tag{7}$$

Applying the law of total expectation to Equation 7 yields the pointwise decomposition

$$L(h, r) = \sum_{t \in \mathcal{T}} \mathbb{E}_x\big[R_{x,t}(h, r)\big], \tag{8}$$

where the conditional risk at each (x, t) is

$$\begin{aligned} R_{x,t}(h, r) &:= \mathbb{E}_{y,m_t|x,t}\Big[(1 - r(x, t))\,\ell_{\text{clf}}(h(x), y) + r(x, t)\,\ell_{\text{exp}}(m_t, y)\Big] \\ &= (1 - r(x, t))\,\mathbb{E}_{y|x}\big[\ell_{\text{clf}}(h(x), y)\big] + r(x, t)\,\mathbb{E}_{y|x}\mathbb{E}_{m_t|x,y,t}\big[\ell_{\text{exp}}(m_t, y)\big], \end{aligned} \tag{9}$$

where the second equality uses linearity of expectation and the conditional independence of m_t from x given (y, t). Minimizing Equation 9 over $r(x, t) \in \{0, 1\}$ yields the Bayes-optimal rejection rule

$$r^*(x, t) = \begin{cases} 1 & \text{if } \mathbb{E}_{y|x}[\ell_{\text{clf}}(\hat{y}, y)] \;\geq\; \mathbb{E}_{y|x}\mathbb{E}_{m_t|x,y,t}[\ell_{\text{exp}}(m_t, y)]\,, \quad \forall \hat{y} \in \mathcal{Y}, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Under the 0–1 loss for both ℓ_{clf} and ℓ_{exp}, Equation 10 simplifies to

$$r^*(x, t) = \mathbb{I}\left[\mathbb{P}_t(m_t = y \mid x) \;\geq\; \max_{y' \in \mathcal{Y}} \mathbb{P}(y' \mid x)\right], \tag{11}$$

since each agent incurs unit cost on error and the decision reduces to comparing correctness probabilities: the rejector defers at (x, t) if and only if the expert's instantaneous accuracy $\mathbb{P}_t(m_t = y \mid x)$ exceeds the classifier's confidence $\max_{y' \in \mathcal{Y}} \mathbb{P}(y' \mid x)$. This generalizes the static Bayes rule of Mozannar and Sontag [15] to the non-stationary setting: rather than a fixed expert accuracy, the deferral threshold is the time-indexed quantity $\mathbb{P}_t(m_t = y \mid x)$, which tracks expert drift across $t \in \mathcal{T}$.

A.2 Consistency of the softmax surrogate loss for per-step model

We now show that the per–step softmax surrogate defined in Equation 5 is Bayes–consistent with respect to the non–stationary 0–1 loss introduced in Equation 9. For simplicity of notation, we write

$$\zeta_y(x) := -\log \frac{\exp(g_y(x))}{\sum_{y' \in \mathcal{Y}^\perp} \exp(g_{y'}(x))}, \qquad \zeta_{\perp,t}(x) := -\log \frac{\exp(g^t_\perp(x))}{\sum_{y' \in \mathcal{Y}^\perp} \exp(g_{y'}(x))}.$$

Under this notation, the per–step surrogate Equation 5 can be rewritten as

$$\phi^t_{\text{SM}}(g_1, \ldots, g_K, g^t_\perp; x, y, m_t) = \zeta_y(x) + \mathbb{I}[m_t = y]\,\zeta_{\perp,t}(x). \tag{12}$$

Fix any (x, t) and write

$$\eta_y(x) := \mathbb{P}(y \mid x), \qquad \pi_t(x) := \mathbb{P}_t(m_t = y \mid x, y, t).$$

The pointwise conditional risk of the surrogate is

$$\begin{aligned} C[\phi^t_{\text{SM}}] &= \mathbb{E}_{y,m_t|x,t}\big[\phi^t_{\text{SM}}(g; x, y, m_t)\big] \\ &= \sum_{y \in \mathcal{Y}} \eta_y(x)\,\zeta_y(x) + \pi_t(x)\,\zeta_{\perp,t}(x). \end{aligned} \tag{13}$$

Convexity of $C[\phi^t_{\mathrm{SM}}]$ in the logits follows immediately from the convexity of the negative log–softmax. To characterize the minimizer, we differentiate Equation 13 with respect to the logits.

$$\begin{aligned}\frac{\partial C[\phi^t_{\mathrm{SM}}]}{\partial g_y(x)} &= \sum_{y' \in \mathcal{Y}} \eta_{y'} \frac{\partial \zeta_{y'}}{\partial g_y} + \pi_t \frac{\partial \zeta_{\perp,t}}{\partial g_y} \qquad (14)\\ &= \sum_{y' \in \mathcal{Y}} \eta_{y'}(x)\big(p_y(x) - \mathbb{I}[y' = y]\big) + \pi_t(x) p_y(x)\\ &= p_y(x)\Big(\sum_{y' \in \mathcal{Y}} \eta_{y'}(x) + \pi_t(x)\Big) - \eta_y(x)\\ &= (1 + \pi_t(x))\, p_y(x) - \eta_y(x). \qquad (15)\end{aligned}$$

Setting the derivative to zero yields

$$p^*_y(x) \;\propto\; \eta_y(x), \qquad \forall\, y \in \mathcal{Y}. \qquad (16)$$

Similarly, differentiating with respect to the defer logit $g^t_\perp(x)$ gives

$$\begin{aligned}\frac{\partial C[\phi^t_{\mathrm{SM}}]}{\partial g^t_\perp(x)} &= \sum_{y \in \mathcal{Y}} \eta_y(x) \frac{\partial \zeta_y(x)}{\partial g^t_\perp(x)} + \pi_t(x) \frac{\partial \zeta_{\perp,t}(x)}{\partial g^t_\perp(x)}\\ &= \sum_{y \in \mathcal{Y}} \eta_y(x)\, p_{\perp,t}(x) + \pi_t(x)\big(p_{\perp,t}(x) - 1\big)\\ &= p_{\perp,t}(x) \sum_{y \in \mathcal{Y}} \eta_y(x) + \pi_t(x) p_{\perp,t}(x) - \pi_t(x)\\ &= p_{\perp,t}(x) \cdot 1 + \pi_t(x) p_{\perp,t}(x) - \pi_t(x)\\ &= (1 + \pi_t(x))\, p_{\perp,t}(x) - \pi_t(x). \qquad (17)\end{aligned}$$

which implies

$$p^*_{\perp,t}(x) \;\propto\; \pi_t(x). \qquad (18)$$

Equations 16–18 show that the minimizing softmax distribution assigns scores proportional to the true conditional label probabilities $\eta_y(x)$ and the defer probability $\pi_t(x)$ over the augmented label space.

The decoding function associated with the surrogate is the usual $\arg\max$ over the softmax logits. Since the softmax is order–preserving, we obtain

$$\arg\max_{y \in \mathcal{Y}} p^*_y(x) = \arg\max_{y \in \mathcal{Y}} \eta_y(x),$$

and the reject decision satisfies

$$p^*_{\perp,t}(x) \geq \max_{y \in \mathcal{Y}} p^*_y(x) \iff \pi_t(x) \geq \max_{y \in \mathcal{Y}} \eta_y(x).$$

Thus the induced classifier and rejector coincide with the Bayes–optimal rules in Equation 11, and the per–step softmax surrogate is Bayes–consistent for the drifting–expert L2D setting.

A.3 Consistency of the softmax surrogate loss for RNN-based model

Recall the sequence-level surrogate loss

$$\phi^{\mathcal{T}}_{\mathrm{SM}}(g) = \sum_{t \in \mathcal{T}} \phi^t_{\mathrm{SM}}(g; x_t, y_t, m_t). \qquad (19)$$

Its population risk decomposes as

$$\mathcal{R}_{\mathrm{RNN}}(g) := \mathbb{E}_{\mathcal{D}}\big[\phi^{\mathcal{T}}_{\mathrm{SM}}(g)\big] = \sum_{t \in \mathcal{T}} \mathbb{E}_{x_t, y_t, m_t}\big[\phi^t_{\mathrm{SM}}(g; x_t, y_t, m_t)\big]\,. \qquad (20)$$

Define the conditional surrogate risk at each (t, x) as

$$C_t(g; x) := \mathbb{E}_{y, m_t | x, t}\big[\phi^t_{\mathrm{SM}}(g; x, y, m_t)\big]\,, \qquad (21)$$

so that

$$\mathcal{R}_{\mathrm{RNN}}(g) = \sum_{t\in\mathcal{T}} \mathbb{E}_{x_t}[C_t(g; x_t)]\,. \tag{22}$$

Each $C_t(\cdot\,; x)$ is strictly convex in the logits, hence admits a unique minimizer $g^{*,t}(x)$, and

$$C_t(g; x) - C_t(g^{*,t}; x) \geq 0, \quad \forall\, t, x, \tag{23}$$

with equality if and only if $g(x,t) = g^{*,t}(x)$. Writing $g^* = \{g^{*,t}\}_{t\in\mathcal{T}}$ and summing over t,

$$\mathcal{R}_{\mathrm{RNN}}(g) - \mathcal{R}_{\mathrm{RNN}}(g^*) = \sum_{t\in\mathcal{T}} \mathbb{E}_{x_t}\big[C_t(g; x_t) - C_t(g^{*,t}; x_t)\big] \geq 0, \tag{24}$$

by Equation 23 applied termwise. Equality in Equation 24 forces $g(x,t) = g^{*,t}(x)$ almost surely for every t by strict convexity, so g^* is the unique minimizer of $\mathcal{R}_{\mathrm{RNN}}$. Under realizability, the argmax decoder applied to g^* recovers the Bayes-optimal classifier and rejector in Equation 11, establishing Bayes-consistency of the RNN-based softmax surrogate with respect to the non-stationary 0–1 L2D risk.

A.4 Dominance of time-aware deferral under non-stationarity

We further introduce and formalize three deferral mechanisms—static, moving-average, and dynamic deferral—and analyze their theoretical properties. We provide a loss-based comparison of three idealized deferral mechanisms over a horizon of T timesteps.

Setup. Let $\ell_{H^t} \in \mathbb{R}_+$ denote the human's per-timestep loss at time t and let the model incur constant loss ℓ_R. A deferral policy chooses either H or R at each timestep.

Policies. Define the average human loss $\overline{\ell_H} := \frac{1}{T}\sum_{t=1}^{T} \ell_{H^t}$. The best *static* policy (always defer or never defer) achieves

$$\ell_S := \min\{\overline{\ell_H}, \ell_R\}.$$

The *oracle dynamic* policy defers at timestep t iff $\ell_{H^t} < \ell_R$, achieving

$$\ell_D := \frac{1}{T}\sum_{t=1}^{T} \min\{\ell_{H^t}, \ell_R\}.$$

Finally, the *prefix moving-average (MA)* policy forms

$$\hat{\ell}_{H^t} := \frac{1}{t-1}\sum_{j=1}^{t-1} \ell_{H^j}, \quad t \geq 2,$$

and defers at time t iff $\hat{\ell}_{H^t} < \ell_R$, yielding average loss ℓ_{MA}.

A.4.1 Dynamic vs. static: a non-stationary dominance theorem

Theorem 1. *Assume there exist timesteps $t_-, t_+ \in \{1, \ldots, T\}$ such that*

$$\ell_{H^{t_-}} < \ell_R \quad \textit{and} \quad \ell_{H^{t_+}} > \ell_R.$$

Then the oracle dynamic policy strictly improves over the best static policy:

$$\ell_D < \ell_S.$$

Proof. Let $\mathcal{T}_H := \{t : \ell_{H^t} < \ell_R\}$ and $\mathcal{T}_R := \{t : \ell_{H^t} \geq \ell_R\}$. By assumption, both sets are nonempty.

The dynamic policy incurs

$$\ell_D = \frac{1}{T}\left(\sum_{t\in\mathcal{T}_H} \ell_{H^t} + \sum_{t\in\mathcal{T}_R} \ell_R\right).$$

Always defer to H. This static policy incurs

$$\overline{\ell_H} = \frac{1}{T}\left(\sum_{t\in\mathcal{T}_H} \ell_{H^t} + \sum_{t\in\mathcal{T}_R} \ell_{H^t}\right).$$

Subtracting,

$$\overline{\ell_H} - \ell_D = \frac{1}{T}\sum_{t\in\mathcal{T}_R} (\ell_{H^t} - \ell_R) \;\geq\; 0,$$

and since there exists $t_+ \in \mathcal{T}_R$ with $\ell_{H^{t_+}} > \ell_R$, the sum is strictly positive, hence $\ell_D < \overline{\ell_H}$.

Never defer. This static policy incurs ℓ_R at every timestep. Subtracting,

$$\ell_R - \ell_D = \frac{1}{T}\sum_{t\in\mathcal{T}_H} (\ell_R - \ell_{H^t}) \;\geq\; 0,$$

and since there exists $t_- \in \mathcal{T}_H$ with $\ell_{H^{t_-}} < \ell_R$, the sum is strictly positive, hence $\ell_D < \ell_R$.

Therefore $\ell_D < \min\{\overline{\ell_H}, \ell_R\} = \ell_S$.

□

Corollary 1. *Theorem 1 establishes that when there exists at least one timestep at which the human incurs strictly lower loss than the model and at least one timestep at which the human incurs strictly higher loss than the model, the oracle dynamic deferral policy strictly outperforms the best static deferral policy.*

A.4.2 Dynamic vs. moving average under monotone degradation

We now show that even an adaptive but lagged moving-average policy is strictly suboptimal under monotone degradation.

Theorem 2. *Assume $\ell_{H^1} < \ell_{H^2} < \cdots < \ell_{H^T}$ and $\ell_{H^1} < \ell_R < \ell_{H^T}$. Let*

$$j := \min\{t : \ell_{H^t} \geq \ell_R\}$$

be the first timestep where the human reaches the model's loss. Then the prefix MA policy switches strictly after j and incurs strictly larger loss than the oracle dynamic policy:

$$\ell_D < \ell_{MA}.$$

Proof. At time $t = j$, the prefix MA estimate averages only $\ell_{H^1}, \ldots, \ell_{H^{j-1}}$, which are all strictly smaller than ℓ_R by definition of j. Thus $\hat{\ell}_{H^j} < \ell_R$, so MA still defers at j. Let t_{MA} denote the first timestep where MA stops deferring; then $t_{\mathrm{MA}} > j$.

Dynamic defers for $t < j$ and uses R for $t \geq j$, giving

$$\ell_D = \frac{1}{T}\left(\sum_{t=1}^{j-1} \ell_{H^t} + (T - j + 1)\ell_R\right).$$

MA defers at least until $t_{\mathrm{MA}} - 1$, hence

$$\ell_{MA} = \frac{1}{T}\left(\sum_{t=1}^{t_{\mathrm{MA}}-1} \ell_{H^t} + (T - t_{\mathrm{MA}} + 1)\ell_R\right).$$

Subtracting,

$$\ell_{MA} - \ell_D = \frac{1}{T}\sum_{t=j}^{t_{\mathrm{MA}}-1} \big(\ell_{H^t} - \ell_R\big).$$

By monotonicity and the definition of j, we have $\ell_{H^t} \geq \ell_R$ for all $t \geq j$, and the sum is over a nonempty index set since $t_{\mathrm{MA}} > j$. Moreover, strict crossing implies $\ell_{H^t} > \ell_R$ for some $t \geq j$, hence the sum is strictly positive and $\ell_{MA} > \ell_D$. □

A.4.3 Dynamic vs. moving average under periodic drift

Theorem 3. *Let $\{\ell_{H^t}\}_{t=1}^{T}$ be periodic with period P and assume $T = mP$. Let ℓ_R be constant. Define the oracle dynamic policy*

$$a_D(t) = \begin{cases} H & \text{if } \ell_{H^t} < \ell_R, \\ R & \text{otherwise}, \end{cases} \qquad \ell_D = \frac{1}{T}\sum_{t=1}^{T} \min\{\ell_{H^t}, \ell_R\}.$$

Let a moving-average policy choose H at time t if $\hat{\ell}_{H^t} < \ell_R$, where

$$\hat{\ell}_{H^t} = \sum_{k=1}^{K} w_k \, \ell_{H^{t-k}}, \quad w_k \geq 0, \; \sum_{k=1}^{K} w_k = 1,$$

covers prefix MA (with $K = t-1$ and $w_k = 1/(t-1)$) and fixed-window MA (fixed K and uniform weights).

Assume there exists a timestep $t^\star$ such that

$$\ell_{H^{t^\star}} > \ell_R \quad \textit{but} \quad \hat{\ell}_{H^{t^\star}} < \ell_R,$$

or symmetrically

$$\ell_{H^{t^\star}} < \ell_R \quad \textit{but} \quad \hat{\ell}_{H^{t^\star}} \geq \ell_R.$$

Then the moving-average policy incurs strictly larger average loss than the oracle dynamic policy:

$$\ell_{MA} > \ell_D.$$

Proof. Let $a_{MA}(t) \in \{H, R\}$ denote the agent chosen by the moving-average policy at time t. Define the per-timestep loss under policy $\pi \in \{D, MA\}$ as

$$L_\pi(t) = \begin{cases} \ell_{H^t} & \text{if } a_\pi(t) = H, \\ \ell_R & \text{if } a_\pi(t) = R. \end{cases}$$

By definition of the oracle policy, for every t we have

$$L_D(t) = \min\{\ell_{H^t}, \ell_R\} \leq L_{MA}(t).$$

Indeed, if $\ell_{H^t} < \ell_R$ then $a_D(t) = H$ and $L_D(t) = \ell_{H^t} \leq L_{MA}(t)$; if $\ell_{H^t} \geq \ell_R$ then $a_D(t) = R$ and $L_D(t) = \ell_R \leq L_{MA}(t)$.

Under the stated assumption, there exists $t^\star$ where $a_{MA}(t^\star) \neq a_D(t^\star)$. In the first case, $\ell_{H^{t^\star}} > \ell_R$ but $\hat{\ell}_{H^{t^\star}} < \ell_R$, so $a_{MA}(t^\star) = H$ while $a_D(t^\star) = R$ and thus

$$L_{MA}(t^\star) - L_D(t^\star) = \ell_{H^{t^\star}} - \ell_R > 0.$$

In the second case, $\ell_{H^{t^\star}} < \ell_R$ but $\hat{\ell}_{H^{t^\star}} \geq \ell_R$, so $a_{MA}(t^\star) = R$ while $a_D(t^\star) = H$ and thus

$$L_{MA}(t^\star) - L_D(t^\star) = \ell_R - \ell_{H^{t^\star}} > 0.$$

Therefore,

$$\sum_{t=1}^{T} \big(L_{MA}(t) - L_D(t)\big) > 0,$$

which implies

$$\ell_{MA} - \ell_D = \frac{1}{T}\sum_{t=1}^{T} \big(L_{MA}(t) - L_D(t)\big) > 0.$$

Hence $\ell_{MA} > \ell_D$. □

Corollary 2. *Taken together, Theorems 1, 2, and 3 establish that time-aware dynamic deferral policies are strictly more expressive than both static deferral policies and lagged moving-average policies.*

A.4.4 Mapping simulated deferral mechanisms to EDA-L2D and baselines

Motivated by this hierarchy, we instantiate a corresponding family of learnable L2D variants with increasing temporal expressiveness. Specifically, we map static deferral to a naive L2D baseline that ignores temporal variation, moving-average deferral to a per-step EDA-L2D model that makes independent decisions based on recent history, and fully dynamic deferral to an RNN-based EDA-L2D model that explicitly captures temporal dependencies in human performance.

B Experimental setup

All experiments are conducted using standard commodity hardware. Since our work focuses on algorithmic design rather than large-scale model training, the computational requirements are modest. All reported results can be reproduced on a single GPU or CPU workstation, without specialized accelerators or large-scale distributed infrastructure.

B.1 Training configuration for CIFAR-10 experiments

All methods use sequences of length $T{=}50$; only the expert accuracy curve differs between the *synthetic expert setting* and the *human annotators setting*. When the expert errs, we assign a uniformly sampled incorrect label.

Synthetic expert setting. Expert accuracy follows a linear decay $p_t = 1 - 0.01\,t$ from 1.0 at $t{=}0$ to 0.5 at $t{=}50$.

Human annotators setting. We estimate each annotator's trajectory from CIFAR-10-H. We restrict to non-attention-check trials, compute running accuracy with window size 20, and compress the resulting series into 50 equal-width temporal bins to obtain a 50-point accuracy curve (e.g., Annotators 454 and 1004).

Naive L2D. We use a WideResNet-$28{\times}4$ with $K{+}1{=}11$ logits, trained end-to-end with the softmax surrogate loss (Table 1).

EDA-L2D, per-step. We train one independent WideResNet-$28{\times}4$ per time window (10 windows $\times$ 5 steps), with the same recipe as Naive L2D (Table 1).

EDA-L2D, RNN (synthetic expert setting). We adopt a two-stage procedure: the backbone is pretrained for classification, then a single-layer GRU deferral module (hidden dimension 256, dropout 0.1) is attached and trained jointly over full sequences. At each step, the GRU receives the backbone feature, a binary indicator of expert correctness at the previous step, and a normalised time index $\tilde{t} \in [0, 1]$; a linear deferral head outputs a scalar logit concatenated with the class logits. We retain the checkpoint with the highest validation system accuracy (Table 2).

EDA-L2D, RNN (human annotators setting). We follow the same two-stage recipe, but use a single-layer LSTM deferral module (hidden dimension 512, dropout 0.5) in place of the GRU. The pretrained backbone is fine-tuned with a smaller learning rate, while the LSTM and heads are optimised with Adam; the best checkpoint is selected by validation loss with early stopping (patience 150; Table 2).

B.2 Training configuration for CheXpert experiments

All three methods—Naive L2D, EDA-L2D (per-step), and EDA-L2D (RNN)—follow the same training recipe (Table 3); the only differences across the two experimental settings are the sequence length and the expert accuracy curve, as described in Section 5.2.

Synthetic expert setting ($T{=}10$). We use a sequence length of $T{=}10$ and simulate expert accuracy with a non-monotonic curve generated from a mixture of Gaussians, yielding the sequence $(0.75,\ 0.80,\ 1.00,\ 0.75,\ 0.50,\ 0.75,\ 1.00,\ 0.75,\ 0.50,\ 0.70)$.

Table 1: Training hyperparameters shared across both expert settings (Naive L2D and EDA-L2D, per-step).

Hyperparameter	Naive L2D / EDA-L2D, per-step
Optimizer	SGD
Learning rate	0.1
Momentum	0.9
Nesterov	✓
Weight decay	5×10^{-4}
LR schedule	Cosine (per iteration)
Batch size	128
Epochs	200
Sequence length T	50
Per-step windows	10×5 steps

Table 2: Training hyperparameters for EDA-L2D, RNN.

Hyperparameter	Synthetic annotator setting	Human annotator setting
Recurrent unit	GRU	LSTM
Hidden dimension	256	512
Recurrent dropout	0.1	0.5
Backbone pretraining	200 epochs (classification)	
Backbone optimizer	SGD	SGD
Backbone learning rate	2×10^{-2}	5×10^{-6} (annotator 454) 1×10^{-6} (annotator 1004)
Recurrent / head optimizer	Adam	Adam
Recurrent / head learning rate	10^{-3}	10^{-3}
Weight decay	5×10^{-4}	5×10^{-4}
LR schedule (backbone)	Cosine (per iteration)	
LR schedule (recurrent/heads)	Cosine, then hold	
Batch size	500	500
Epochs	200	150

Real human expert setting (T=21). We use a sequence length of T=21 and estimate the expert's accuracy trajectory from Figure 3 of Ömer Kasalak et al. [29]. For each time step, we read off the per-step error count (n=25), compute the raw accuracy as $1 - \text{errors}/25$, and linearly rescale the resulting 21-point curve to the range $[0.68, 0.90]$, yielding the sequence $(0.86, 0.90, 0.77, 0.77, 0.77, 0.81, 0.81, 0.90, 0.86, 0.68, 0.90, 0.81, 0.72, 0.68, 0.72, 0.72, 0.81, 0.77, 0.81, 0.81, 0.86)$.

Naive L2D. We initialise a DenseNet-121 backbone with ImageNet pretraining and first train it for 3 epochs without deferral, using a shared cross-entropy loss over the 14×3 output head (negative, positive, and defer per pathology). At Stage 2, we then fine-tune the pretrained backbone for 1 epoch with the full softmax surrogate deferral loss at learning rate 1×10^{-4}.

EDA-L2D, per-step. we first pretrain a single shared DenseNet-121 backbone without deferral. At stage 2, we then fine-tune one independent model per time step from this shared backbone, with the expert accuracy fixed at the value corresponding to that step.

EDA-L2D, RNN. We pretrain a DenseNet-121 backbone for 3 epochs without deferral and share the resulting checkpoint across all seeds. We then attach a single-layer LSTM with hidden dimension $1{,}024$ to the backbone features. At each time step, the LSTM receives a concatenation of the CNN feature and a 14-dimensional binary vector encoding per-pathology expert correctness at the previous step (zero-initialised at t=0). A linear deferral head maps the LSTM hidden state to 14 deferral logits. This deferral module is trained for 4 epochs at learning rate 1×10^{-3}.

Table 3: Training hyperparameters for CheXpert experiments.

Hyperparameter	Naive L2D	EDA-L2D, per-step	EDA-L2D, RNN
Optimizer	Adam	Adam	Adam
Learning rate	1×10^{-4}	1×10^{-4}	1×10^{-3}
β_1, β_2	0.9, 0.999	0.9, 0.999	0.9, 0.999
Weight decay	1×10^{-5}	1×10^{-5}	1×10^{-5}
LR schedule	ReduceLROnPlateau	ReduceLROnPlateau	ReduceLROnPlateau
Plateau factor	0.1	0.1	0.1
Plateau patience	2	2	2
Sequence length T	10	10	10
LSTM hidden dim	—	—	1,024
LSTM layers	—	—	1

Table 4: Training hyperparameters for hate speech experiments.

Hyperparameter	Naive L2D	EDA-L2D, per-step	EDA-L2D, RNN
Optimizer	Adam	Adam	Adam
Learning rate	1×10^{-3}	1×10^{-3}	1×10^{-3}
Batch size	64	64	64
Epochs	5	5	20
Early stopping	✓	✓	✓
Embedding dim	100	100	100
CNN filters	300	300	300
CNN kernel sizes	3, 4, 5	3, 4, 5	3, 4, 5
Dropout	0.5	0.5	0.5
LSTM hidden dim	—	—	256
LSTM layers	—	—	1
Sequence length T	10	10	10

B.3 Training configuration for Hate Speech experiments

All three methods—Naive L2D, EDA-L2D (per-step), and EDA-L2D (RNN)—follow the same training recipe (Table 3). Expert accuracy curves are constructed as described in Section 5.3. All experiments use sequence length T=10.

Naive L2D. We use a TextCNN encoder with three parallel convolutional filters of kernel sizes 3, 4, and 5 (300 channels each), applied over 100-dimensional GloVe embeddings. The pooled feature vector is passed to a 3-way classification head and a scalar deferral head, whose outputs are concatenated and normalised via softmax. A single model is shared across all time steps.

EDA-L2D, per-step. We train one independent TextCNN model per time step using the same architecture as the Naive baseline. At each step t, the expert accuracy is fixed at the value corresponding to that step.

EDA-L2D, RNN. We use a CNN-LSTM architecture. At each time step, a shared TextCNN encoder produces a 900-dimensional feature vector. These are fed sequentially into a single-layer LSTM with hidden dimension 256. A classification head maps the per-step CNN features to 3-class logits, while a deferral head maps the LSTM hidden state to a scalar defer logit, concatenated to form a 4-dimensional softmax output per step. Training uses early stopping based on validation system accuracy.

C Additional Experimental Results

C.1 Synthetic Data

We construct a controlled sequential Gaussian–mixture environment to induce time-varying expert accuracy across two sub-populations $A \in \{0, 1\}$. The covariates lie in $\mathcal{X} = \mathbb{R}^d$ with binary targets

Table 5: Comparison of L2D variants on synthetic data. Results are averaged over timesteps; we report mean ± standard error for system accuracy and coverage.

Method	System accuracy (%)	Coverage (%)	Gap to best (%)
Naive L2D	71.58 ± 0.43	53.69 ± 0.40	-1.38
EDA-L2D, Per-step	71.79 ± 0.27	54.21 ± 0.25	-1.17
EDA-L2D, RNN	$\mathbf{72.96 \pm 0.36}$	$\mathbf{53.71 \pm 0.77}$	$\mathbf{0.00}$

$Y \in \{0, 1\}$. For each group a and class y, the class-conditional distribution is $X \mid (Y{=}y, A{=}a) \sim \mathcal{N}(\mu_{y,a}, \Sigma_{y,a})$ with $\mu_{y,a} \in [0, 10]^d$ and diagonal $\Sigma_{y,a}$ whose entries are drawn from $\mathrm{Unif}(0, 10)$. Sequences of length T evolve by adding Gaussian noise and a group-specific drift, and labels may flip with small probability to model non-stationarity.

Setup Across 200 trials we sample the group proportion $P(A{=}1) \sim \mathrm{Unif}(0.02, 0.98)$, fix $d{=}10$ and $T{=}10$, and generate $N{=}6000$ sequences per trial. Within each sequence, features evolve via $X_t = X_{t-1} + \varepsilon_t + \Delta_a$, with $\varepsilon_t \sim \mathcal{N}(0, \sigma^2 I)$ and group drift Δ_a (smaller for $A{=}0$, larger for $A{=}1$). Labels follow $Y_t = Y_{t-1}$ with probability $1 - p_{\text{flip}}$ and flip otherwise. Expert predictions E_t are correct with a time-decayed accuracy $p_a(t) = p_{0,a} \exp(-\lambda_a t/T)$, yielding a slower decay for $A{=}0$ and a faster decay for $A{=}1$. We train each model for five epochs using Adam with a learning rate of 10^{-3} and weight decay 10^{-5}. The data are partitioned into training, validation, and test sets containing 70%, 15%, and 15% of the samples, respectively, and we report test metrics aggregated across trials.

Models and baselines. We compare three heads that map each time step's feature vector to three logits over $\{0, 1, \perp\}$. (i) *Naive L2D*: a single time-invariant linear head shared across all steps, providing strong parameter sharing and a stationary inductive bias. (ii) *EDA-L2D, Per-step param.*: T independent linear heads for per step, capturing nonstationarity at the cost of T-fold parameters. (iii) *EDA-L2D, RNN param.*: a factorized design with a time-local classifier branch and a sequence-aware deferral branch that feeds $[x_t, e_{t-1}]$ into a GRU followed by a linear head for the defer logit; the three logits are concatenated to form the final output at each step.

Results. Table 5 reports the comparative performance of the three L2D variants in the synthetic sequential environment. The time-agnostic Naive L2D serves as a stationary baseline and attains the lowest system accuracy. Per-step EDA-L2D yields a modest gain by capturing coarse nonstationarity, albeit without temporal smoothing. The RNN-based EDA-L2D achieves the best accuracy ($+1.38\%$ over Naive L2D), confirming the benefit of inserting a sequence-aware module to drive the deferral head in L2D.

C.2 CheXpert

In addition to the main periodic-expert results reported in the paper, we conduct an auxiliary experiment on CheXpert to examine model behavior under a monotonic degradation pattern in expert reliability.

Setup. We follow the same data and task configuration described in Sec. 5.2, but replace the periodic expert with a simplified linearly degrading expert. Specifically, at timestep t, the expert's correctness probability decays linearly from 100% at $t = 1$ to 70% at the final step. Full model architecture and training details follow those described in Appendix B.2.

Results. As shown in Fig. 7a, both EDA-L2D variants consistently outperform the naive baseline under monotonic expert degradation. EDA-L2D(RNN) achieves gains of 1.05%, 3.50%, and 1.66% on Cardiomegaly, Consolidation, and Pleural Effusion respectively, while EDA-L2D(Per-step) yields gains of 1.34%, 2.70%, and 1.68%. Fig. 7b confirms the same adaptive coverage pattern observed in the periodic setting: the naive baseline maintains a fixed deferral rate throughout, whereas both EDA-L2D variants progressively increase classifier reliance as the expert degrades over time.

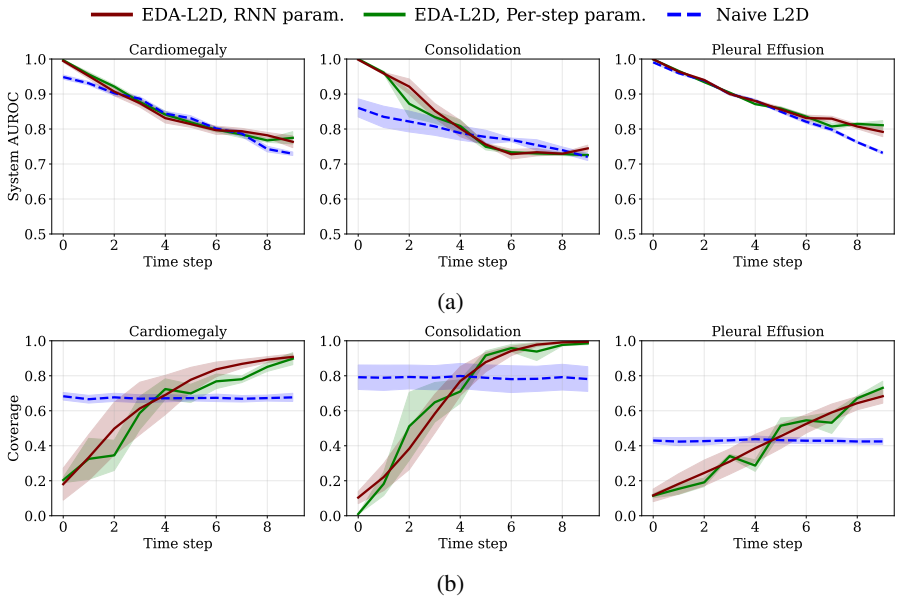


Figure 7: Plots of system AUROC (a) and coverage (b), where coverage denotes the ratio of classifier-taken examples, comparing our methods with the baseline on the CheXpert dataset. At each timestep t, we evaluate using an expert with linearly decreasing from 100% to 70%. Results are averaged over 10 runs, and error bars indicate the standard error of the mean.

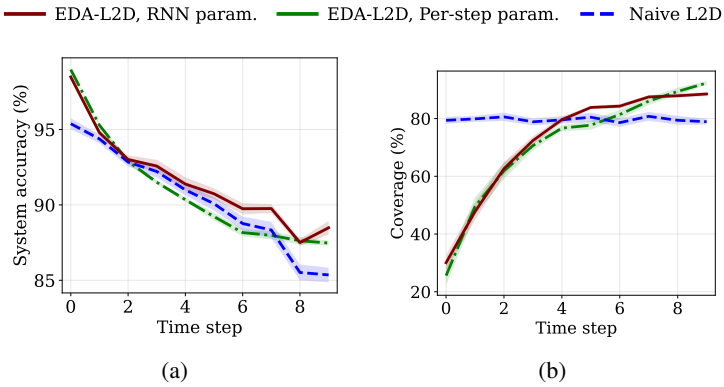


Figure 8: System accuracy (a) and classifier coverage (b) on the hate speech dataset under a synthetic expert whose accuracy decays linearly from 100% to 55% over 10 timesteps. Results are averaged over 10 runs, and error bars indicate the standard error of the mean.

C.3 Hate speech and offensive language detection

In addition to the real annotator results reported in the main paper, we conduct an auxiliary experiment under a synthetic linearly degrading expert to examine model behavior under a controlled monotonic drift pattern.

Setup. We follow the same data and task configuration described in Sec. 5.3, but replace the real annotator performance curves with a synthetic expert whose correctness probability decays linearly from 100% at $t = 1$ to 55% at the final step. Full model architecture and training details follow those described in Appendix B.3.

Results. As shown in Fig. 8, both EDA-L2D variants outperform the naive baseline throughout the sequence, with gains becoming more pronounced as the expert degrades. At $t = 9$, EDA-L2D(RNN) and EDA-L2D(Per-step) achieve 88.48% and 87.47% system accuracy respectively, compared to

85.36% for the naive baseline. Fig. 8b shows that both variants dynamically adjust classifier coverage in response to expert drift, while the naive baseline maintains a fixed deferral rate throughout.