THE UNACTIONABILITY OF CONCEPT DRIFT DETECTION.

Anonymous authors
Paper under double-blind review

ABSTRACT

Change detectors are commonly used alongside fixed heuristic rules as adaptation mechanisms in online learning algorithms deployed in non-stationary environments. This work investigates whether a change detector signal contains sufficient information to determine an appropriate model adaptation strategy. We show theoretically that a detector can be unactionable: distinct post-drift distributions may produce the same detector signal while having different optimal adaptation strategies. We further derive an information-theoretic bound showing that the information available to an adaptation policy is limited by the information exposed by the detector, and that this can lead to unavoidable misadaptation when the entropy of the optimal adaptation decision exceeds detector capacity. We evaluate these results on controlled synthetic streams and study their practical implications across 24 streams from the USP DS repository. Empirical results show substantial variation in the utility of fixed detector-triggered adaptation across stream-learning architectures and datasets, motivating adaptation policies that utilise more information than a fixed detector signal alone.

1 INTRODUCTION

Many Online Stream Learning algorithms operate in settings where samples arrive frequently and in large quantities requiring these algorithms to be performant in both inference speed and memory usage. Furthermore, these stream learners are required to adapt to the non-stationary dynamics of the data stream itself (i.e., concept drift or data shift) (Gama et al., 2014). In academic contexts, stream learning algorithms often use change / drift detectors in combination with heuristic (static) adaptation policies. For example, many stream learning ensembles use change detectors to determine when to replace base learners (Bifet et al., 2009; 2010; Gomes et al., 2017; 2019).

Conversely, domain knowledge and careful parameter tuning are often crucial for utilising change detectors and adaptation policies effectively in practical setttings (Silva et al., 2024). Research has also shown that the perception of drift can be dependent on the range of data given to a change detector (Gower-Winter et al., 2026), the types of drift they are designed to detect, and how the detectors are parameter-tuned (Bayram et al., 2022; Cerqueira et al., 2026). Furthermore, the perception of concept drift is often stream learner specific (Hinder et al., 2023) and it has been shown that certain types of detectors are only reliable as adaptive mechanisms if the change they detect in some stream correlates with a performance improvement (Kuncheva & Žliobaitė, 2009).

This work asks whether change detection is sufficient for determining model adaptation strategies in non-stationary environments. We show that, in general, a detector signal need not contain sufficient information to select the appropriate adaptation operation.

1. **Unactionability of Change Detection:** We establish conditions under which a detector signal cannot uniquely determine the appropriate adaptation operation without incurring excess risk. In particular, we construct post-drift distributions that produce identical detector observations but favour different adaptation operations.
2. **Information Bottleneck Bounds:** We show that the information capacity of a change detector (usually 1-bit) acts as an upper bound on optimal adaptation selection. When then show that as the entropy of the optimal adaptation decision grows relative to detector capacity, this limitation induces a non-zero lower bound on the probability of misadaptation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

3. **Empirical & Benchmark Validation:** We explore our theoretical findings on synthetic streams and evaluate four model classes (Decision Trees, kNNs, Ensembles and Neural Networks) across 24 streams in the USP DS repository, for which we show substantial variation in the utility of fixed (detector driven) adaptations.

2 Background and Related Work

A data stream is a sequence of potentially infinite instances arriving sequentially over time (Read & Žliobaitė, 2025):

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t), \ldots\} \sim P_t(X, Y). \tag{1}$$

Unlike traditional offline settings, data streams are non-i.i.d. because the underlying joint distribution $P_t(X, Y)$ may evolve over time. This phenomenon, known as *concept drift*, occurs when:

$$P_t(X, Y) \neq P_{t+1}(X, Y), \tag{2}$$

which can manifest as covariate drift ($P(X)$), label drift ($P(Y)$), posterior drift ($P(Y|X)$) or representation drift whereby a model's parameters are no longer optimal on the post drift distribution.

Change / Drift Detectors are a class of methods used to identify drift events by evaluating windows of data and outputting a binary indicator $Z \in \{0, 1\}$ signalling whether drift has occurred (Hinder et al., 2024). Unsupervised detectors such as D3 (Gözüaçık et al., 2019), IBDD (Souza et al., 2021), and MCD (Wan et al., 2024) rely solely on feature values X, limiting their scope to detecting covariate shift $P(X)$ rather than posterior shift $P(Y|X)$. Supervised detectors such as DDM (Gama et al., 2004) and PUDD (Lu et al., 2025) monitor loss, error rates or prediction uncertainties, requiring instance labels Y and predictions $\widehat{Y}$ tying detection directly to the performance of the predictor they accompany (Hinder et al., 2024). Semi-supervised detectors, such as STUDD (Cerqueira et al., 2023), typically only require data from the predictor they accompany. Hybrid detectors such as ADWIN (Bifet & Gavalda, 2007), and Kolmogorov-Smirnov (Raab et al., 2020) may operate across all three paradigms depending on the input signal provided.

In many of the state-of-the-art learning stream learning algorithms, change detectors are used as adaptive mechanisms. The most common use of a change detector is a reset mechanism whereby upon, detection, some aspect of the stream learner is reset. For example, the Hoeffding Adaptive Tree (HAT) places detectors on each node in the tree and resets those nodes when drift is detected (Bifet & Gavalda, 2009). In many ensemble strategies, detectors are used to identify when underperforming base models should be replaced (Bifet et al., 2009; 2010; Gomes et al., 2017; 2019). LAST (a decision tree architecture) uses change detectors to identify when a split decision should be made (Assis et al., 2024). Neural Networks architectures apply a broader range of adaptive strategies. For example, CAND (Gunasekara et al., 2022) uses ADWIN to estimate the loss of base networks in a neural ensemble. ATNN uses DDM to create new base learners (Wen et al., 2024). PUDD (Lu et al., 2025) was used to reset the classification head of a network, and CAMEL (Yu et al., 2026) uses MCD to add and prune networks in a mixture-of-experts like architecture.

Change detectors are used as active adaptation mechanisms whereby their output signal determines when adaptation occurs. Conversely, passive adaptation strategies, such as sliding windows (Buschjäger et al., 2022) or experience replay buffers (Hayes et al., 2019) induce adaptation implicitly. Passive adaptation is commonly used in online kNN algorithms (Read et al., 2012; Losing et al., 2016; Bahri, 2022) and ensemble strategies (Kolter & Maloof, 2007). This work focusses exclusively on active adaptation strategies. However, we note that many passively adapting algorithms also use fixed active adaptation strategies in the form of heuristic rules. Despite rarely being identified as such, heuristic rules are equivalent to change detectors when used as adaptive mechanisms.

Change detectors have been extensively studied. Hinder et al. (2024) showed that the perception of drift in supervised detectors is often based on the supervised learning algorithm they are paired with. Supervised detectors have also been shown to frequently produce false alarms (Bayram et al., 2022). Cerqueira et al. (2026) showed that drift detectors are often effective at detecting either abrupt or gradual drift, but rarely both. Souza et al. (2020) noted that concept drift is often impractical or impossible to verify in real-world settings. Lukats et al. (2025) remarked that unsupervised detectors that fire frequently tend to perform best when measuring stream learner performance, this finding is corroborated in the results of Souza et al. (2020) who investigated supervised detectors. Furthermore, Bifet (2017) showed simple resetting strategies can outperform change detectors as adaptive

mechanisms, and Gower-Winter et al. 2026 showed that even models without change detectors can outperform models equipped with change detectors.

Rather than evaluating the efficacy of specific detectors, this work's novelty lies in addressing a more foundational question: *Are change detectors, when paired with fixed adaptation policies, theoretically capable of serving as effective adaptive mechanisms?*

3 Unactionability through General Unactionability

This section formally describes the conditions for unactionability. The intuition behind unactionability is as follows: given some drift signal, can one construct two or more drift events which produce said signal but require different adaptation strategies. If so, that signal is unactionable.

3.1 Formal Framework

Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the instance space, $\mathcal{Y}$ the label space, and $\mathcal{P} = \Delta(\mathcal{X} \times \mathcal{Y})$ the set of joint probability distributions over $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{H}$ denote a hypothesis space of models $h : \mathcal{X} \to \mathcal{Y}$ evaluated under a bounded loss function: $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, c], \qquad c < \infty$.

Let s_t denote the complete state of the stream learner immediately before an adaptation decision. The state may contain model parameters as well as any internal statistics required for future learning. An adaptation operator $\Omega : \mathcal{S} \to \mathcal{S}$ maps the current learner state to an adapted state. Let:

$$\mathbf{\Omega} = \{\Omega_1, \Omega_2, \ldots, \Omega_M\}$$

be a finite set of candidate adaptation operators.

When drift occurs, the data-generating process transitions from a reference distribution $P_t \in \mathcal{P}$ to a post-drift distribution $P_{t+1} \in \mathcal{P}$. We evaluate adaptation over a finite horizon $H \in \mathbb{N}$. Let:

$$(X_\tau, Y_\tau) \sim P_{t+1}, \qquad \tau = 1, \ldots, H.$$

For an adaptation operator Ω, the post-adaptation learner state is:

$$s_0^\Omega = \Omega(s_t),$$

and the learner subsequently updates according to its online learning rule:

$$s_{\tau+1}^\Omega = U\left(s_\tau^\Omega, (X_\tau, Y_\tau)\right).$$

Let $h(s)$ denote the predictive hypothesis induced by learner state s. The finite-horizon post-drift risk of an adaptation operator is defined as:

$$R_H(\Omega; s_t, P_{t+1}) = \mathbb{E}\left[\frac{1}{H}\sum_{\tau=1}^{H} \ell\left(h(s_\tau^\Omega)(X_\tau), Y_\tau\right)\right].$$

The set of optimal adaptation operators for a post-drift distribution is:

$$\Omega_H^*(P_{t+1}) = \arg\min_{\Omega \in \mathbf{\Omega}} R_H(\Omega; s_t, P_{t+1}).$$

The finite-horizon captures the fact that adaptation operations may trade immediate performance against subsequent recovery. For example, a reset may initially increase error by discarding a useful model state while subsequently improving performance as the learner observes post-drift data.

Definition 1 (Change Detector and Post-Drift Fibre)**.** *A **change detector** is an algorithm $T : \mathcal{P} \times \mathcal{P} \to \mathcal{Z}$, where $\mathcal{Z} = \{0, 1\}$ represents a binary value indicating whether drift has been detected. For any observed signal $z \in \mathcal{Z}$, the **post-drift fibre** $T^{-1}(z) \subseteq \mathcal{P}$ is the level set of post-drift distributions consistent with signal z:*

$$T^{-1}(z) = \{P \in \mathcal{P} \mid T(P_t, P) = z\} \tag{3}$$

*An **adaptation policy** conditioned strictly on the detector signal is a mapping $\pi : \mathcal{Z} \to \Delta(\mathbf{\Omega})$, assigning a probability distribution over $\mathbf{\Omega}$ given z.*

Remark 1. In practice, detector input signals vary. Unsupervised detectors typically take the covariates X as input while many supervised detectors will use the loss $\ell(h(X), Y)$ as input. We will first use an input-agnostic signal for the general unactionability theorem, which we later specialise.

3.2 General Unactionability Theorem

Definition 2 (Non-Subordinate Adaptation Operators)**.** *Two adaptation operators $\Omega_i, \Omega_j \in \mathbf{\Omega}$ are non-subordinate with respect to the finite-horizon risk R_H if there exist post-drift distributions $P^{(1)}, P^{(2)} \in \mathcal{P}$ such that:*

$$R_H(\Omega_i; s_t, P^{(1)}) < R_H(\Omega_j; s_t, P^{(1)}) \text{ and } R_H(\Omega_j; s_t, P^{(2)}) < R_H(\Omega_i; s_t, P^{(2)}).$$

Definition 3 (Finite-Horizon Excess Risk)**.** *For adaptation operator Ω and post-drift distribution P, define the finite-horizon excess risk as:*

$$\Delta_H(\Omega, P) = R_H(\Omega; s_t, P) - \min_{\Omega' \in \Omega} R_H(\Omega'; s_t, P).$$

Clearly, $\Delta_H(\Omega, P) \geq 0$, with $\Delta_H(\Omega, P) = 0 \iff \Omega \in \Omega^*_H(P)$.

Theorem 1 (Unactionability of Model Adaptation)**.** *Let $\mathcal{H}$ be a hypothesis space and let Ω contain at least two non-subordinate adaptation operators. For any reference learner state $s_t \in \mathcal{S}$, reference distribution $P_t \in \mathcal{P}$, and change detector $T : \mathcal{P} \times \mathcal{P} \to \mathcal{Z}$,if the post-drift fibre $T^{-1}(z) = \{P \in \mathcal{P} : T(P_t, P) = z\}$ contains no adaptation operator that is optimal for every distribution in the fibre, then for any adaptation policy $\pi : \mathcal{Z} \to \Delta(\Omega)$, there exists $\epsilon_H > 0$ such that:*

$$\sup_{P \in T^{-1}(z)} \mathbb{E}_{\Omega \sim \pi(z)} \left[R_H(\Omega; s_t, P) - \min_{\Omega' \in \Omega} R_H(\Omega'; s_t, P) \right] \geq \epsilon_H > 0.$$

Thus, a detector signal that is insufficient to identify the appropriate adaptation action necessarily incurs positive finite-horizon excess risk.

See Appendix A.1 for proof.

3.3 Unactionability for various drift types

In this section, we provide theorems for model-agnostic action unidentifiability for three categories of Concept Drift: (1) Posterior Drift, (2) Covariate Drift and (3) Representation Drift, demonstrating conditions under which binary change detection signals are unactionable. We choose these drift types because they cover the majority of drift dynamics discussed in literature (Cerqueira et al., 2026). In general, each theorem identifies two drift phenomena per drift type that can generically occur and produce the same drift signal. One phenomenon requires that no adaptation occurs despite detection while the other phenomenon requires adaptation upon detection. Given that both phenomena produce the same drift signal, but require different adaptation strategies, unactionability is established. We include the proof sketches for each in the Appendix.

Theorem 2 (Posterior Drift Unactionability)**.** *Let h_t be a predictive model trained to convergence on an initial distribution P_t. Let $T_{post} : \Delta(\mathcal{L}) \times \Delta(\mathcal{L}) \to \mathcal{Z}$ be a loss-based supervised change detector. Let $\mathbf{\Omega}_0$ denote the No-Action operator (retaining h_t) and $\mathbf{\Omega}_{\text{reset}}$ denote the Full Reset operator (re-initializing h_t to an empty model $\mathcal{H}_\emptyset$).*

There exist two distinct post-drift distributions $P^{(1)}_{t+1}$ (Bayes Error Inflation) and $P^{(2)}_{t+1}$ (Localised Concept Drift) such that:

1. *$T_{post}(P_t, P^{(1)}_{t+1}) = T_{post}(P_t, P^{(2)}_{t+1}) = z$ and $z > 0$.*
2. *$\Omega^*_H(P^{(1)}_{t+1}) = \{\Omega_0\}$ and $\Omega^*_H(P^{(2)}_{t+1}) = \{\Omega_{\text{reset}}\}$.*

Thus, no supervised detector signal z can determine if model adaptation is required.

Theorem 3 (Covariate Drift Unactionability)**.** *Let $T_{cov} : \Delta(\mathcal{X}) \times \Delta(\mathcal{X}) \to \mathcal{Z}$ be an unsupervised change detector evaluating feature covariates $P_t(X)$ and $P_{t+1}(X)$. Let h_t be a model trained to convergence on an initial distribution P_t. Let Ω_0 denote the No-Action operator (retaining h_t) and Ω_{reset} denote the Full Reset operator (re-initializing h_t to an empty model $\mathcal{H}_\emptyset$).*

There exist two post-drift joint distributions $P^{(1)}_{t+1}$ (Covariate Drift without Posterior Drift) and $P^{(2)}_{t+1}$ (Covariate Drift with Posterior Drift) over $\mathcal{X} \times \mathcal{Y}$ such that:

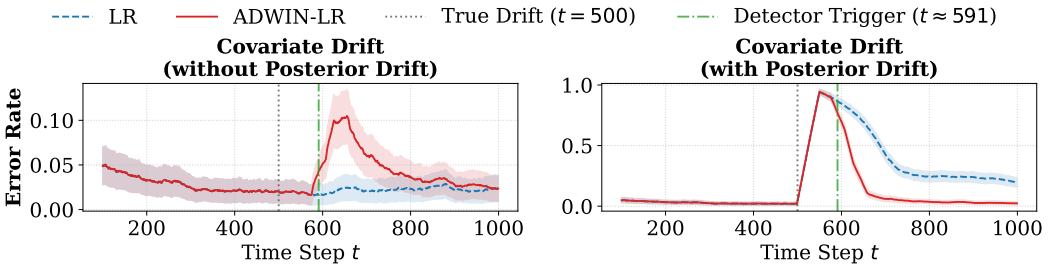

Figure 1: **Unactionability of Change Detection.** Average Error Rate of Logistic Regression without a change detector (LR) vs. ADWIN-triggered reset (ADWIN-LR) for covariate drift scenarios: covariate drift without (left) and with (right) posterior drift resp. Unactionability is established due to ADWIN triggering in both cases (i.e., same drift signal), but having different best-performing adaptation policies (no-reset for the left figure, and reset for the right).

1. $T_{cov}(P_t, P_{t+1}^{(1)}) = T_{cov}(P_t, P_{t+1}^{(2)}) = z$ *and* $z > 0$.
2. $\Omega_H^*(P_{t+1}^{(1)}) = \{\Omega_0\}$ *and* $\Omega_H^*(P_{t+1}^{(2)}) = \{\Omega_{\text{reset}}\}$.

Thus, no unsupervised detector signal z can determine if model adaptation is required.

Theorem 4 (Representation Drift Unactionability)**.** *Let $h_t = g_\phi \circ f_\psi \in \mathcal{H}$ be a compositional hypothesis (e.g., a neural network) comprising a feature extractor $f_\psi : \mathcal{X} \to \mathcal{Z}_{lat}$ and a decision head $g_\phi : \mathcal{Z}_{lat} \to \mathcal{Y}$, trained to convergence on an initial distribution P_t. Let $T_{repr} : \Delta(\mathcal{Z}_{lat}) \times \Delta(\mathcal{Z}_{lat}) \to \mathcal{Z}$ be a representation-based change detector monitoring the distribution of latent embeddings $P(Z_{lat})$. Let Ω_0 denote the No-Action operator (retaining h_t) and Ω_{reset} denote the Head Reset operator (re-initializing parameters ϕ while preserving ψ).*

There exist two post-drift joint distributions $P_{t+1}^{(1)}$ (uninformative subspace drift) and $P_{t+1}^{(2)}$ (latent task inversion) over $\mathcal{X} \times \mathcal{Y}$ such that:

1. $T_{repr}(P_t, P_{t+1}^{(1)}) = T_{repr}(P_t, P_{t+1}^{(2)}) = z$ *and* $z > 0$
2. $\Omega_H^*(P_{t+1}^{(1)}) = \{\Omega_0\}$ *and* $\Omega_H^*(P_{t+1}^{(2)}) = \{\Omega_{\text{reset}}\}$.

Thus, no representation drift detector signal z can determine if model adaptation is required.

3.4 Empirical Validation

We explore Theorems 2-4 on synthetic data streams. Due to page limitations we only describe the experimental setup and results for the covariate setting (Theorem 3) and refer the reader to section B in the appendix for the validation of the posterior and representation drift settings as well as additional experimental details. Unless stated otherwise, the post-drift time horizon is $H = 500$.

Consider a data stream where features are sampled uniformly over domain $\mathcal{X} = [0, 1]^2$ with pre-drift (P_0) boundary $Y = \mathbb{I}(X_1 + X_2 > 1)$. Two Logistic Regression models (with differing adaptation policies) are deployed on this stream. The first model (LR) simply trains on incoming samples in an online fashion, the second model (ADWIN-LR) also trains in an online fashion but will reset itself when ADWIN (which is monitoring the feature covariates) detects drift. Both models are initially trained on 500 instances sampled from P_0.

At timestep $t = 500$, one of two drift events are induced: (1) covariate drift without posterior shift: whereby samples from the post-drift distribution P_1 are uniformly sampled from domain $\mathcal{X} = [0.5, 1]^2$, and (2) covariate drift with posterior shift: whereby the same domain shift occurs $\mathcal{X} = [0.5, 1]^2$ but the decision boundary also shifts to $Y = 1 - \mathbb{I}(X_1 + X_2 > 1)$. We then train both models on $H = 500$ samples from the post-drift distribution P_1 recording the error rate.

Fig. 1 reports the prequential error across 200 independent stream replications across both scenarios (100 per scenario). Note that in both post-drift scenarios, ADWIN detects drift at the same timestep

($t \approx 591$) which establishes the identical drift signal condition necessary for determining unactionability. Fig. 1 (left) shows the error for the scenario with only covariate shift, here we see that LR outperforms ADWIN-LR because the detection signal triggers a reset operation. Fig. 1 (right) shows the error for the scenario with both covariate and posterior shift, here we see that ADWIN-LR outperforms LR because the reset operation enables forgetting of the old decision boundary. Given these results, we establish two non-subordinate adaptation operations (i.e., no action is preferred in the first scenario, while resetting is preferred in the second). With both identical detector response and the presence of two or more non-subordinate adaptation operators, unactionability is established.

Remark 2. **Magnitude of Misadaptation**: The excess risk incurred from misadaptation may be non-uniform across data streams. In Fig. 1, the cost of resetting in the stream with no posterior drift is not equivalent to the cost of not resetting in the stream with posterior drift. This may motivate one to choose ADWIN-LR as the preferred method because its average excess risk is lower across the two drift scenarios. The efficacy of this conclusion relies on the availability of post-drift event labelled samples. This work does not investigate missing or delayed labels. In such settings, the consequence of misadaptation is expected to be greater.

4 Information Limits of Change Detection for Model Adaptation

Section 3 establishes that a binary drift signal z is unidentifiable over candidate adaptation operators. A natural counter-hypothesis is that actionability can be attained either by refining detection signal or by deploying multi-detector architectures (often done in practice). We now show that detector capacity limits the information available for adaptation. When the entropy of the optimal adaptation decision grows relative to detector capacity, this limitation induces a non-zero lower bound on the probability of misadaptation. Proofs and an empirical exploration are included in the Appendix.

Recall that we model concept drift as a change in generating distribution from one timestep to another $P_t \neq P_{t+1}$. Let $W \sim \mu_{P_t}$ denote a random post-drift distribution. Because $\Omega^*_H(W)$ may contain multiple optimal operators, we fix an arbitrary deterministic tie-breaking rule and let $\Omega^\dagger_H(W) \in \Omega^*_H(W)$ denote the resulting selected optimal adaptation operator. We write $\Omega^\dagger_H$ as a discrete random variable induced by the distribution of W. For the remainder of this section, we assume that the optimal adaptation operator is unique, i.e., $|\Omega^*_H(W)| = 1$ Thus, $\Omega^\dagger_H(W)$ uniquely identifies the optimal adaptation operator.

A change detector acts as a lossy compression channel $T : \mathcal{P} \times \mathcal{P} \to \mathcal{Z}$, mapping the high-dimensional distribution W to a low-dimensional detector signal $Z = T(P_t, W) \in \mathcal{Z} = \{0, 1\}$. An adaptation policy π attempts to identify the optimal operator(s) $\mathbf{\Omega}^*$ from Z.

Theorem 5 (Information Bottleneck Inequality for Model Adaptation)**.** *For any pre-drift distribution P_t, random post-drift distribution $W \sim \mu_{P_t}$, change detector $T : \mathcal{P} \times \mathcal{P} \to \mathcal{Z}$, and adaptation policy $\pi : \mathcal{Z} \to \Delta(\Omega)$, let $Z = T(P_t, W)$ and $\widehat{\Omega} \sim \pi(Z)$. Then:*

$$I(\Omega^\dagger_H; \widehat{\Omega}) \leq I(\Omega^\dagger_H; Z) \leq I(W; Z) \leq H(Z).$$

Furthermore, the minimum probability of selecting a non-optimal adaptation operator $P_e = \Pr\left(\widehat{\Omega} \notin \Omega^*_H(W)\right)$ satisfies:

$$P_e \geq \frac{H(\Omega^\dagger_H) - H(Z) - 1}{\log |\Omega|}.$$

Corollary 6 (Complexity-Dependent Information Bottleneck)**.** *Suppose that the entropy of the optimal adaptation decision satisfies $H(\Omega^\dagger_H) \geq c \log |\Omega|$ for some constant $c > 0$. Then $P_e \geq c - \frac{H(Z)+1}{\log |\Omega|}$. In particular, for a binary detector $H(Z) \leq 1$, hence $P_e \geq c - \frac{2}{\log |\Omega|}$ Thus, when the entropy of the optimal adaptation decision grows proportionally to the logarithm of the adaptation space while detector capacity remains fixed, the probability of misadaptation is bounded away from zero as $|\Omega|$ increases.*

Remark 3 (**Adaptation Complexity.**). The information bottleneck depends on the entropy of the optimal adaptation decision rather than solely on the cardinality of the adaptation space. Although architectural complexity can increase the number of possible adaptation operations, a large action space does not necessarily imply a high-entropy optimal decision. The complexity-dependent lower bound therefore requires the additional condition that $H(\Omega^\dagger_H)$ grows with $\log |\Omega|$.

For example, consider a balanced decision tree of depth d with adaptation operators corresponding to valid subtree reset operations. The number of possible adaptation operations follows the recurrence

$$A(d) = 1 + [A(d-1)]^2, \qquad A(1) = 2.$$

For $d = 5$, this gives $|\Omega| = 458{,}330$. If the optimal adaptation decisions are approximately uniformly distributed over these operations, then $H(\Omega_H^\dagger) \approx \log|\Omega| \approx 18.8$ bits, while a binary detector has at most one bit of information. Corollary 6 then predicts a non-zero lower bound on misadaptation.

Theorem 5 shows that the information available to an adaptation policy is bounded by the information exposed by the detector. Corollary 6 further shows that increasing adaptation complexity creates an asymptotic information bottleneck when the entropy of the optimal adaptation decision grows proportionally with the logarithm of the action space.

5 Case Study: USP DS Repository

We investigate the practical implications of our theoretical findings by evaluating four classes of algorithms (Decision Trees, Ensembles, kNNs, and Neural Networks) on 24 classification data streams taken from the USP DS repository (Souza et al., 2020). These streams consist of semi-real and real-world data streams from several domains such as Health Care, Meteorology and Urban Development, exhibiting various drift types. For the incremental Decision Trees, we investigate 7 algorithms that adopt varying adaptation policies: (1) EFDT: an incremental decision tree with no drift adaptation policy (Manapragada et al., 2018). (2) HAT: uses ADWIN to replace subtrees (Bifet & Gavalda, 2009). (3) LAST: uses ADWIN to determine when to make node splits (Assis et al., 2024) and (4) ADWIN-EFDT, (5) PH-EFDT, (6) KS-EFDT, (7) D3-EFDT: which use the ADWIN (Bifet & Gavalda, 2007), Page-Hinkley (Sebastião & Fernandes, 2017), Kolmogorov-Smirnov (Raab et al., 2020), and D3 (Gözüaçık et al., 2019) detectors respectively and completely reset themselves when drift is detected. For the ensemble architectures we investigate: (1) ARF (Gomes et al., 2017), (2) SRP (Gomes et al., 2019) and (3) DEMS Sun et al. (2025): which all use ADWIN to replace base learners in the ensemble, and (4) AMF (Mourtada et al., 2021): which does not use change detectors. We also investigate variants of ARF, SRP, and DEMS that have their detectors disabled (i.e., they have No Change detectors: NC-ARF, NC-SRP, NC-DEMS resp.) For the kNN variants, we investigate standard online (1) kNN with sliding window (Read et al., 2012), (2) w-kNN which uses a weighted sliding window (Bahri, 2022), and (3) SAMkNN which automatically adjusts its sliding windows using heuristic rules (Losing et al., 2016). For the Neural Networks, we initialise a standard MLP with a hidden layer equal to size $(d + c)/2$ where d is the number of features and c is the number of classes. We explore three variants: (1) ANN: with no change detection. (2) Reset (R-ANN): neural network that resets itself when ADWIN detects drift. (3) Head (H-ANN): A network that resets the classification head when ADWIN detects drift. We use a sliding window of 500 samples as a replay buffer when training the networks. Lastly, we also explore (4) CAND which uses an ensemble of neural networks and ADWIN detectors for selecting the appropriate network for prediction (Gunasekara et al., 2022).

It is infeasible to exhaustively search for optimal adaptation strategies in real streams to formally verify unactionability. Instead, we investigate an observable consequence of the theory: whether the utility of fixed detector-triggered adaptation is consistent across datasets and learning architectures. We do this by performing prequential (test-then-train) evaluation on each method for each stream. We use Cohen's Kappa (Cohen, 1960) as the evaluation metric and 10 pseudo-random replicates are run for all stochastic algorithms. Fractional-based ranking for each algorithm on each stream is determined by a Mann-Whitney U test ($\alpha = 0.05$). A method that ranks higher than another on a stream is considered to have a better adaptation strategy. To determine if one method performs better than another across all streams, fractional ranks are used alongside a Friedman Test ($\alpha = 0.05$) to identify within-group differences and Pairwise Wilcoxon Signed-Rank Test. ($\alpha = 0.05$) is used to identify between algorithm differences. Each class of methods is investigated independently to isolate the effects of the adaptive mechanisms. See the appendix for additional details.

Table 1 reports the results. No significant difference ($p = 0.36$) in performance was found across the 7 decision trees algorithms investigated. No significant difference was found for the 3 kNN architectures ($p = 0.45$). For the ensemble architectures, SRP and DEMS performed significantly better

Table 1: Average rank across 24 streams from the USP DS Repository (lower is better). Bold values indicate statistically best-performing methods within a model class (column).

Decision Trees ($p = 0.36$)		**Ensembles** ($p = 0.01$)		**Neural Networks** ($p < 0.01$)		**kNNs** ($p = 0.45$)	
ADWIN-EFDT	3.73	AMF	4.65	**ANN**	**1.90**	kNN	2.04
D3-EFDT	4.21	ARF	4.19	**CAND**	**2.06**	SAMkNN	1.85
EFDT	4.00	NC-ARF	4.33	**H-ANN**	**2.56**	w-kNN	2.10
HAT	4.33	**DEMS**	**3.25**	R-ANN	3.48		
KS-EFDT	3.48	NC-DEMS	4.31				
LAST	4.31	**SRP**	**3.13**				
PH-EFDT	3.83	NC-SRP	4.15				

($p = 0.01$) than the other 5 methods investigated. Finally, ANN, CAND and H-ANN performed significantly better than R-ANN ($p < 0.01$). These results suggest that the utility of fixed adaptation strategies (via change detection) is architecture specific.

Fig. 2 plots the win-loss ratios of methods that do not use change detectors as adaptive mechanisms vs. architectural equivalents that do. For the decision tree architectures, results demonstrate that no single fixed detector-triggered adaptation policy consistently dominates the evaluated alternatives. e.g., EFDT is strictly better or equal in predictive performance to its architectural equivalents on 10 streams. This finding is not limited to detector types: EFDT is better than or equivalent to ADWIN-EFDT (ADWIN is a supervised detector) on 10 streams and D3-EFDT (D3 is an unsupervised detector) on 12 stream resp. Nor is it limited to adaptation policies where EFDT is better than or equivalent to HAT (which uses detectors to replace subtrees) on 17 streams and LAST (which uses detectors to determine when to split) on 15 streams. These findings are further supported by the passively adapting KNN and w-KNN models which outperform SAMkNN (which uses heuristic rules to reintegrate past samples) on 10 streams.

For the ensemble models, results show that ARF, DEMS and SRP are generally equivalent or better than their variants which do not use change detectors to reset base learners NC-ARF, NC-DEMS and NC-SRP which lose or draw 18, 23 and 23 times to ARF, DEMS and SRP resp. In general, these results show ARF, DEMS and SRP (which were designed to use change detectors) perform better when they utilise change detectors to reset their base learners. Interestingly, AMF (which does not use change detectors as adaptive mechanisms) outperforms or draws to ARF, DEMS and SRP on 10, 10 and 9 streams resp. Further investigation (see Fig. 7 in the appendix) reveals that when comparing AMF to the detector-less variants NC-ARF, NC-DEMS, NC-SRP, the win-loss ratio does not change significantly. The ensemble results indicate that detector-triggered replacement can be beneficial for some architectures, while the AMF comparisons show that competitive performance is also possible without such detector-triggered adaptation.

Finally, a base ANN with an experience replay buffer outperforms or draws against H-ANN and R-ANN (which use reset mechanisms determined by ADWIN) on 23 and 22 of the streams resp., and CAND (an ensemble of networks) on 10 streams. The neural-network results show substantial variation in the effect of reset mechanisms, with the detector-free ANN performing at least as well as the evaluated reset-based alternatives on most of the streams. These results suggest that ANN reset adaptation mechanisms may be ill-suited for ANNs. As noted in Sect. 2, there does not appear to be consensus on how change detectors should be used when deploying neural networks.

5.1 Beyond fixed adaptation policies

In summary, our evaluation demonstrates substantial heterogeneity in the utility of fixed detector-triggered adaptation across learning architectures and data streams. We do not suggest that change detectors are useless. For many of the streams investigated, using a change detector to trigger a reset mechanism was clearly the better strategy. This work's critique of the state-of-the-art lies in the over-reliance of fixed adaptation policies, and advocating for the development of learnable adaptation policies. Change detectors may play a role in such a process. Findings from this work suggest that a learnable adaptation policy should: (1) consider the algorithm it is adapting. e.g., Reset mechanisms may not be appropriate for neural networks, but highly beneficial for ensemble architectures. (2) consider the dynamics of the data stream. e.g., covariate drift may require action

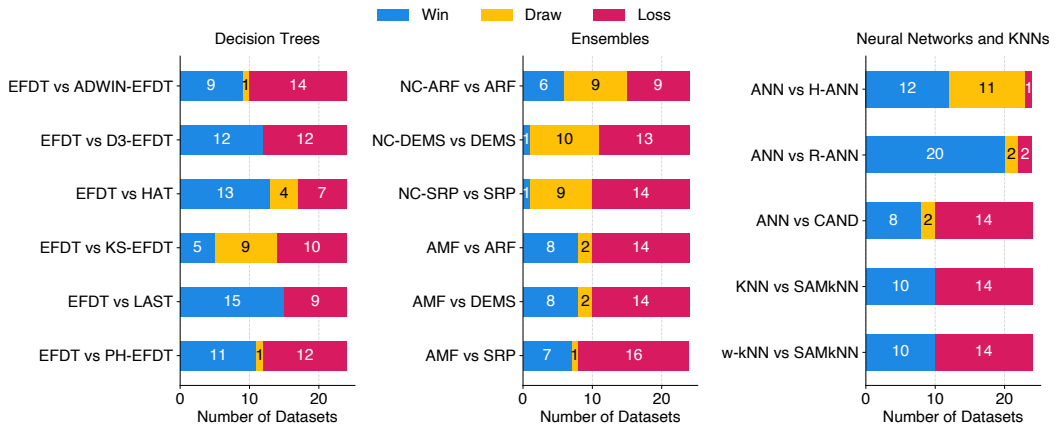

Figure 2: **Win-Loss ratios**: of algorithms evaluated on the USP DS repository. Results are separated by class of models (columns) and comparisons are done between algorithms that do not use change detectors as adaptive mechanisms (left) and those that do (right) (e.g., AMF vs. ARF, AMF does not use change detectors, while ARF does).

in one stream while requiring inaction in another. To achieve this, multiple information signals from the stream may be required.

6 Conclusions and Future Work

This work studied whether fixed adaptation policies driven by change detector signals are sufficient to select appropriate adaptation operations in online learning. Our theoretical results show that detector signals can be unactionable when distinct post-drift environments produce identical detector observations but favour different adaptation operations. We further show that limited detector information can constrain adaptation decisions through an information-theoretic bottleneck. Controlled synthetic experiments explore these theoretical constructions, while the USP DS case study demonstrates substantial heterogeneity in the utility of fixed detector-triggered adaptation across learning architectures and data streams. These findings suggest that effective adaptation may benefit from richer signals and policies that account jointly for the learner state and the dynamics of the evolving data stream. Learning-based approaches, including reinforcement learning, provide one possible direction for learning such adaptation policies online.

AI use statement

In this work, we used generative AI tools for: (1) Reviewing text for clarity and suggesting changes (e.g., identifying spelling mistakes) (2) Reviewing / Reformatting proofs. For example, we originally called the post-drift fibre a reverse generator function. Gemini pointed out the set notation term for such a function is a fibre. Gemini was also used to ensure the notation of the drift scenario theorems were consistent. Additionally, (3) we asked Gemini to generate stylesheets for the Figures included in this work.

We have not used generative AI tools for: (1) Generating synthetic data sets (2) Implementing methods, (3) assisting with translation, cleaning and reformatting datasets, (4) supporting qualitative and thematic data analysis. In general, all edits to the manuscript were done by hand (even those suggested by generative AI).

We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

To ensure reproducibility, we will make the source code for both the synthetic and empirical studies available upon publication. We include a reference to all datasets, a description of all pre-processing steps, the random seeds, and list all software packages used in the Appendix.

References

Daniel Nowak Assis, Jean Paul Barddal, and Fabrício Enembreck. Just change on change: Adaptive splitting time for decision trees in data stream classification. In *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, pp. 351–357, 2024.

Maroua Bahri. Effective weighted k-nearest neighbors for dynamic data streams. In *2022 IEEE International Conference on Big Data (Big Data)*, pp. 3341–3347. IEEE, 2022.

Firas Bayram, Bestoun S Ahmed, and Andreas Kassler. From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, 245:108632, 2022.

Albert Bifet. Classifier concept drift detection and the illusion of progress. In *Int. Conf. on Artificial Intelligence and Soft Computing*, pp. 715–725. Springer, 2017.

Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proc. of the 2007 SIAM Int. Conf. on Data Mining*, pp. 443–448. SIAM, 2007.

Albert Bifet and Ricard Gavalda. Adaptive learning from evolving data streams. In *Int. Symposium on Intelligent Data Analysis*, pp. 249–260. Springer, 2009.

Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Richard Kirkby, and Ricard Gavalda. New ensemble methods for evolving data streams. In *Proc. of the 15th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pp. 139–148, 2009.

Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. Leveraging bagging for evolving data streams. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 135–150. Springer, 2010.

Jock A Blackard and Denis J Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and Electronics in Agriculture*, 24(3):131–151, 1999.

Sebastian Buschjäger, Sibylle Hess, and Katharina J Morik. Shrub ensembles for online classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6123–6131, 2022.

Vitor Cerqueira, Heitor Murilo Gomes, Albert Bifet, and Luis Torgo. STUDD: A student-teacher method for unsupervised concept drift detection. *Machine Learning*, 112(11):4351–4378, 2023.

Vitor Cerqueira, Heitor Murilo Gomes, Marco Heyden, Bernhard Pfahringer, and Albert Bifet. A framework for evaluating and benchmarking concept drift detection methods. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 8671–8682, 2026.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. The UCR Time Series Classification Archive, 2018.

Dua Dheeru and E Karra Taniskidou. UCI Machine Learning Repository. 2017.

Gregory Ditzler and Robi Polikar. Incremental learning of concept drift from streaming imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2283–2301, 2012.

Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In *Brazilian Symposium on Artificial Intelligence*, pp. 286–295. Springer, 2004.

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

Heitor M Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfharinger, Geoff Holmes, and Talel Abdessalem. Adaptive random forests for evolving data stream classification. *Machine Learning*, 106(9):1469–1495, 2017.

Heitor Murilo Gomes, Jesse Read, and Albert Bifet. Streaming random patches for evolving data stream classification. In *2019 IEEE international conference on data mining (ICDM)*, pp. 240–249. IEEE, 2019.

Heitor Murilo Gomes, Anton Lee, Nuwan Gunasekara, Yibin Sun, Guilherme Weigert Cassales, Justin Jia Liu, Marco Heyden, Vitor Cerqueira, Maroua Bahri, Yun Sing Koh, Bernhard Pfahringer, and Albert Bifet. CapyMOA: Efficient machine learning for data streams in python, 2025. URL https://arxiv.org/abs/2502.07432.

Brandon Gower-Winter, Misja Groen, and Georg Krempl. The window dilemma: Why concept drift detection is ill-posed. In *International Symposium on Intelligent Data Analysis*, pp. 371–384. Springer, 2026.

Ömer Gözüaçık, Alican Büyükçakır, Hamed Bonab, and Fazli Can. Unsupervised concept drift detection with a discriminative classifier. In *Proc. of the 28th ACM Int. Conf. on Information and Knowledge Management*, pp. 2365–2368, 2019.

Nuwan Gunasekara, Heitor Murilo Gomes, Bernhard Pfahringer, and Albert Bifet. Online hyperparameter optimization for streaming neural networks. In *2022 international joint conference on neural networks (IJCNN)*, pp. 1–9. IEEE, 2022.

Michael Harries. Splice-2 comparative evaluation: Electricity pricing. 1999.

Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9769–9776. IEEE, 2019.

Fabian Hinder, Valerie Vaquet, Johannes Brinkrolf, and Barbara Hammer. On the change of decision boundary and loss in learning with concept drift. In *Int. Symposium on Intelligent Data Analysis*, pp. 182–194. Springer, 2023.

Fabian Hinder, Valerie Vaquet, and Barbara Hammer. One or two things we know about concept drift—a survey on monitoring in evolving environments. part a: detecting concept drift. *Frontiers in Artificial Intelligence*, 7:1330257, 2024.

J Zico Kolter and Marcus A Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research*, 8:2755–2790, 2007.

Ludmila I Kuncheva and Indrė Žliobaitė. On the window size for classification in changing environments. *Intelligent Data Analysis*, 13(6):861–872, 2009.

Viktor Losing, Barbara Hammer, and Heiko Wersing. Interactive online learning for obstacle classification on a mobile robot. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2015.

Viktor Losing, Barbara Hammer, and Heiko Wersing. Knn classifier with self adjusting memory for heterogeneous concept drift. In *IEEE 16th Int. Conf. on Data Mining (ICDM)*, pp. 291–300. IEEE, 2016.

Pengqian Lu, Jie Lu, Anjin Liu, and Guangquan Zhang. Early concept drift detection via prediction uncertainty. In *Proc. of the AAAI Conf. on Artificial Intelligence*, volume 39, pp. 19124–19132, 2025.

Daniel Lukats, Oliver Zielinski, Axel Hahn, and Frederic Stahl. A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams. *Int. Journal of Data Science and Analytics*, 19(1):1–31, 2025.

Chaitanya Manapragada, Geoffrey I Webb, and Mahsa Salehi. Extremely fast decision tree. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1953–1962, 2018.

Jacob Montiel, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine, Heitor Murilo Gomes, Jesse Read, Talel Abdessalem, et al. River: machine learning for streaming data in python. 2021.

Jaouad Mourtada, Stéphane Gaïffas, and Erwan Scornet. Amf: Aggregated mondrian forests for online learning. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 83(3): 505–533, 2021.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Christoph Raab, Moritz Heusinger, and Frank-Michael Schleif. Reactive soft prototype computing for concept drift streams. *Neurocomputing*, 416:340–351, 2020.

Jesse Read and Indrė Žliobaitė. Supervised learning from data streams: An overview and update. *ACM Computing Surveys*, 2025.

Jesse Read, Albert Bifet, Bernhard Pfahringer, and Geoff Holmes. Batch-incremental versus instance-incremental learning in dynamic and evolving data. In *International symposium on intelligent data analysis*, pp. 313–323. Springer, 2012.

Denis Moreira Dos Reis, Peter Flach, Stan Matwin, and Gustavo Batista. Fast unsupervised online drift detection using incremental kolmogorov-smirnov test. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1545–1554, 2016.

Raquel Sebastião and José Maria Fernandes. Supporting the page-hinkley test with empirical mode decomposition for change detection. In *Int. Symposium on Methodologies for Intelligent Systems*, pp. 492–498. Springer, 2017.

Ricardo Petri Silva, Sylvio Barbon Junior, Bruno Bogaz Zarpelão, and Leonimer Flavio De Melo. Unsupervised tuning for drift detectors using change detector segmentation. *IEEE Access*, 12: 54256–54271, 2024.

Vinícius M. A. Souza, Diego F. Silva, João Gama, and Gustavo E. A. P. A. Batista. Data stream classification guided by clustering on nonstationary environments and extreme verification latency. pp. 873–881, 2015.

Vinicius M. A. Souza, Denis M. dos Reis, André G. Maletzke, and Gustavo E. A. P. A. Batista. Challenges in benchmarking stream learning algorithms with real-world data. *Data Mining and Knowledge Discovery*, 34:1805–1858, 2020.

Vinicius MA Souza. Asphalt pavement classification using smartphone accelerometer and complexity invariant distance. *Engineering Applications of Artificial Intelligence*, 74:198–211, 2018.

Vinicius MA Souza, Antonio RS Parmezan, Farhan A Chowdhury, and Abdullah Mueen. Efficient unsupervised drift detector for fast and high-dimensional data streams. *Knowledge and Information Systems*, 63(6):1497–1527, 2021.

Yibin Sun, Bernhard Pfahringer, Heitor Murilo Gomes, and Albert Bifet. Dynamic ensemble member selection for data stream classification. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 2821–2831, 2025.

Liudmila Ulanova, Nurjahan Begum, Mohammad Shokoohi-Yekta, and Eamonn Keogh. Clustering in the face of fast changing streams. In *Proceedings of the 2016 SIAM international conference on data mining*, pp. 1–9. SIAM, 2016.

Alexander Vergara, Shankar Vembu, Tuba Ayhan, Margaret A Ryan, Margie L Homer, and Ramón Huerta. Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, 166:320–329, 2012.

Ke Wan, Yi Liang, and Susik Yoon. Online drift detection with maximum concept discrepancy. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 2924–2935, 2024.

Yi Min Wen, Xiang Liu, and Hang Yu. Adaptive tree-like neural network: Overcoming catastrophic forgetting to classify streaming data with concept drifts. *Knowledge-Based Systems*, 293:111636, 2024.

En Yu, Jie Lu, Kun Wang, Xiaoyu Yang, and Guangquan Zhang. Drift-aware collaborative assistance mixture of experts for heterogeneous multistream learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 16199–16207, 2026.

Xingquan Zhu. Stream data mining repository. *URL:http://www.cse.fau.edu/xqzhu/stream.html*, 2010.

Indrė Žliobaitė. Combining similarity in time and space for training set formation under concept drift. *Intelligent Data Analysis*, 15(4):589–611, 2011.

A PROOFS

This sections contains proofs for all theorems in this work.

A.1 PROOF OF THEOREM 1: UNACTIONABILITY

Proof. Let $T^{-1}(z) = \{P \in \mathcal{P} : T(P_t, P) = z\}$ be the post-drift fibre associated with detector signal z. By assumption, $\bigcap_{P \in T^{-1}(z)} \Omega_H^*(P) = \varnothing$. Because $\mathbf{\Omega}$ is finite, there exists a finite collection $\mathcal{P}_z = \{P^{(1)}, \ldots, P^{(K)}\} \subseteq T^{-1}(z)$ such that $\bigcap_{k=1}^{K} \Omega_H^*(P^{(k)}) = \varnothing$.

For any adaptation operator $\Omega_m \in \mathbf{\Omega}$, define:

$$\Delta_H(\Omega_m, P^{(k)}) = R_H(\Omega_m; s_t, P^{(k)}) - \min_{\Omega' \in \mathbf{\Omega}} R_H(\Omega'; s_t, P^{(k)}).$$

Since the optimal-action sets have empty intersection, for every $\Omega_m \in \Omega$ there exists some $P^{(k)} \in \mathcal{P}_z$ for which $\Delta_H(\Omega_m, P^{(k)}) > 0$. Because Ω and $\mathcal{P}_z$ are finite, there exists $\epsilon_H > 0$ such that:

$$\max_{1 \le k \le K} \Delta_H(\Omega_m, P^{(k)}) \ge \epsilon_H \qquad \forall \Omega_m \in \mathbf{\Omega}.$$

Let $p_m = \pi(\Omega_m \mid z), \qquad \sum_{m=1}^{M} p_m = 1$. Since every $P^{(k)}$ belongs to the same detector fibre, the adaptation policy must assign the same action distribution $\{p_m\}$ to every $P^{(k)}$. Therefore:

$$\sup_{P \in T^{-1}(z)} \mathbb{E}_{\Omega \sim \pi(z)}[\Delta_H(\Omega, P)] \ge \max_{1 \le k \le K} \sum_{m=1}^{M} p_m \Delta_H(\Omega_m, P^{(k)}) \quad \ge \epsilon_H.$$

This establishes the stated positive lower bound. □

Note that for the following Theorems 2 - 4, the general proof sketch follows: for the finite adaptation horizon H considered, each scenario is constructed such that retaining the existing learner has lower finite-horizon risk in the first setting, while reset has lower finite-horizon risk in the second. From this, we establish unactionability.

A.2 Proof of Theorem 2: Posterior Drift

Proof Sketch. Let $\mathcal{X} = [0,1]^2$ with uniform measure, and P_t have optimal decision boundary $Y = \mathbb{I}(X_1 + X_2 > 1)$. Let $h_t \in \mathcal{H}$ be a stream learner initialised with state $s_0 \in \mathcal{S}$ and adaptation operator $\omega \in \Omega$ trained to zero error ($R_H(\omega; s_0, P_t) = 0$).

Construction of $P_{t+1}^{(1)}$ (Bayes Error Inflation): Define $P_{t+1}^{(1)}$ by adding uniform symmetric label noise with probability z. The true posterior becomes $\mathbb{P}_{P_{t+1}^{(1)}}(Y = 1 \mid X) = (1-2z)\mathbb{I}(X_1+X_2 > 1)$. Because the decision boundary is geometrically unchanged, h_t remains the Bayes optimal classifier, achieving immediate expected error $R_H(\omega; s_t, P_{t+1}^{(1)}) = z$. Executing $\mathbf{\Omega}_{\text{reset}}$ resets h_t. The reset model must re-learn on noisy observations, incurring a penalty over H post-drift samples before recovering the original h_t model. Thus, $\Omega_H^*(P_{t+1}^{(1)}) = \{\Omega_0\}$.

Construction of $P_{t+1}^{(2)}$ (Localized Concept Shift): Define $P_{t+1}^{(2)}$ by inverting labels within sub-region $X_{\text{sub}} = \{X \in \mathcal{X} \mid X_1 \leq \sqrt{z} \text{ and } X_2 \leq \sqrt{z}\}$. The pre-drift model h_t predicts incorrect labels on X_{sub}, yielding error $R_H(\omega; s_t, P_{t+1}^{(2)}) = z$. Retaining h_t ($\mathbf{\Omega}_0$) results in new persistent error-rate z. Executing $\mathbf{\Omega}_{\text{reset}}$ allows $\mathcal{H}_\emptyset$ to learn the new boundary at $X_1, X_2 = \sqrt{z}$, achieving converged post-drift error of 0.0. Thus, $\Omega^*(P_{t+1}^{(2)}) = \{\Omega_{\text{reset}}\}$.

Since $T_{\text{post}}(P_t, P_{t+1}^{(1)}) = T_{\text{post}}(P_t, P_{t+1}^{(2)}) = z$, the detector signal is identical, establishing adaptation-incompatibility over $\{\Omega_0, \Omega_{\text{reset}}\}$. □

A.3 Proof of Theorem 3: Covariate Drift

Proof Sketch. Let $\mathcal{X} = [0,1]^2$ with uniform measure, and P_0 have optimal decision boundary $Y = \mathbb{I}(X_1 + X_2 > 1)$. Let $h_0 \in \mathcal{H}$ be a stream learner initialised with state $s_0 \in \mathcal{S}$ and adaptation operator $\omega \in \Omega$ trained to zero error ($R_H(\omega; s_0, P_0) = 0$).

Induce a marginal feature shift by constructing post-drift marginals $P_{t+1}^{(1)}(X) = P_{t+1}^{(2)}(X) = \mathcal{U}([0, 0.5] \times [0, 1])$, restricting feature support to $X_{\text{post}} = \{X \in \mathcal{X} \mid X_1 \leq 0.5\}$. Because the feature marginal shift is identical, any unsupervised detector yields $T_{\text{cov}}(P_t(X), P_{t+1}^{(1)}(X)) = T_{\text{cov}}(P_t(X), P_{t+1}^{(2)}(X)) = z > 0$.

Construction of $P_{t+1}^{(1)}$ ($P(X)$ Drift): Retain the true posterior $Y = \mathbb{I}(X_1 + X_2 > 1)$. Because the conditional decision boundary is geometrically unchanged on X_{post}, the active hypothesis h_t retains optimal performance. Executing Ω_{reset} forces unnecessary re-learning on H post-drift samples, incurring strict excess sample complexity risk before recovery. Thus, $\Omega^*(P_{t+1}^{(1)}) = \{\Omega_0\}$.

Construction of $P_{t+1}^{(2)}$ ($P(X)$ and $P(Y|X)$ Drift): Invert the posterior over X_{post} such that $Y = 1 - \mathbb{I}(X_1 + X_2 > 1)$.. The pre-drift hypothesis h_t now yields maximum classification error. Executing Ω_0 leads to persistent non-zero error, whereas executing Ω_{reset} allows the re-initialized model to learn the inverted decision boundary on X_{post} over time horizon H, minimizing post-drift risk. Thus, $\Omega^*(P_{t+1}^{(2)}) = \{\Omega_{\text{reset}}\}$.

Since $T_{\text{cov}}(P_t(X), P_{t+1}^{(1)}(X)) = T_{\text{cov}}(P_t(X), P_{t+1}^{(2)}(X)) = z$, the unsupervised drift signal z is adaptation-incompatible over $\{\Omega_0, \Omega_{\text{reset}}\}$. □

A.4 Proof of Theorem 4: Representation Drift

Proof Sketch. Let $\mathcal{Z}_{\text{lat}} = \mathbb{R}^2$ be a two-dimensional latent space. Define the initial latent mapping $Z_{\text{lat}} = (Z_1, Z_2)^T = f_\psi(X)$ under P_0 such that class labels are linearly separable strictly along Z_1: $P_0(Y = 1 \mid Z_{\text{lat}}) = \mathbb{I}(Z_1 > 0)$, while $Z_2 \sim \mathcal{N}(0, 1)$ serves as an orthogonal uninformative variable. Let the decision head $g_\phi(Z_{\text{lat}}) = \mathbb{I}(\phi_1 Z_1 + \phi_2 Z_2 > 0)$ with state $s_0 \in \mathcal{S}$ and adaptation operator $\omega \in \Omega$ be trained to optimal parameters $\phi = (1, 0)^T$, achieving zero error ($R_H(\omega; s_0, P_0) = 0$).

Induce representation shift in latent space to yield an identical discrepancy signal $z = D\left(P_t(Z_{\text{lat}}), P_{t+1}(Z_{\text{lat}})\right) > 0$ under a distribution distance metric D (e.g., Wasserstein distance or Maximum Mean Discrepancy):

Construction of $P_{t+1}^{(1)}$ (Uninformative Subspace Shift): Translate the uninformative latent component by offset $\delta > 0$ such that $Z_2 \sim \mathcal{N}(\delta, 1)$, while leaving the task-relevant coordinate Z_1 and posterior $P_{t+1}^{(1)}(Y \mid Z_{\text{lat}})$ invariant. Because $\phi_2 = 0$, the decision head g_ϕ remains optimal. Executing Ω_{reset} resets head weights ϕ, forcing sample-inefficient re-optimization under H post-drift observations. Thus, $\Omega^*(P_{t+1}^{(1)}) = \{\Omega_0\}$.

Construction of $P_{t+1}^{(2)}$ (Latent Task Inversion): Construct a translation or reflection along Z_1 such that $Z_1 \sim \mathcal{N}(-\delta, 1)$ while keeping $Z_2 \sim \mathcal{N}(0, 1)$, such that the metric discrepancy matches $D(P_t(Z_{\text{lat}}), P_{t+1}^{(2)}(Z_{\text{lat}})) = z$. The decision boundary defined by $\phi = (1, 0)^T$ now systematically misclassifies instances. Executing Ω_0 causes persistent prediction failure over time horizon H, whereas executing Ω_{reset} re-initializes ϕ, enabling rapid convergence to the inverted decision boundary $\phi^* = (-1, 0)^T$. Thus, $\Omega^*(P_{t+1}^{(2)}) = \{\Omega_{\text{reset}}\}$.

Since $T_{\text{repr}}(P_t(Z_{\text{lat}}), P_{t+1}^{(1)}(Z_{\text{lat}})) = T_{\text{repr}}(P_t(Z_{\text{lat}}), P_{t+1}^{(2)}(Z_{\text{lat}})) = z$, representation drift detection alone is adaptation-incompatible over $\{\Omega_0, \Omega_{\text{reset}}\}$. □

A.5 Proof of Theorem 5: Information Bottleneck

Proof. By construction $\Omega_H^\dagger \longrightarrow W \longrightarrow Z \longrightarrow \widehat{\Omega}$ forms a Markov chain. Applying the data-processing inequality yields:

$$I(\Omega_H^\dagger; \widehat{\Omega}) \leq I(\Omega_H^\dagger; Z) \leq I(W; Z).$$

Since mutual information is bounded by entropy, $I(W; Z) \leq H(Z)$, which proves 5. For the error bound, Fano's inequality gives:

$$H(P_e) + P_e \log(|\Omega| - 1) \geq H(\Omega_H^\dagger \mid Z).$$

Using $H(P_e) \leq 1$ and $H(\Omega_H^\dagger \mid Z) = H(\Omega_H^\dagger) - I(\Omega_H^\dagger; Z)$, together with $I(\Omega_H^\dagger; Z) \leq H(Z)$, gives:

$$1 + P_e \log |\Omega| \geq H(\Omega_H^\dagger) - H(Z),$$

and therefore:

$$P_e \geq \frac{H(\Omega_H^\dagger) - H(Z) - 1}{\log |\Omega|}.$$

□

A.6 Proof of Corollary 6: Complexity-Dependent Information Bottleneck

Proof. Substituting $H(\Omega_H^\dagger) \geq c \log |\Omega|$ into Equation 5 gives:

$$P_e \geq \frac{c \log |\Omega| - H(Z) - 1}{\log |\Omega|} = c - \frac{H(Z) + 1}{\log |\Omega|}.$$

For a binary detector, $H(Z) \leq 1$, giving the stated result. □

B Empirical validation of unactionability theorem

In this section we include the validation of Theorems 2 and 4. Experimental details can be found in section D

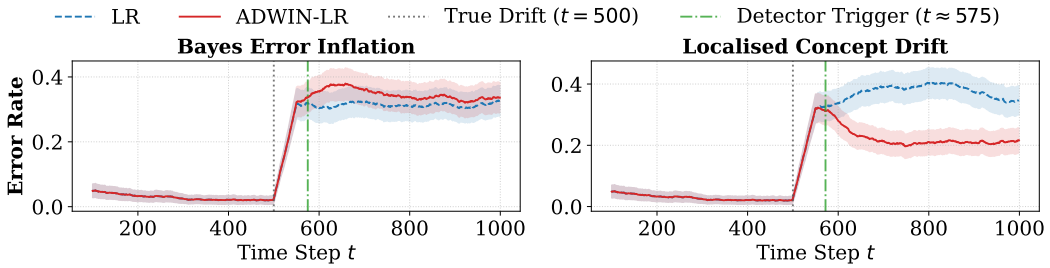

Figure 3: **Unactionability of Change Detection: Posterior Drift.** Average Error Rate of Logistic Regression without a change detector (LR) vs. ADWIN-triggered reset (ADWIN-LR) for posterior drift scenarios: bayes error inflation (left) and localised concept drift (right). Unactionability is established due to ADWIN triggering in both cases (i.e., same drift signal), but having different optimal adaptation policies (no-reset for the left figure, and reset for the right).

B.1 Validation of Theorem 2: Posterior Drift

Consider a data stream where features are sampled uniformly over domain $\mathcal{X} = [0, 1]^2$ with pre-drift (P_0) boundary $Y = \mathbb{I}(X_1 + X_2 > 1)$. Two Logistic Regression models (with differing adaptation policies) are deployed on this stream. The first model (LR) simply trains on incoming samples in an online fashion, the second model (ADWIN-LR) also trains in an online fashion but will reset itself when ADWIN (which is monitoring the loss of the model it is paired with) detects drift. Both models are initially trained on 500 instances sampled from P_0.

At timestep $t = 500$, one of two drift events are induced: (1) bayes error inflation: whereby samples now have a 30% of having their labels flipped, and (2) localised concept drift: whereby all samples$X_1 < \sqrt{z}$ and $X_2 < \sqrt{z}$ have their labels flipped. We then train both models on $H = 500$ samples from the post-drift distribution P_1 recording the error rate.

Fig. 3 reports the prequential error across 200 independent stream replications across both scenarios (100 per scenario). Note that in both post-drift scenarios, ADWIN detects drift at the same timestep ($t \approx 575$) which establishes the identical drift signal condition necessary for determining unactionability. Fig. 3 (left) shows the error for the scenario with bayes error inflation, here we see that LR outperforms ADWIN-LR because the detection signal triggers a reset operation. Fig. 3 (right) shows the error for the scenario with localised concept shift, here we see that ADWIN-LR outperforms LR because the reset operation enables forgetting of the old decision boundary. Given these results, we have established two non-subordinate adaptation operations (i.e., no action is better in the first scenario, while resetting is better in the second). With both identical detector response and the presence of two or more non-subordinate adaptation operators, we demonstrate unactionability.

B.2 Validation of Theorem 4: Representation Drift

Consider a data stream where features are projected into a latent space represented as $Z_1, Z_2 \sim \mathcal{N}(0, 1)$ with pre-drift (P_0) boundary $Y = \mathbb{I}(Z_1 > 0)$. Two Logistic Regression models (with differing adaptation policies) are deployed on this stream using the latent projection as input. The first model (LR) simply trains on incoming samples in an online fashion, the second model (ADWIN-LR) also trains in an online fashion but will reset itself when ADWIN (which is monitoring the L^2 norm of this latent space: $||Z_{\text{lat}}||^2 = \sqrt{(Z_1)^2 + (Z_2)^2}$) detects drift. Both models are initially trained on 500 instances sampled from P_0.

At timestep $t = 500$, one of two drift events are induced: (1) uninformative subspace drift: whereby samples from Z_2 are sampled from $\mathcal{N}(3, 1)$ (2) latent task inversion: whereby samples from Z_2 are sampled from $\mathcal{N}(3, 1)$ and the decision boundary shift to: $P_t(Y = 1 \mid Z_{\text{lat}}) = \mathbb{I}(Z_1 < 3.0)$. We then train both models on $H = 500$ samples from the post-drift distribution P_1 recording the error rate.

Fig. 4 reports the prequential error across 200 independent stream replications across both scenarios (100 per scenario). Note that in both post-drift scenarios, ADWIN detects drift at the same timestep

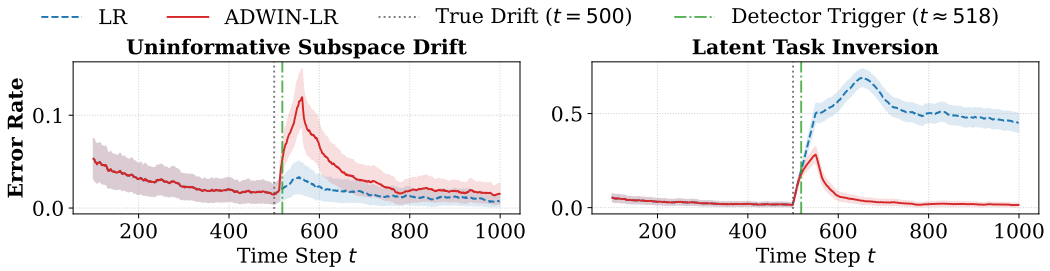

Figure 4: **Unactionability of Change Detection: Representation Drift.** Average Error Rate of Logistic Regression without a change detector (LR) vs. ADWIN-triggered reset (ADWIN-LR) for representation drift scenarios: uninformative subspace drift (left) and latent task inversion (right). Unactionability is established due to ADWIN triggering in both cases (i.e., same drift signal), but having different optimal adaptation policies (no-reset for the left figure, and reset for the right).

($t \approx 518$) which establishes the identical drift signal condition necessary for determining unactionability. Fig. 4 (left) shows the error for the scenario with uninformative subspace drift, here we see that LR outperforms ADWIN-LR because the detection signal triggers a reset operation. Fig. 3 (right) shows the error for the scenario with latent task inversion, here we see that ADWIN-LR outperforms LR because the reset operation enables forgetting of the old decision boundary. Given these results, we have established two non-subordinate adaptation operations (i.e., no action is better in the first scenario, while resetting is better in the second). With both identical detector response and the presence of two or more non-subordinate adaptation operators, we establish unactionability.

C Exploration of Information Bottlenecks

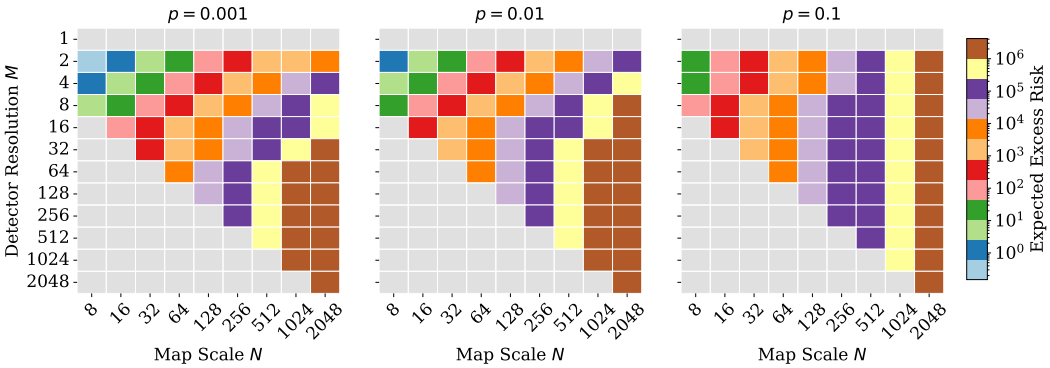

Figure 5: **Information Bottleneck of Coarse-Grained Detectors.** Given a grid of size $N \times N$ containing binary values that change with probability p, and detectors of size $M \times M$ placed on the grid. This Figure show how excess risk (number of stationary cells for which change is detected) increases as the block size of the detector increases.

We use this synthetic setup to illustrate the information bottleneck implied by Theorem 5. As detector granularity decreases, an increasing number of underlying drift variables are compressed into a single binary detector signal, resulting in increasing operational cost from coarse-grained adaptation.

Consider a grid of size $N \times N$ ($N \in \mathbb{Z}$) where each cell in the grid contains a binary value sampled randomly. We then place oracle (perfect) change detectors of size $M \times M$ over the grid ($M \in \mathbb{Z}$ and $0 < M \leq N$). For example, a grid of size $N = 4$ will contain 16 detectors when $M = 1$, 4 detectors with $M = 2$ and 1 detector when $M = 4$. As M increases, the drift detectors monitor a larger (coarser) area of the information space (the grid). This grid-based setup is analogous to having independent change detectors monitor discrete feature subspaces or base learners in an ensemble architecture.

We then induce drift whereby each grid cell will randomly, and independently, change their value with probability p. Given that we aim to show how enforcing coarse-grained change detection (i.e., change detectors that condense large amounts of information into a single bit) induce information bottlenecks. We define the expected excess risk of a grid that undergoes drift as the number of stationary cells for which change is (incorrectly) detected:

$$\mathbb{E}[\mathcal{R}(N, M, p)] = N^2(1-p)(1-(1-p)^{M^2-1}) \tag{4}$$

where $(1-p)$ is the probability that a given cell is stationary and $(1-(1-p)^{M^2-1})$ defines the probability that at least one cell in a detector's monitored subspace drifts.

Using this setup (i.e., initialising a grid, then inducing drift, and monitoring the number of cells that were incorrectly detected as having drift) we can study how information compression via coarse grained detectors leads to increased excess risk. Figure 5 plots the average excess risk achieved over varying grid $N = \{8, 16, \ldots, 2048\}$ and detector $M = \{1, 2, 4, \ldots, 2048\}$ sizes from 100 replicates per grid size and detector resolution pair. We also investigate the incurred excess risk over three values of $p = \{0.001, 0.01, 0.1\}$ representing low, medium and high drift probability respectively. Results show that as detector granularity decreases, the excess risk increases. This is due to the information bottleneck: a coarser detector compresses the multi-bit signal of cells it monitors into a single bit to indicate whether change has occurred (in any of the those cells).

Formally, the drift vector $\mathbf{d} \in \{0,1\}^{M^2}$ which describes how a grid cell changes from its initial state to its post drift state $M \times M$ block possesses a joint entropy of $H(\mathbf{d}) = M^2 H_b(p)$ bits, where $H_b(p)$ is the binary entropy function. Because the detector compresses $\mathbf{d}$ into a single binary signal $Z \in \{0,1\}$, the mutual information is strictly capacity-bounded at $I(\mathbf{d}; Z) \leq H(Z) \leq 1$ bit. This yields an information loss of:

$$H(\mathbf{d} \mid Z) = H(\mathbf{d}) - I(\mathbf{d}; Z) \geq M^2 H_b(p) - 1 \tag{5}$$

By Fano's inequality, this missing $M^2 H_b(p) - 1$ bits of state information imposes an irreducible lower bound, forcing any coarse grained detector to falsely detect on stationary cells.

The exception to this is when $M = 1$ whereby a detector monitors only a single bit signal. This is rarely done and highly infeasible in practice. Unsupervised detectors often monitor the entire feature space or individual features, supervised detectors often monitor error-rates or prediction uncertainties of the models they are deployed alongside. Information capacity reduction is a necessary tradeoff that may incur non-zero excess risk for misadaptation.

Interestingly, the excess risk calculated using Equation 4 exhibits non-monotonic properties as $p \to 1.0$ which we explore in the next section.

C.1 Non-Monotonicity Phenomenon in Information Bottleneck Experiments

In the previous section, we noted that the excess risk (number of false resets) exhibited non-monotonic behaviour as the probability of a drift occurring increases ($p \to 1.0$). This section investigates this phenomena further. As noted in Fig. 5, the maximum excess risk at $p = 0.1$ (3.8×10^6) is lower than at $p = 0.01$ (4.2×10^6) when $N = M = 2048$. This is however not the case when $N = 2048$ and $M = 2$ where the excess risks are: ($p = 0.1$: 1.0×10^6) and ($p = 0.01$: 1.2×10^5). This finding may imply that when drift events are more likely, coarser resolutions of change detection are less likely to be information bottlenecks.

Recall that in the grid experiments, the expected excess risk given a grid of size $N \times N$, detectors of resolution $M \times M$ where drift events in each grid cell happen independently with probability p follows:

$$\mathbb{E}[\mathcal{R}(N, M, p)] = N^2(1-p)(1-(1-p)^{M^2-1}) \tag{6}$$

Taking the derivative of Eq. 6 with respect to p we get:

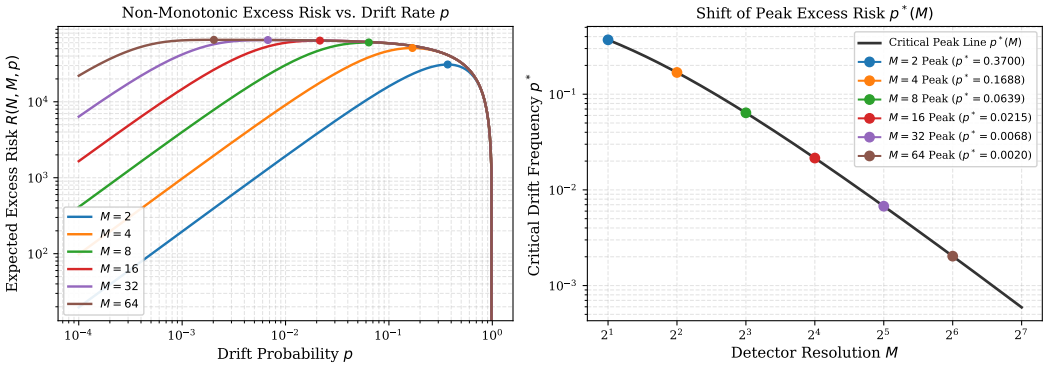

Figure 6: **Non-Monotonic Excess Risk.** Plots of the excess risk of a detector with resolution M as the probability of drift occurring the grid ($N = 256$) $p \to 1.0$. The left figure demonstrates how excess risk exhibits non-monotonicity where each detector resolution achieves some maximum excess risk before tending to zero excess risk as $p \to 1.0$. The right figure shows that the peak of excess risk occurs for larger M occurs at smaller as smaller values of p. These findings reinforce that fine-grained detection should be generally preferred, but it is possible to get away with coarse-grained detection in environments with highly probable drift events.

$$\frac{\partial \mathcal{R}}{\partial p} = N^2[M^2(1-p)^{M^2-1} - 1] \tag{7}$$

To find the probability at which maximum excess risk occurs p^*, we set $\frac{\partial \mathcal{R}}{\partial p} = 0$ from which we derive:

$$p^* = 1 - (M^2)^{-\frac{1}{M^2-1}} \tag{8}$$

Fig. 6 (left) plots Eq. 6 as $p \to 1.0$ for various detectors M on a grid with resolution $N = 256$ and Fig. 6 (right) plots the corresponding p^* values. We observe two findings from these plots: (1) as the probability of a drift event $p \to 1.0$, the expected excess risk $\mathbb{E}[\mathcal{R}(N, M, p)] \to 0.0$. (2) as detector resolution $M \to N$, the probability at which maximum excess risk occurs $p^* \to 0.0$. In the grid experiments, this occurs because despite each drift event in each cell being sampled independently, as $p \to 1.0$, drift events appear correlated (even though they are not). Thus, detecting drift is one cell, more reliably indicates drift in another cell, thus reducing the information bottleneck.

The analysis still clearly shows that increased information capacity reduces excess risk, but may imply that detector resolution itself can be optimised. In data streams (and time series), features are often correlated (and in the frequency in which they drift) and drift patterns may be localised to various feature subspaces. One may be able to construct a monitoring scheme where excess risk is minimised (e.g., if two features are highly correlated, you need only monitor one of those two features or monitor both with a single detector). This may present a fruitful avenue of future research to address the impracticality of measuring all possible signals where drift can occur in real-world streams.

D Experimental Details

This section provides additional experimental details for both the synthetic experiments and USP DS repository case study.

D.1 Unactionability Synthetic Experiments

For the unactionability synthetic experiments, a logistic regression model implemented in `river` (Montiel et al., 2021) with default parameters and a standard scaler is used. The reset-variant adds

an ADWIN (Bifet & Gavalda, 2007) detector that resets the model when drift is detected. We used the default delta for ADWIN in the `river` implementation. All results are generated using a pseudo-random number generator initialised with seed = 42. Furthermore, each stream consists of 1000 samples with a drift occurring at timestep 500. All results are evaluated over 100 replicates (replicate seeds are generated from the initial random seed of 42).

For each drift setting, we implement one of the drift scenarios described in the corollaries (Sect. A) which are described as follows:

Posterior Drift

- **Bayes Error Inflation**: from $t = 500$, z percent of instances have their labels flipped.
- **Localised Concept Drift**: from $t = 500$, samples where $X_1 < \sqrt{z}$ and $X_2 < \sqrt{z}$ have their labels flipped.

In this work, we used $z = 0.3$. In this setting, ADWIN is given an error signal per labelled sample and prediction pair (i.e., 1 if prediction is wrong, 0 otherwise).

Covariate Drift

- **Shift in only** $P(X)$: from $t = 500$, X_1 is sampled from range $\mathcal{U}[0, 0.5]$ instead of $\mathcal{U}[0, 1.0]$.
- **Shift in both** $P(X)$ **and** $P(Y|X)$: from $t = 500$, X_1 is sampled from range $\mathcal{U}[0, 0.5]$ instead of $\mathcal{U}[0, 1.0]$ *and* all samples have their labels flipped.

In this setting, ADWIN is given the feature covariates as input.

Representation Drift

In this setting, we assume each sample is represented by some latent representation $Z_1, Z_2 \sim \mathcal{N}(0, 1)$. ADWIN is given the L^2 norm of this latent representation as input: $||Z_{\text{lat}}||^2 = \sqrt{(Z_1)^2 + (Z_2)^2}$.

- **Uninformative Subspace Drift**: from $t = 500$, Z_2 is sampled from range $\mathcal{N}(3, 1)$.
- **Latent Task Inversion**: from $t = 500$, Z_1 is sampled from range $\mathcal{N}(3, 1)$ *and* all the decision boundary shifts to: $P_t(Y = 1 \mid Z_{\text{lat}}) = \mathbb{I}(Z_1 < 3.0)$.

D.2 Case Study

Table 2 includes a summary of the number features, class, and instances for every data stream used in this work. For detailed descriptions on each data stream, we refer the reader to the accompanying citations. As noted, we acquired these data streams from the USP DS repository[1] (Souza et al., 2020). All streams are stored as CSV files. During preprocessing, we found that the `Asfault`, `Posture` and `SmartMeter` streams were malformed. Further investigation revealed that this was due to a formatting error whereby two samples were stored as a single entry. These malformed entries were simply deleted. Furthermore, the feature 'at3' in the `Luxembourg` stream was revealed to be overly informative (i.e., it perfectly predicts the class label). We did not include it when running experiments.

All code and experiments were run using `Python 3` with the Neural Network Implementations sourced from `scikit-learn` (Pedregosa et al., 2011) and Stream Learning Decision Trees from `river` (Montiel et al., 2021). The ensemble and kNN implementation were sourced from `CapyMOA` (Gomes et al., 2025). For CAND, we used `CapyMOA` to create a wrapper class to access the Java implementation from MOA directly. For parameter selection, we opted to use default parameters for all models as they are informed by decades of research. All experiments were run using 2020 MacBook Pro with 2 GHz Quad-Core Intel i5 and 16GB RAM.

All experiments run in this work are stored as CSV files, and all figures are generated from these files. As noted before, we made use of pseudo-random number generators to ensure reproducibility. In general, a default seed value of 42 was used. For the 10 replicates in the Case Study, we used the

[1] `https://sites.google.com/view/uspdsrepository`

following seeds that were initially themselves randomly generated: [1341, 1967, 2414, 3258, 3485, 3676, 5588, 5616, 7079, 8172].

For evaluation, we recorded the Kappa (Cohen, 1960) score for each method every 50 samples over the last 50 samples (we treat the first 50 samples as a warm-up period and ignore the scores from those instances). For each stream, method and class type (Decision Tree, kNN, etc.) triplet, methods are sorted in descending order using their mean Kappa score. The model at index 0 is assigned an initial rank of 1. We then perform fractional ranking fractional ranking on the rest of the methods with the same class type using the Mann-Whitney U test with Holm correction on the Kappa scores (from all seed runs).This is more robust than simply taking the mean or median values because it considers predictions over the entire stream when ranking each model. If two methods are determined to not be statistically distinct, they are given the same rank. For example, consider three methods of the same model class that achieve Kappa scores 0.5, 0.5, 0.3 for some stream, the ranks for these methods would be: 1.5, 1.5, 3. The rest of the evaluation procedure follows the description in Section 5.

Table 2: The USP DS streams (Souza et al., 2020) evaluated in the work.

Name	Features	Classes	Instances
Asfault (AF) (Souza, 2018)	64	5	8,565
Cancer (CA) (Souza et al., 2021)	2,352	7	9,256
Chess (CH) (Žliobaitė, 2011)	8	3	503
Electricity (EL) (Harries, 1999)	8	2	45,312
Forest Cover Type (FC) (Blackard & Dean, 1999)	54	7	581,012
Gas Sensor Array (GS) (Vergara et al., 2012)	128	6	13,910
Heartbeats (HB) (Ulanova et al., 2016)	280	8	77,904
Insects-Abrupt (IA) (Souza et al., 2020)	33	6	52,848
Insects-Gradual (IG) (Souza et al., 2020)	33	6	24,150
Insects-Incremental (II) (Souza et al., 2020)	33	6	57,018
Insects-Reoccurring (IR) (Souza et al., 2020)	33	6	79,986
Keystroke (KS) (Souza et al., 2015)	10	4	1,600
Luxembourg (LX) (Žliobaitė, 2011)	30	2	1,901
MIRS (MS) (Dau et al., 2018)	3,600	2	4,260
NOAA Weather (NW) (Ditzler & Polikar, 2012)	8	2	18,159
Outdoor Objects (OD) (Losing et al., 2015)	21	40	4000
Ozone (OZ) (Dheeru & Taniskidou, 2017)	72	2	2,534
Posture (PO) (Reis et al., 2016)	3	11	164,859
Powersupply (PS) (Zhu, 2010)	2	24	29,928
Rialto (RT) (Losing et al., 2016)	23	10	82,250
SmartMeter (SM) (Souza et al., 2021)	96	10	22,949
StarLightCurves (SL) (Souza et al., 2021)	1024	3	9,236
UWave (UV) (Souza et al., 2021)	315	8	4,478
Yoga (YG) (Dau et al., 2018)	426	2	3,300

E Additional USP DS Repository Results

Tables 3 and 4 reports the average Cohen's Kappa score for each method investigated in the case study. Fig. 7 shows the comparison between AMF and ARF, DEMS and SRP with and without drift detectors.

Table 3: Cohen's Kappa for each method (column) and data stream (row). ADWIN, KS, D3 and PH are shorthand for the EFDT reset variants investigated. s-kNN is shorthand for SAMkNN.

	Decision Trees							kNNs		
	EFDT	**HAT**	**LAST**	**ADWIN**	**KS**	**PH**	**D3**	**kNN**	**w-kNN**	**s-kNN**
AF	0.59	0.52	0.67	0.56	0.56	0.62	0.31	0.67	0.67	0.67
CA	0.03	0.03	0.09	0.04	0.03	0.03	0.01	0.11	0.10	0.06
CH	0.48	0.36	0.26	0.48	0.48	0.48	0.38	0.49	0.47	0.50
EL	0.53	0.58	0.51	0.62	0.60	0.60	0.62	0.58	0.50	0.59
FC	0.48	0.44	0.55	0.46	0.47	0.48	0.60	0.78	0.77	0.69
GS	0.63	0.35	0.54	0.70	0.65	0.67	0.79	0.87	0.86	0.90
HB	0.70	0.73	0.84	0.83	0.83	0.76	0.82	0.89	0.90	0.88
IA	0.42	0.47	0.39	0.46	0.44	0.53	0.35	0.57	0.60	0.64
IG	0.39	0.39	0.37	0.43	0.44	0.48	0.39	0.61	0.63	0.68
II	0.42	0.44	0.43	0.39	0.38	0.41	0.36	0.49	0.52	0.52
IR	0.36	0.38	0.33	0.48	0.39	0.46	0.35	0.60	0.63	0.65
KS	0.66	0.61	0.59	0.66	0.66	0.66	0.70	0.87	0.88	0.88
LX	0.13	0.13	0.12	0.13	0.13	0.13	0.09	0.21	0.21	0.21
MS	0.43	0.31	0.33	0.35	0.37	0.43	0.35	0.50	0.51	0.51
NW	0.30	0.29	0.27	0.25	0.23	0.25	0.16	0.38	0.37	0.32
OD	0.55	0.57	0.57	0.56	0.61	0.55	0.81	0.77	0.78	0.83
OZ	0.33	0.32	0.33	0.34	0.33	0.33	0.36	0.40	0.41	0.36
PO	0.40	0.35	0.35	0.22	0.15	0.38	0.04	0.46	0.46	0.45
PS	0.13	0.12	0.13	0.12	0.12	0.13	0.08	0.13	0.13	0.15
RT	0.50	0.27	0.51	0.55	0.56	0.53	0.59	0.73	0.71	0.79
SM	0.38	0.45	0.30	0.39	0.37	0.34	0.40	0.47	0.48	0.53
SL	0.71	0.59	0.50	0.70	0.70	0.71	0.66	0.76	0.73	0.73
UW	0.31	0.21	0.34	0.35	0.35	0.31	0.36	0.55	0.55	0.57
YG	0.07	0.06	0.12	0.08	0.14	0.08	0.15	0.65	0.66	0.66

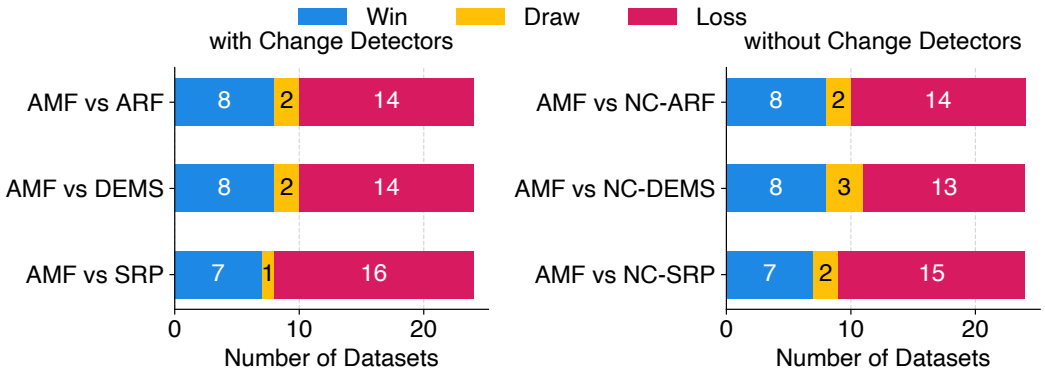

Figure 7: **Win-Loss ratio of AMF vs. other Ensemble Methods**: The left column plots the win-loss ratio of AMF vs ARF, DEMS and SRP when the latter use drift detectors to replace base learners, the right column shows the win-loss ration when ARF, DEMS and SRP have their adaptive policies disabled. Results show that performance gaps between AMF and the other ensemble methods are largely due to architectural differences rather than the presence of a drift detector.

Table 4: Cohen's Kappa for each method (column) and data stream (row). DMS, NARF, NDMS, NSRP are shorthand for DEMS, NC-ARF, NC-DEMS, NC-SRP and HNN, RNN and CND are shorthand for H-ANN, R-ANN and CAND resp.

	Ensembles							**Neural Networks**			
	AMF	**ARF**	**NARF**	**DMS**	**NDMS**	**SRP**	**NSRP**	**ANN**	**HNN**	**RNN**	**CND**
AF	0.67	0.73	0.74	0.74	0.74	0.75	0.74	0.76	0.76	0.73	0.76
CA	0.11	0.02	0.02	0.02	0.02	0.02	0.02	0.27	0.26	0.24	0.20
CH	0.46	0.46	0.46	0.52	0.52	0.52	0.52	0.14	0.14	0.14	0.49
EL	0.62	0.73	0.64	0.68	0.61	0.68	0.62	0.39	0.27	0.03	0.64
FC	0.71	0.80	0.79	0.83	0.77	0.84	0.77	0.64	0.53	0.23	0.60
GS	0.91	0.78	0.83	0.86	0.83	0.86	0.84	0.84	0.83	0.67	0.88
HB	0.89	0.90	0.91	0.91	0.91	0.91	0.91	0.90	0.91	0.90	0.88
IA	0.53	0.65	0.54	0.67	0.54	0.67	0.54	0.65	0.65	0.52	0.67
IG	0.55	0.66	0.55	0.69	0.57	0.69	0.57	0.68	0.67	0.54	0.69
II	0.55	0.57	0.57	0.58	0.57	0.58	0.57	0.58	0.58	0.49	0.58
IR	0.45	0.64	0.46	0.65	0.47	0.65	0.47	0.68	0.68	0.54	0.69
KS	0.91	0.88	0.88	0.90	0.90	0.91	0.91	0.57	0.36	0.26	0.87
LX	0.20	0.26	0.26	0.24	0.24	0.25	0.25	0.28	0.26	0.25	0.30
MS	0.51	0.55	0.57	0.39	0.38	0.58	0.58	0.74	0.74	0.70	0.53
NW	0.34	0.42	0.42	0.39	0.39	0.39	0.40	0.44	0.43	0.16	0.43
OD	0.84	0.64	0.64	0.70	0.68	0.70	0.68	0.35	0.22	0.13	0.54
OZ	0.34	0.36	0.36	0.35	0.35	0.36	0.36	0.41	0.36	0.38	0.35
PO	0.50	0.48	0.49	0.46	0.43	0.45	0.43	0.36	0.28	0.04	0.43
PS	0.08	0.13	0.10	0.14	0.13	0.14	0.13	0.14	0.10	0.07	0.15
RT	0.82	0.71	0.79	0.79	0.79	0.79	0.79	0.51	0.40	0.19	0.59
SM	0.63	0.65	0.64	0.70	0.66	0.70	0.66	0.66	0.64	0.54	0.67
SL	0.78	0.80	0.81	0.82	0.81	0.83	0.81	0.88	0.88	0.86	0.72
UW	0.51	0.50	0.48	0.49	0.49	0.49	0.49	0.56	0.57	0.57	0.52
YG	0.63	0.44	0.44	0.44	0.44	0.45	0.45	0.72	0.69	0.68	0.33