DRIFT-TO-ACTION CONTROLLERS: BUDGETED INTERVENTIONS WITH ONLINE RISK CERTIFICATES

Ismail Lamaakal[1*], **Chaymae Yahyati**[1*], **Khalid El Makkaoui**[1], **Ibrahim Ouahbi**[1], **Yassine Maleh**[2]

[1]Multidisciplinary Faculty of Nador, Mohammed First University, Oujda 60000, Morocco
[2]Laboratory LaSTI, ENSAK, Sultan Moulay Slimane University, Khouribga 54000, Morocco

{ismail.lamaakal, chaymae.yahyati, k.elmakkaoui, i.ouahbi}@ump.ac.ma
y.maleh@usms.ma

ABSTRACT

Deployed machine learning systems face distribution drift, yet most monitoring pipelines stop at alarms and leave the response underspecified under labeling, compute, and latency constraints. We introduce *Drift2Act*, a drift-to-action controller that treats monitoring as constrained decision-making with explicit safety. *Drift2Act* combines a sensing layer that maps unlabeled monitoring signals to a belief over drift types with an *active risk certificate* that queries a small set of delayed labels from a recent window to produce an anytime-valid upper bound $U_t(\delta)$ on current risk. The certificate gates operation: if $U_t(\delta) \leq \tau$, the controller selects low-cost actions (e.g., recalibration or test-time adaptation); if $U_t(\delta) > \tau$, it activates abstain/handoff and escalates to rollback or retraining under cooldowns. In a realistic streaming protocol with label delay and explicit intervention costs, *Drift2Act* achieves near-zero safety violations and fast recovery at moderate cost on WILDS Camelyon17, DomainNet, and a controlled synthetic drift stream, outperforming alarm-only monitoring, adapt-always adaptation, schedule-based retraining, selective prediction alone, and an ablation without certification. Overall, online risk certification enables reliable drift response and reframes monitoring as decision-making with safety.

1 INTRODUCTION

Modern machine learning models are increasingly deployed as long-lived services, where inputs evolve due to changing sensors, users, policies, environments, and feedback loops (Modi et al., 2017; Polyzotis et al., 2019; Perdomo et al., 2020). In such settings, distribution drift is the rule rather than the exception: a system that performs well at launch may degrade weeks later, and the degradation may be global (covariate shift), semantic (concept drift), or localized (subgroup-specific drift) (Quiñonero-Candela et al., 2008; Shimodaira, 2000; Widmer & Kubat, 1996; Tsymbal, 2004; Raji et al., 2020). A large literature addresses *catching* drift through statistical tests and representation monitoring (Gretton et al., 2012; Rabanser et al., 2019), and a growing literature studies *adapting* models via test-time adaptation and continual learning (Wang et al., 2021; Kirkpatrick et al., 2017). However, production reliability requires a third ingredient: once evidence of drift appears, the system must decide *what to do next* under real operational constraints (Modi et al., 2017; Polyzotis et al., 2019).

Two gaps persist in current practice. First, drift detection often ends at an alarm (Rabanser et al., 2019). Alarms are not actions: they do not specify whether to recalibrate, adapt, request labels, retrain, or roll back, nor do they account for budgets, latency, cooldowns, and governance constraints (Modi et al., 2017; Polyzotis et al., 2019; Raji et al., 2020). Second, adaptation is frequently applied without verified safety (Wang et al., 2021). Under delayed supervision, a system may silently operate in an unsafe regime, or may overreact and incur unnecessary cost. These gaps are amplified at scale, where heterogeneous streams and limited labeling capacity make it infeasible to continuously validate performance (Modi et al., 2017; Polyzotis et al., 2019).

[1]* Equally contributed and led the project.

We propose a unified approach that treats **monitoring as decision-making with safety**. Our method introduces an *active risk certificate* that produces an anytime-valid upper bound $U_t(\delta)$ on the current windowed risk R_t using a small number of labels sampled from the live stream. The certificate functions as a safety layer: if $U_t(\delta) \leq \tau$ the system can operate normally and apply inexpensive corrections, while if $U_t(\delta) > \tau$ the system triggers an explicit fallback (abstain/handoff) and escalates to stronger interventions such as rollback or retraining. To translate drift evidence into effective actions, we additionally maintain a belief state $b_t(d) = \mathbb{P}(D_t = d \mid z_{1:t})$ over drift types $d \in \{\text{none}, \text{covariate}, \text{concept}, \text{subgroup}\}$ derived from representation, uncertainty, and calibration monitors. The controller then selects actions by maximizing a belief-weighted utility under budget and cooldown constraints.

Our evaluation uses a realistic streaming protocol with delayed labels, explicit intervention costs, and heavy-action cooldowns. Across WILDS Camelyon17, DomainNet, and a controlled synthetic drift stream, the proposed controller achieves near-zero safety violations at moderate cost, dominating the safety–cost frontier in Figure 2a and recovering rapidly after drift events in Figure 2b. Table 1 summarizes improvements in safety, recovery time, and worst-group robustness relative to alarm-only monitoring, adapt-always baselines, and schedule-driven retraining.

Our contributions are threefold: We introduce an active, anytime-valid risk certificate for drift monitoring under delayed supervision, we develop a belief-driven controller that maps drift evidence to cost-aware interventions under operational constraints, and we propose a streaming evaluation protocol that jointly measures safety, recovery, and operational cost.

2 Related Work

Drift detection and monitoring. Classical drift detection is closely related to sequential change-point testing, including CUSUM-style procedures and Page–Hinkley-type tests (Page, 1954; Hinkley, 1970), and stream-specific detectors that adapt windows or monitor error-rate statistics online (Bifet & Gavaldà, 2007; Gama et al., 2004; 2014). Modern drift monitoring often uses representation-based two-sample testing, where changes are detected in an embedding space rather than raw inputs, including kernel Maximum Mean Discrepancy (MMD) tests (Gretton et al., 2012) and learned test statistics (Kirchler et al., 2020). Another widely used paradigm is *classifier two-sample testing*, which trains a discriminator to separate reference vs. recent samples and uses its accuracy/AUC as a shift signal (Lopez-Paz & Oquab, 2016). Empirical studies highlight that dataset-shift detectors can fail silently or over-trigger depending on the shift type and monitoring choice (Rabanser et al., 2019). Related work also studies specific shift structures such as label shift, with black-box estimation and detection procedures (Lipton et al., 2018). In our paper, these monitoring signals are not endpoints; they are inputs to a belief state that drives interventions.

Adaptation under distribution shift. A large literature updates models online under shift using unlabeled or sparsely labeled deployment data, including test-time adaptation (TTA) and self-supervised test-time training (Wang et al., 2021; Sun et al., 2020). These methods can be effective for covariate and representation drift, but may be unstable under semantic changes or feedback loops, motivating safety-aware gating. Continual learning methods address non-stationarity by preventing catastrophic forgetting, using regularization, replay, or exemplar-based updates (Kirkpatrick et al., 2017; Lopez-Paz & Ranzato, 2017; Rebuffi et al., 2017; Li & Hoiem, 2016). Our controller treats adaptation as one action among several (including label acquisition, rollback, and retraining) and uses online risk certification to avoid prolonged unverified operation.

Selective prediction and uncertainty. Selective prediction (abstention) reduces harm by deferring uncertain examples, with theoretical and empirical work on risk–coverage trade-offs and learned selection modules (Geifman & El-Yaniv, 2017; 2019). Uncertainty estimation tools that commonly support abstention and monitoring include deep ensembles and approximate Bayesian inference via dropout (Lakshminarayanan et al., 2017; Gal & Ghahramani, 2016), as well as confidence and calibration diagnostics (Guo et al., 2017). We include abstain/handoff as a safe fallback, but trigger it using a *certified* upper bound on current risk rather than confidence alone.

Conformal inference and risk control. Conformal prediction provides distribution-free predictive inference under exchangeability (Vovk et al., 2005), and recent work generalizes conformal ideas to control broader risk functionals beyond coverage (Angelopoulos et al., 2022). In parallel, confidence

sequences provide anytime-valid bounds for streaming means under bounded observations and optional stopping (Kirchler et al., 2020). Our active risk certificate builds on these anytime-valid ideas to upper-bound windowed deployment risk using a small, randomly sampled labeled subset, and then integrates the bound directly into the control loop.

ML systems, governance, and monitoring at scale. Production ML monitoring emphasizes logging, alerting, and remediation under operational constraints, while documenting and auditing model behavior via standardized reporting artifacts (Sculley et al., 2015; Breck et al., 2017; Mitchell et al., 2019; Gebru et al., 2021). Benchmarks for distribution shift help evaluate monitoring and adaptation realistically, including WILDS (Koh et al., 2021) and DomainNet streams commonly instantiated via multi-source domain adaptation benchmarks. Our framework aligns with this systems view by producing auditable certification signals and explicit actions with costs/cooldowns, evaluated under a streaming protocol that reports safety violations and worst-group robustness.

In summary, prior work addresses detection, adaptation, or abstention in isolation. We unify these components by framing drift monitoring as a constrained decision problem with an anytime-valid safety certificate that gates action selection.

3 Sensing Layer: Monitoring Signals and Belief over Drift Type

The sensing layer summarizes recent deployment behavior into a compact evidence vector that can be computed with little or no immediate supervision. This evidence is then mapped into a belief over drift types, enabling downstream actions that depend on whether the system is facing covariate shift, concept shift, or localized subgroup drift.

3.1 Monitoring windows and representations

At each time step t, the system maintains a recent input window $\mathcal{B}_t = \{x_{t-n+1}, \ldots, x_t\}$ and a reference window $\mathcal{B}_{\text{ref}}$ that represents nominal conditions. We monitor drift in a representation space induced by the deployed model through an embedding function $g_\theta : \mathcal{X} \to \mathbb{R}^p$:

$$r \;=\; g_\theta(x). \tag{1}$$

In this equation, $x \in \mathcal{X}$ is an input, θ are the current model parameters, $g_\theta(\cdot)$ is the feature extractor, and $r \in \mathbb{R}^p$ is the embedding used by the monitors. We denote the sets of embeddings from the recent and reference windows by

$$\mathcal{R}_t \;=\; \{g_\theta(x) : x \in \mathcal{B}_t\}, \qquad \mathcal{R}_{\text{ref}} \;=\; \{g_\theta(x) : x \in \mathcal{B}_{\text{ref}}\}. \tag{2}$$

Here $\mathcal{R}_t$ and $\mathcal{R}_{\text{ref}}$ are the monitored embedding collections for the recent and reference windows, respectively.

3.2 Evidence signals

We compute complementary drift signals that capture representation shift, uncertainty drift, and calibration degradation. The first signal is a kernel two-sample statistic (MMD) computed between $\mathcal{R}_t$ and $\mathcal{R}_{\text{ref}}$:

$$\text{MMD}_t^2 \;=\; \left\| \frac{1}{|\mathcal{R}_t|} \sum_{r \in \mathcal{R}_t} \phi(r) \;-\; \frac{1}{|\mathcal{R}_{\text{ref}}|} \sum_{r' \in \mathcal{R}_{\text{ref}}} \phi(r') \right\|^2. \tag{3}$$

In this equation, $\phi(\cdot)$ is the feature map of a positive definite kernel, $|\mathcal{R}_t|$ and $|\mathcal{R}_{\text{ref}}|$ are window sizes, and MMD_t^2 increases when the embedding distribution departs from nominal conditions.

The second signal is an uncertainty drift score based on predictive entropy. The model produces class probabilities $p_\theta(y \mid x)$, and we define the entropy for an input x as

$$H_\theta(x) \;=\; -\sum_{y \in \mathcal{Y}} p_\theta(y \mid x)\, \log p_\theta(y \mid x). \tag{4}$$

Here $\mathcal{Y}$ is the label set and $H_\theta(x)$ is higher when the model is less confident. We compare mean entropy in the recent and reference windows:

$$\Delta H_t \;=\; \frac{1}{|\mathcal{B}_t|} \sum_{x \in \mathcal{B}_t} H_\theta(x) \;-\; \frac{1}{|\mathcal{B}_{\text{ref}}|} \sum_{x \in \mathcal{B}_{\text{ref}}} H_\theta(x). \tag{5}$$

In this equation, ΔH_t captures shifts in uncertainty that often correlate with out-of-distribution inputs or changes in class-conditional structure.

When delayed labels are available, we also monitor calibration drift using a streaming ECE proxy computed on the labeled set available at time t, denoted $\mathcal{S}_t$ (defined in Appendix A). Let $q_\theta(x) = \max_y p_\theta(y \mid x)$ be the model confidence and let $\{I_b\}_{b=1}^B$ be confidence bins. The ECE estimate is

$$\mathrm{ECE}_t \;=\; \sum_{b=1}^{B} \frac{|\mathcal{S}_{t,b}|}{|\mathcal{S}_t|} \big|\mathrm{acc}_b(t) - \mathrm{conf}_b(t)\big|. \tag{6}$$

In this equation, $\mathcal{S}_{t,b} = \{(x,y) \in \mathcal{S}_t : q_\theta(x) \in I_b\}$ is the labeled subset in bin b, $\mathrm{acc}_b(t)$ is the empirical accuracy in that bin, and $\mathrm{conf}_b(t)$ is the empirical mean confidence in that bin. We use $\Delta\mathrm{ECE}_t = \mathrm{ECE}_t - \mathrm{ECE}_{\mathrm{ref}}$ as a drift signal, where $\mathrm{ECE}_{\mathrm{ref}}$ is computed under nominal conditions.

We concatenate these monitors into a single evidence vector

$$z_t \;=\; \big[\, \mathrm{MMD}_t^2,\; \Delta H_t,\; \Delta\mathrm{ECE}_t \,\big]^\top \;\in\; \mathbb{R}^m. \tag{7}$$

In this equation, m is the number of active monitors.

3.3 Belief over drift type

We model drift type as a latent discrete state D_t taking values in

$$\mathcal{D} \;=\; \{\mathrm{none},\; \mathrm{covariate},\; \mathrm{concept},\; \mathrm{subgroup}\}. \tag{8}$$

The belief state is the posterior probability of each drift type given evidence up to time t:

$$b_t(d) \;=\; \mathbb{P}(D_t = d \mid z_{1:t}), \qquad d \in \mathcal{D}. \tag{9}$$

In this equation, $z_{1:t} = \{z_1, \ldots, z_t\}$ is the evidence history and $b_t(d)$ is the probability assigned to type d.

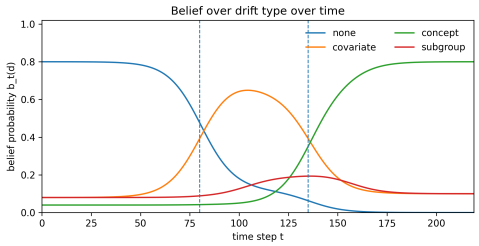


Figure 1: **Belief evolution over drift types.** Posterior $b_t(d) = \mathbb{P}(D_t = d \mid z_{1:t})$ over $\{\mathrm{none}, \mathrm{covariate}, \mathrm{concept}, \mathrm{subgroup}\}$ across time; dashed lines mark drift events that shift belief and guide intervention choice.

We compute $b_t(d)$ using a lightweight Markov belief update with a transition model and an evidence likelihood. Let $T_{d'd} = \mathbb{P}(D_t = d \mid D_{t-1} = d')$ be the transition matrix and let $p(z_t \mid D_t = d)$ be the likelihood of observing evidence z_t under drift type d. The update is

$$\tilde{b}_t(d) \;=\; \sum_{d' \in \mathcal{D}} T_{d'd}\, b_{t-1}(d'), \qquad b_t(d) \;=\; \frac{p(z_t \mid D_t = d)\, \tilde{b}_t(d)}{\sum_{d'' \in \mathcal{D}} p(z_t \mid D_t = d'')\, \tilde{b}_t(d'')}. \tag{10}$$

In these equations, $\tilde{b}_t(d)$ is the predicted belief before incorporating the new evidence, the numerator multiplies the predicted belief by the likelihood of the observed evidence under type d, and the denominator normalizes the distribution so that $\sum_{d \in \mathcal{D}} b_t(d) = 1$. We fit $p(z_t \mid D_t = d)$ on synthetic drift episodes constructed from the base dataset, enabling drift-type inference directly from monitoring signals.

4 Active Risk Certificate: Online Safety Layer

This section introduces an active risk certificate that provides an anytime-valid upper bound on current deployment risk using only a small number of labels sampled from the live stream. The certificate acts as a safety layer that gates operational decisions: when the bound indicates potential violation of a target risk threshold, the system triggers a safe fallback and escalates interventions (see Appendix B for more details).

4.1 WINDOWED RISK UNDER DELAYED SUPERVISION

At time t, we define a recent evaluation window that represents the current deployment regime:

$$W_t \;=\; \{t-N+1,\ldots,t\}, \tag{11}$$

where N is the window length and W_t indexes the most recent N time steps. The (windowed) deployment risk is the mean loss over examples in W_t:

$$R_t \;=\; \frac{1}{N}\sum_{i\in W_t}\ell_i, \qquad \ell_i \;=\; \ell\big(\hat{y}_\theta(x_i),y_i\big). \tag{12}$$

In these equations, $\ell_i \in [0,1]$ is the per-example loss at index i and $\ell(\cdot,\cdot)$ is a bounded task loss (for classification, $\ell_i = \mathbb{I}(\hat{y}_\theta(x_i)\neq y_i)$). The quantity R_t captures the current error/risk level in the most recent regime. Because labels may be delayed, the system cannot evaluate all ℓ_i in W_t; it instead adaptively requests labels for a subset of indices.

4.2 ACTIVE LABEL SAMPLING WITHIN THE WINDOW

Let $Q_t \subseteq W_t$ denote the set of indices in the current window for which labels are requested and observed by time t (or by a short fixed evaluation delay). The controller determines the sample size $|Q_t| = n_t$ based on drift evidence and budget constraints. For each queried index $i \in Q_t$, the system obtains y_i and computes the realized loss ℓ_i. The empirical risk on queried points is

$$\widehat{R}_t \;=\; \frac{1}{n_t}\sum_{i\in Q_t}\ell_i. \tag{13}$$

In this equation, $\widehat{R}_t$ is a sample mean of losses computed from the queried subset, n_t is the number of queried labels, and Q_t is selected by uniformly sampling indices from W_t (or by uniform sampling within each slice, when slice guarantees are desired). Uniform sampling is the key condition that allows finite-sample validity of the certificate with minimal assumptions about drift.

4.3 ANYTIME-VALID UPPER CONFIDENCE BOUND

We construct an anytime-valid upper bound $U_t(\delta)$ on the true window risk R_t from the queried losses. Let $\delta \in (0,1)$ be a target failure probability. We define a nonnegative confidence radius $\mathrm{rad}(n_t,\delta)$ such that

$$\mathbb{P}\Big(\forall t\in\{1,\ldots,T\}:\; R_t \le \widehat{R}_t + \mathrm{rad}(n_t,\delta)\Big) \;\ge\; 1-\delta, \tag{14}$$

and set the certificate as

$$U_t(\delta) \;=\; \widehat{R}_t + \mathrm{rad}(n_t,\delta). \tag{15}$$

In these equations, $\widehat{R}_t$ is the empirical risk on the queried subset, $\mathrm{rad}(n_t,\delta)$ is an anytime-valid uncertainty term that shrinks as more labels are queried, and $U_t(\delta)$ is the certified upper bound on current risk. We instantiate $\mathrm{rad}(\cdot,\cdot)$ using a confidence-sequence construction for bounded random variables, ensuring validity under optional stopping and adaptive query schedules. The explicit form and proof are provided in Appendix D.

4.4 SAFETY GATING AND ESCALATION

The certificate gates operation with respect to a target risk threshold $\tau \in (0,1)$:

$$\text{Safe operation at time } t \iff U_t(\delta) \le \tau. \tag{16}$$

In this decision rule, $U_t(\delta) \le \tau$ certifies that current risk is below the target threshold, enabling normal operation and inexpensive interventions (e.g., recalibration). When $U_t(\delta) > \tau$, the system treats the regime as potentially unsafe and triggers a fallback policy (e.g., abstain/handoff) while escalating to stronger actions such as label acquisition, rollback, or retraining:

$$U_t(\delta) > \tau \;\Rightarrow\; \text{activate fallback and schedule escalation.} \tag{17}$$

This gating mechanism separates *monitoring evidence* (which suggests risk may be rising) from *certified safety* (which bounds the risk using labeled verification).

4.5 LABEL EFFICIENCY THROUGH TARGETED VERIFICATION

The controller uses drift evidence z_t and the drift-type belief b_t (Section 3) to decide when to query labels for certification. Specifically, it increases n_t when monitors indicate rising drift or when the certificate uncertainty $\mathrm{rad}(n_t, \delta)$ is large relative to the safety margin $\tau - \widehat{R}_t$, and it decreases n_t when evidence is stable and the certificate is tight. This creates an explicit coupling between "Catch" and "Operate": labels are treated as a limited operational resource that is spent primarily when needed to maintain certified safety.

4.6 SAFETY GUARANTEE

Theorem 1 (Certified safe operation under random window sampling). Assume that at each time t the queried set Q_t is obtained by uniform random sampling from the window W_t (or by uniform sampling within pre-defined strata) and that the loss satisfies $\ell_i \in [0, 1]$. Then, for any $\delta \in (0, 1)$, the certificate in Eq. (18) satisfies

$$\mathbb{P}(\forall t \in \{1, \ldots, T\} : \; R_t \leq U_t(\delta)) \;\geq\; 1 - \delta. \tag{18}$$

Consequently, with probability at least $1 - \delta$, the system never remains in an unsafe regime without the certificate indicating a violation: whenever $R_t > \tau$, it holds that $U_t(\delta) > \tau$, and the gating rule activates fallback and escalation. A proof and the concrete confidence-sequence construction are given in Appendix D.

5 DRIFT-TO-ACTION CONTROLLER

This section converts drift evidence and the active risk certificate into concrete operational interventions. The controller is designed to be deployable: it selects from a small action set, respects budgets and cooldowns, and escalates only when safety is threatened.

5.1 DECISION VARIABLES AND OBJECTIVE

At each time step t, the controller observes the current evidence vector z_t (Section 3), the belief state $b_t(\cdot)$ over drift types, and the current safety certificate $U_t(\delta)$ (Section 4). It then selects an action $a_t \in \mathcal{A}$ (Table 2) and, when applicable, a label request size k_t for the label-query action.

We express the controller goal as minimizing expected cumulative task loss plus operational cost, while maintaining certified safety whenever the system is allowed to predict:

$$\min_{\pi} \; \mathbb{E}\left[\sum_{t=1}^{T} \ell_t \;+\; \lambda\, c(a_t)\right] \qquad \text{s.t.} \qquad U_t(\delta) \leq \tau \;\text{ or fallback is engaged.} \tag{19}$$

In this optimization, π denotes a policy mapping the controller's observations to actions, $\ell_t = \ell(\hat{y}_\theta(x_t), y_t)$ is the per-step task loss, $c(a_t)$ is the per-step operational cost from Section A.5, and $\lambda \geq 0$ trades off predictive performance against operational burden. The constraint states that the system must either be certified safe, meaning $U_t(\delta) \leq \tau$, or operate under an explicit fallback mode (abstain/handoff) that prevents unverified predictions.

5.2 OPERATIONAL CONSTRAINTS: BUDGETS AND COOLDOWNS

Real systems cannot retrain or roll back arbitrarily often. We model two classes of constraints. First, a labeling budget limits the total number of requested labels:

$$\sum_{t=1}^{T} k_t \;\leq\; B_{\mathrm{lab}}, \tag{20}$$

where k_t is the number of labels requested at time t and B_{lab} is a fixed labeling budget. Second, heavy interventions have cooldowns. Let Δ_{rt} and Δ_{rb} be cooldown durations for retraining and rollback. If $t^{\mathrm{rt}}_{\mathrm{last}}$ is the most recent time a retrain was triggered, the retrain feasibility constraint is

$$a_t = A_4 \;\Rightarrow\; t - t^{\mathrm{rt}}_{\mathrm{last}} \geq \Delta_{\mathrm{rt}}, \tag{21}$$

and analogously for rollback with A_5 and Δ_{rb}. In these constraints, Δ_{rt} and Δ_{rb} capture engineering and infrastructure limitations such as retraining time, deployment approvals, and rollback validation.

5.3 RECEDING-HORIZON UTILITY UNDER SAFETY GATING

Rather than solving a full sequential decision problem, we use a receding-horizon utility that is simple, robust, and practical to implement. For each candidate action $a \in \mathcal{A}$, we compute a one-step utility that combines predicted risk reduction and operational cost:

$$\mathcal{U}_t(a) \;=\; \Delta_t(a) \;-\; \lambda\, c(a) \;-\; \gamma\, \mathrm{Viol}_t(a). \tag{22}$$

In this equation, $\Delta_t(a)$ is the predicted improvement from taking action a (larger is better), $c(a)$ is the operational cost, and $\mathrm{Viol}_t(a)$ is a penalty that discourages actions leading to uncertified operation. The weight $\gamma > 0$ enforces safety preferences. The controller selects

$$a_t \;=\; \arg\max_{a \in \mathcal{A}_t^{\mathrm{feas}}} \; \mathcal{U}_t(a), \tag{23}$$

where $\mathcal{A}_t^{\mathrm{feas}} \subseteq \mathcal{A}$ is the set of feasible actions at time t after enforcing budget and cooldown constraints.

We define $\Delta_t(a)$ using the drift-type belief $b_t(d)$ to encode that different drifts benefit from different interventions:

$$\Delta_t(a) \;=\; \sum_{d \in \mathcal{D}} b_t(d)\, G(d, a), \tag{24}$$

where $G(d, a)$ is a gain table specifying the expected benefit of action a under drift type d. In this definition, $\mathcal{D} = \{\mathrm{none}, \mathrm{covariate}, \mathrm{concept}, \mathrm{subgroup}\}$ is the drift-type set, $b_t(d)$ is the posterior probability of drift type d, and $G(d, a)$ can be learned from historical rollouts or calibrated via synthetic drift simulations. This design yields a controller that is interpretable and easily extensible: gains can be updated without changing the certificate.

5.4 SAFETY-FIRST ESCALATION LOGIC

The certificate $U_t(\delta)$ gates the action space. When the certificate indicates potential violation, the controller prioritizes immediate safety via fallback and then schedules stronger interventions:

$$U_t(\delta) > \tau \;\Rightarrow\; a_t = A_6 \;\text{ and }\; a_{t'} \in \{A_5, A_4\} \text{ is scheduled when feasible.} \tag{25}$$

In this rule, A_6 is abstain/handoff, which prevents unverified predictions, and $\{A_5, A_4\}$ represent rollback and retraining, which restore or improve performance. When the certificate is safe, the controller selects low-cost corrective actions that match the dominant drift type in b_t:

$$U_t(\delta) \le \tau \;\Rightarrow\; a_t \in \{A_0, A_1, A_2, A_3\} \text{ chosen by Eq. (25).} \tag{26}$$

This separation ensures that safety violations trigger immediate protective behavior, while normal operation remains cost-efficient.

6 BENCHMARKS, BASELINES, AND MAIN RESULTS

We evaluate on two real drift benchmarks and one controlled synthetic drift stream. The first benchmark is **WILDS Camelyon17** (Koh et al., 2021; Bándi et al., 2019), where environments correspond to hospitals and distribution shift arises from site-specific acquisition artifacts. The stream orders hospitals to induce multiple drift transitions, including a return to a previously seen hospital to test whether controllers avoid irreversible overreaction. The second benchmark is **DomainNet** (Peng et al., 2019), which exhibits large domain gaps across sketch, clipart, painting, real, and quickdraw; the stream orders domains to induce heavy representation and covariate drift. The synthetic benchmark, **SyntheticDrift-CIFAR**, composes sudden covariate drift (corruption severity) (Hendrycks & Dietterich, 2019), gradual concept drift (class-conditional remapping ramped by α_t) (Gama et al., 2014; Tsymbal, 2004), and subgroup drift (a minority slice receives a style transform) (Gebru et al., 2021; Sagawa et al., 2020), enabling controlled tests of safety and recovery.

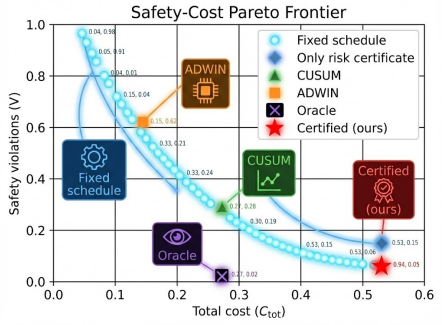
All methods use the same monitoring signals and belief model, and differ only in how they act. The **Alarm-only** baseline raises alarms when evidence exceeds a threshold but performs no interventions (Rabanser et al., 2019). The **Adapt-always** baseline performs test-time adaptation continuously (Wang et al., 2021). The **Retrain-on-schedule** baseline retrains periodically and ignores drift evidence, reflecting common production heuristics (Modi et al., 2017). The **Selective prediction**

Table 1: **Main streaming results under operational constraints.** We report total cost C_{tot}, safety violations V, detection delay T_{det}, recovery time T_{rec}, and minimum worst-group accuracy $\min \text{Acc}^{\text{wg}}$ across Camelyon17, DomainNet, and SyntheticDrift-CIFAR using $\tau = 0.20$, $\delta = 0.05$, and label delay $d = 50$.

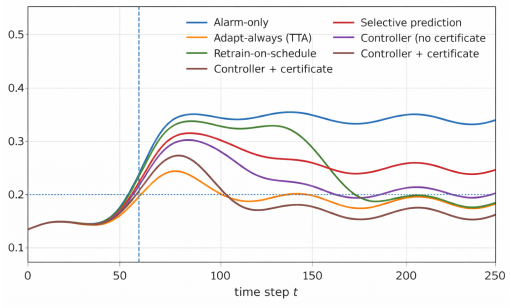
Method	C_{tot}	V	T_{det}	T_{rec}	$\min \text{Acc}^{\text{wg}}$
Camelyon17 stream					
Alarm-only	3.2	46	22	210	0.71
Adapt-always (TTA)	58.0	3	18	74	0.79
Retrain-on-schedule	41.5	9	35	96	0.77
Selective prediction only	9.8	14	24	160	0.74
Controller (no certificate)	18.7	7	19	98	0.78
Controller + certificate (ours)	24.6	0	20	62	0.81
DomainNet stream					
Alarm-only	4.1	63	28	260	0.49
Adapt-always (TTA)	64.0	6	21	110	0.58
Retrain-on-schedule	45.0	14	40	132	0.56
Selective prediction only	12.4	27	30	205	0.52
Controller (no certificate)	22.5	11	23	141	0.57
Controller + certificate (ours)	29.8	1	24	88	0.61
SyntheticDrift-CIFAR stream					
Alarm-only	2.6	52	16	185	0.68
Adapt-always (TTA)	52.0	4	13	66	0.79
Retrain-on-schedule	36.0	10	25	92	0.77
Selective prediction only	8.6	19	18	148	0.72
Controller (no certificate)	16.8	8	14	84	0.78
Controller + certificate (ours)	21.2	0	15	58	0.82

only baseline abstains based on uncertainty but does not adapt or retrain (Geifman & El-Yaniv, 2017; Lakshminarayanan et al., 2017). The **Controller (no certificate)** ablation selects actions using belief-weighted gains but does not gate operation using the certified bound $U_t(\delta)$. The proposed **Controller + certificate** uses full safety gating and active risk verification (Kirchler et al., 2020).

Table 1 summarizes the primary metrics used throughout the paper: total operational cost C_{tot}, safety violations V, detection delay T_{det}, recovery time T_{rec}, and minimum worst-group accuracy over drift windows. Figure 2a visualizes the safety–cost trade-off by plotting V against C_{tot} for each method, showing that the certified controller occupies the low-violation region at moderate cost. Figure 2b illustrates the temporal recovery dynamics around a drift onset, including intervention markers that indicate when major actions are executed (see Appendix F for more results).



(a) **Safety–cost Pareto frontier.** Methods on the Camelyon17 stream plotted by total cost C_{tot} (x-axis) and safety violations V (y-axis); the certified controller attains low violations at moderate cost.



(b) **Recovery curves around a drift event.** Deployed risk $R_t^\star$ over time with drift onset (vertical dashed line) and safety threshold τ (horizontal dotted line); tick marks indicate interventions taken by the certified controller.

Figure 2: **Certified controller performance under cost–safety trade-offs and drift.** (Left) Safety–cost Pareto frontier on Camelyon17. (Right) Recovery dynamics around a drift event relative to threshold τ, with intervention times marked.

7 Conclusion

We reframed drift monitoring as decision-making under operational constraints. Our drift-to-action controller combines drift note: a belief over drift types from unlabeled monitors, an active risk certificate that provides an anytime-valid upper bound on current risk from a small set of randomly sampled delayed labels, and a cost-aware policy that selects interventions under budgets and cooldowns. In realistic streaming evaluations with label delay, this reduces safety violations and speeds recovery at moderate cost compared to alarm-only monitoring, always-on adaptation, and schedule-based retraining.

REFERENCES

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control. *arXiv preprint*, 2022. URL https://arxiv.org/abs/2208.02814.

Albert Bifet and Ricard Gavaldà. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*, pp. 443–448, 2007. doi: 10.1137/1.9781611972771.42. URL https://doi.org/10.1137/1.9781611972771.42.

Eric Breck, Shanqing Cai, Eric Nielsen, Michael Salib, and D. Sculley. The ml test score: A rubric for ml production readiness and technical debt reduction. In *2017 IEEE International Conference on Big Data (Big Data)*, pp. 1123–1132, 2017. doi: 10.1109/BigData.2017.8258038. URL https://doi.org/10.1109/BigData.2017.8258038.

Péter Bándi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, Quanzheng Li, Farhad Ghazvinian Zanjani, Svitlana Zinger, Keisuke Fukuta, Daisuke Komura, Vlado Ovtcharov, Shenghua Cheng, Shaoqun Zeng, Jeppe Thagaard, Anders B. Dahl, Huangjing Lin, Hao Chen, Ludwig Jacobsson, Martin Hedlund, Melih Çetin, Eren Halıcı, Hunter Jackson, Richard Chen, Fabian Both, Jörg Franke, Heidi Küsters-Vandevelde, Willem Vreuls, Peter Bult, Bram van Ginneken, Jeroen van der Laak, and Geert Litjens. From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 38(2):550–560, 2019. doi: 10.1109/TMI.2018.2867350.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016. URL https://arxiv.org/abs/1506.02142.

João Gama, Pedro Medas, Gladys Castillo, and Pedro Pereira Rodrigues. Learning with drift detection. In *Proceedings of the 17th Brazilian Symposium on Artificial Intelligence (SBIA)*, pp. 286–295, 2004. doi: 10.1007/978-3-540-28645-5_29. URL https://doi.org/10.1007/978-3-540-28645-5_29.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):44:1–44:37, 2014. doi: 10.1145/2523813. URL https://doi.org/10.1145/2523813.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021. doi: 10.1145/3458723. URL https://arxiv.org/abs/1803.09010.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1705.08500.

Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019. URL https://arxiv.org/abs/1901.09192.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel two-sample test. In *Journal of Machine Learning Research*, volume 13, pp. 723–773, 2012. URL https://jmlr.org/papers/v13/gretton12a.html.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017. URL https://arxiv.org/abs/1706.04599.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://arxiv.org/abs/1903.12261.

David V. Hinkley. Inference about the change-point in a sequence of random variables. *Biometrika*, 57 (1):1–17, 1970. doi: 10.1093/biomet/57.1.1. URL https://doi.org/10.1093/biomet/57.1.1.

Matthias Kirchler, Shahryar Khorasani, Marius Kloft, and Christoph Lippert. Two-sample testing using deep learning, 2020. URL https://arxiv.org/abs/1910.06239.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks, 2017. URL https://doi.org/10.1073/pnas.1611835114.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts, 2021. URL https://arxiv.org/abs/2012.07421.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1612.01474.

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *European Conference on Computer Vision (ECCV)*, pp. 614–629, 2016. doi: 10.1007/978-3-319-46493-0_37. URL https://doi.org/10.1007/978-3-319-46493-0_37.

Zachary C. Lipton, Yu-Xiang Wang, and Alex Smola. Detecting and correcting for label shift with black box predictors. *arXiv preprint*, 2018. URL https://arxiv.org/abs/1802.03916.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint*, 2016. URL https://arxiv.org/abs/1610.06545.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1706.08840.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAccT)*, 2019. doi: 10.1145/3287560.3287596. URL https://arxiv.org/abs/1810.03993.

Akshay Naresh Modi, Chiu Yuen Koo, Chuan Yu Foo, Clemens Mewald, Denis M. Baylor, Eric Breck, Heng-Tze Cheng, Jarek Wilkiewicz, Levent Koc, Lukasz Lew, Martin A. Zinkevich, Martin Wicke, Mustafa Ispir, Neoklis Polyzotis, Noah Fiedel, Salem Elie Haykal, Steven Whang, Sudip Roy, Sukriti Ramesh, Vihan Jain, Xin Zhang, and Zakaria Haque. Tfx: A tensorflow-based production-scale machine learning platform. In *KDD 2017*, 2017.

E. S. Page. Continuous inspection schemes. In *Biometrika*, volume 41, pp. 100–115, 1954. doi: 10.1093/biomet/41.1-2.100. URL https://doi.org/10.1093/biomet/41.1-2.100.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1406–1415, 2019. URL https://arxiv.org/abs/1812.01754.

Juan C. Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. Performative prediction. In *International Conference on Machine Learning*, 2020. URL https://arxiv.org/abs/2002.06673.

Neoklis Polyzotis, Martin Zinkevich, Sudip Roy, Eric Breck, and Steven Whang. Data validation for machine learning. In A. Talwalkar, V. Smith, and M. Zaharia (eds.), *Proceedings of Machine Learning and Systems*, volume 1, pp. 334–347, 2019. URL https://proceedings.mlsys.org/paper_files/paper/2019/file/928f1160e52192e3e0017fb63ab65391-Paper.pdf.

Joaquín Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence (eds.). *Dataset Shift in Machine Learning*. The MIT Press, 2008. doi: 10.7551/mitpress/9780262170055.001.0001. URL https://mitpress.mit.edu/9780262545877/dataset-shift-in-machine-learning/.

Stephan Rabanser, Stephan Günnemann, and Zachary C. Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1810.11953.

Inioluwa Deborah Raji, Andrew Smart, Rebecca N. White, Margaret Mitchell, Timnit Gebru, Ben Hutchinson, Jamila Smith-Loud, Daniel Theron, and Parker Barnes. Closing the AI accountability gap: Defining an end-to-end framework for internal algorithmic auditing. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 2020. URL https://arxiv.org/abs/2001.00973.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5533–5542, 2017. doi: 10.1109/CVPR.2017.587.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-group generalization. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/abs/1911.08731.

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. In *Advances in Neural Information Processing Systems (NeurIPS), Workshop on Software Engineering for Machine Learning*, 2015. URL https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000. doi: 10.1016/S0378-3758(00)00115-4. URL https://www.sciencedirect.com/science/article/pii/S0378375800001154.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning (ICML)*, 2020. URL https://arxiv.org/abs/1909.13231.

Alexey Tsymbal. The problem of concept drift: Definitions and related work, 2004. URL https://www.semanticscholar.org/paper/The-problem-of-concept-drift%3A-definitions-and-work-Tsymbal/30eac73e9b482bc28b5b68cd585557de48d0618f.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005. doi: 10.1007/b106715. URL https://doi.org/10.1007/b106715.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=uXl3bZLkr3c.

Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996. doi: 10.1007/BF00116900. URL https://link.springer.com/article/10.1007/BF00116900.

APPENDIX

A PROBLEM SETUP AND OPERATING CONSTRAINTS

We consider a deployed learning system that processes a potentially non-stationary stream over a finite horizon $t \in \{1, \dots, T\}$. At each time step t, the system receives an input $x_t \in \mathcal{X}$, produces an output (prediction or abstention), and may later receive the corresponding label $y_t \in \mathcal{Y}$. Distribution drift is captured by allowing the deployment distribution $\mathcal{D}_t$ governing (x_t, y_t) to change with time.

A.1 STREAMING DATA WITH DELAYED SUPERVISION

We model delayed supervision with an observation time variable $o_t \in \{t, t+1, \dots\} \cup \{\infty\}$ indicating when the label y_t becomes available. The event $o_t = \infty$ represents that y_t is never observed. The set of labeled examples available by time t is

$$\mathcal{S}_t \;=\; \{(x_i, y_i) \;:\; 1 \le i \le t \text{ and } o_i \le t\}. \tag{27}$$

In this definition, i indexes an earlier time step at which input x_i was observed, the condition $o_i \le t$ indicates that its label y_i has arrived by time t, and $\mathcal{S}_t$ is the only supervised data accessible for calibration, adaptation, and retraining decisions at time t. The evolving data-generating process is represented by a time-indexed distribution $\mathcal{D}_t$ for pairs (x_t, y_t), allowing temporal, covariate, label, concept, or subgroup shifts.

A.2 PREDICTIVE MODEL AND OUTPUTS

The deployed predictor is parameterized by θ and outputs a categorical distribution:

$$p_\theta(y \mid x) \;=\; \mathrm{softmax}\big(h_\theta(x)\big). \tag{28}$$

Here $h_\theta : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ maps an input x to logits, $\mathrm{softmax}(\cdot)$ converts logits into class probabilities, $p_\theta(y \mid x)$ is the predicted probability assigned to label $y \in \mathcal{Y}$, and θ denotes all model parameters. The system's point prediction is

$$\hat{y}_\theta(x) \;=\; \arg\max_{y \in \mathcal{Y}} p_\theta(y \mid x), \tag{29}$$

where $\hat{y}_\theta(x)$ is the most probable class under the predictive distribution.

A.3 DRIFT EVIDENCE FROM UNLABELED MONITORING

To sense drift without requiring immediate labels, the system maintains two unlabeled buffers: a reference buffer $\mathcal{B}_{\mathrm{ref}}$ representing nominal conditions and a recent buffer $\mathcal{B}_t$ of size n containing the most recent inputs,

$$\mathcal{B}_t \;=\; \{x_{t-n+1}, \dots, x_t\}. \tag{30}$$

A suite of monitoring functions computes statistics comparing these buffers, producing an evidence vector

$$z_t \;=\; \big(M_1(\mathcal{B}_t, \mathcal{B}_{\mathrm{ref}}), \dots, M_m(\mathcal{B}_t, \mathcal{B}_{\mathrm{ref}})\big) \;\in\; \mathbb{R}^m. \tag{31}$$

In this equation, $M_j(\cdot, \cdot)$ denotes the j-th monitor, such as a representation discrepancy measure, a two-sample classifier score, or an uncertainty shift statistic. The vector z_t aggregates heterogeneous drift signals into a fixed-dimensional summary that can be computed online and passed to the belief model and controller.

A.4 INTERVENTIONS AND ACTION SPACE

At each time step t, the system selects an intervention a_t from a discrete action set $\mathcal{A}$. Actions represent operational responses available in real deployments, including inexpensive adjustments (e.g., recalibration), medium-cost responses (e.g., test-time adaptation), and heavy-weight interventions (e.g., retraining or rollback). Table 2 defines each action precisely and assigns a normalized operational cost used throughout the paper.

Table 2: **Action space and normalized operational costs used by the controller.**

Action	Name	Operational effect	Cost (units)
A_0	No-op	Continue inference with current parameters θ and current output policy.	0.0
A_1	Recalibrate	Update a lightweight calibrator using $\mathcal{S}_t$; logits unchanged.	0.2
A_2	Test-time adapt	Apply a small number of adaptation steps on $\mathcal{B}_t$; update $\theta \leftarrow \theta'$.	1.0
A_3	Query labels	Request k_t labels from recent inputs; add to $\mathcal{S}_{t'}$ when they arrive.	$0.05\,k_t$
A_4	Retrain	Trigger retraining using $\mathcal{S}_t$ plus logged data; deploy updated θ.	12.0
A_5	Rollback	Revert to a previously validated checkpoint θ_{safe}.	1.5
A_6	Abstain/handoff	Abstain on selected inputs and route to fallback (e.g., human review), reducing coverage.	0.3

A.5 Cost model

We represent deployment burden with a nonnegative cost function $c : \mathcal{A} \to \mathbb{R}_{\geq 0}$. To reflect distinct operational constraints, we decompose cost into labeling, compute, and latency components:

$$c(a_t) \;=\; c_{\text{label}}(a_t) \;+\; c_{\text{comp}}(a_t) \;+\; c_{\text{lat}}(a_t). \tag{32}$$

In this equation, $c_{\text{label}}(a_t)$ quantifies human annotation or review effort (relevant for A_3 and potentially A_6), $c_{\text{comp}}(a_t)$ measures computational overhead (dominant for A_2 and A_4), and $c_{\text{lat}}(a_t)$ captures latency or throughput degradation induced by the intervention. The cumulative operational cost over the deployment horizon is

$$C_{\text{tot}} \;=\; \sum_{t=1}^{T} c(a_t), \tag{33}$$

where $c(a_t)$ is the per-step cost and C_{tot} summarizes the overall resource and process burden.

A.6 Safety objective under drift

We evaluate predictive quality using a bounded loss function $\ell(\hat{y}, y) \in [0, 1]$, such as the zero-one loss for classification. The population risk at time t is

$$R_t \;=\; \mathbb{E}_{(x,y)\sim\mathcal{D}_t}\Big[\ell\big(\hat{y}_\theta(x), y\big)\Big]. \tag{34}$$

In this definition, $\mathcal{D}_t$ is the time-dependent deployment distribution, $\hat{y}_\theta(x)$ is the model's predicted label under parameters θ, and $\ell(\cdot, \cdot)$ is the task loss. The primary safety requirement is to keep risk below a target threshold τ whenever the system produces an automated prediction:

$$R_t \;\leq\; \tau. \tag{35}$$

When the system can abstain and hand off uncertain cases, it uses an acceptance function $s_t : \mathcal{X} \to \{0, 1\}$, where $s_t(x) = 1$ indicates that the system predicts and $s_t(x) = 0$ indicates abstention. The selective risk and coverage are

$$R_t^{\text{sel}} \;=\; \mathbb{E}_{(x,y)\sim\mathcal{D}_t}\Big[\ell\big(\hat{y}_\theta(x), y\big) \;\Big|\; s_t(x) = 1\Big], \tag{36}$$

$$\kappa_t \;=\; \mathbb{P}_{(x,y)\sim\mathcal{D}_t}\big(s_t(x) = 1\big). \tag{37}$$

In these equations, R_t^{sel} measures error conditional on the system choosing to predict, and κ_t is the fraction of inputs receiving automated predictions. The safety requirement with abstention is

$$R_t^{\text{sel}} \;\leq\; \tau \qquad \text{and} \qquad \kappa_t \;\geq\; \kappa_{\min}, \tag{38}$$

where $\kappa_{\min}$ enforces a minimum coverage level, preventing degenerate solutions that satisfy safety by abstaining on nearly all inputs.

A.7 EVALUATION METRICS

We evaluate monitoring-and-control methods using metrics that reflect both reliability and operational constraints. In controlled streaming evaluations, we denote by t_0 the onset time of a drift event. Let $\mathrm{Alarm}_t \in \{0, 1\}$ indicate whether the monitoring stack triggers an alarm at time t. The detection delay is

$$T_{\mathrm{det}} \;=\; \min\{t \geq t_0 : \mathrm{Alarm}_t = 1\} \;-\; t_0. \tag{39}$$

Here T_{det} is the number of time steps between drift onset and the first alarm, measuring how quickly the monitoring layer detects change.

To measure how quickly the system returns to safe operation after drift, we define a deployed risk metric $R_t^\star$ that equals R_t when the system always predicts and equals R_t^{sel} when abstention is enabled. The recovery time is

$$T_{\mathrm{rec}} \;=\; \min\{t \geq t_0 : R_t^\star \leq \tau\} \;-\; t_0. \tag{40}$$

In this equation, T_{rec} counts the time steps from drift onset until the deployed risk metric falls below the safety threshold τ.

We quantify unsafe operation through the number of safety violations over the horizon:

$$V \;=\; \sum_{t=1}^{T} \mathbb{I}(R_t^\star > \tau)\,. \tag{41}$$

Here $\mathbb{I}(\cdot)$ is an indicator function, and V counts the total number of time steps where deployed risk exceeds the target threshold.

To assess subgroup robustness under drift, we assume a set of groups $\mathcal{G}$, such as demographic groups, domains, or discovered clusters. The worst-group accuracy at time t is

$$\mathrm{Acc}_t^{\mathrm{wg}} \;=\; \min_{g \in \mathcal{G}} \; \mathbb{P}_{(x,y)\sim\mathcal{D}_t}\big(\hat{y}_\theta(x) = y \;\big|\; g(x) = g\big). \tag{42}$$

In this definition, $g(x)$ denotes the group membership of input x, the conditional probability measures accuracy within group g, and the minimum over $\mathcal{G}$ isolates the most adversely affected group at time t. Alongside T_{det}, T_{rec}, and V, we report the cumulative operational cost C_{tot} from Eq. (8), yielding a joint view of reliability, responsiveness, and operational burden.

B ALGORITHM

Algorithm 1 summarizes the complete online loop. The loop updates monitoring signals, drift-type belief, and the active risk certificate; it then selects an action under feasibility constraints, applies the action, and logs outcomes for auditing and later analysis.

C EXPERIMENTS

We evaluate drift monitoring and response in a streaming setting where data arrive sequentially and the deployed system must decide when to recalibrate, adapt, request labels, retrain, roll back, or fall back (see Appendix E for more details). The stream is constructed by concatenating either time-ordered chunks (when timestamps exist) or environment-ordered chunks (when datasets are organized by domains). Let $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ denote environments (e.g., hospitals in Camelyon17 or domains in DomainNet). The stream assigns each interval to one environment:

$$(x_t, y_t) \sim \mathcal{D}_t, \qquad \mathcal{D}_t = \mathcal{D}^{(k)} \;\text{ for }\; t \in \{T_{k-1} + 1, \ldots, T_k\}. \tag{43}$$

In this equation, $\mathcal{D}^{(k)}$ is the environment-specific distribution for $\mathcal{E}_k$, T_k is the final time index of environment k, and $T_0 = 0$. This protocol produces realistic distribution transitions while preserving within-environment temporal coherence.

To test robustness beyond discrete domain switches, we also use mixture-controlled drift schedules that interpolate between a nominal distribution $\mathcal{D}^{\mathrm{ref}}$ and a shifted distribution $\mathcal{D}^{\mathrm{shift}}$:

$$\mathcal{D}_t \;=\; (1 - \alpha_t)\,\mathcal{D}^{\mathrm{ref}} \;+\; \alpha_t\,\mathcal{D}^{\mathrm{shift}}. \tag{44}$$

Algorithm 1. Drift-to-Action Control with Active Risk Certificate

Input: Reference buffer $\mathcal{B}_{\mathrm{ref}}$, monitor window size n, certificate window size N, risk target τ, failure level δ, labeling budget B_{lab}, retrain cooldown Δ_{rt}, rollback cooldown Δ_{rb}, cost tradeoff λ, gain table $G(d, a)$

Output: Action sequence $\{a_t\}_{t=1}^{T}$ and audit log $\mathcal{L}$

Initialize model parameters θ
Initialize calibrator parameters η
Initialize drift-type belief $b_0(d)$ for all $d \in \mathcal{D}$ with $\sum_{d\in\mathcal{D}} b_0(d) = 1$
Initialize labeled set $\mathcal{S}_0 \leftarrow \emptyset$ and audit log $\mathcal{L} \leftarrow \emptyset$
Initialize last intervention times $t^{\mathrm{rt}}_{\mathrm{last}} \leftarrow -\infty$ and $t^{\mathrm{rb}}_{\mathrm{last}} \leftarrow -\infty$
Initialize remaining labeling budget $B \leftarrow B_{\mathrm{lab}}$
for $t \leftarrow 1$ **to** T **do**
  Receive input x_t and update recent buffer $\mathcal{B}_t = \{x_{t-n+1}, \ldots, x_t\}$
  Compute evidence vector $z_t = \big(M_1(\mathcal{B}_t, \mathcal{B}_{\mathrm{ref}}), \ldots, M_m(\mathcal{B}_t, \mathcal{B}_{\mathrm{ref}})\big)$
  Update belief $b_t(d) = \mathbb{P}(D_t = d \mid z_{1:t})$ for all $d \in \mathcal{D}$
  Construct certifiable window $W_t = \{t - d - N + 1, \ldots, t - d\}$
  Choose audit size $n_t \leftarrow \mathrm{QueryPolicy}(z_t, b_t, U_{t-1}(\delta), B)$
  Uniformly sample audit indices $Q_t \subseteq W_t$ with $|Q_t| = n_t$
  $J_t \leftarrow Q_t \setminus \{i : (x_i, y_i) \in \mathcal{S}_{t-1}\}$
  Reveal/request labels $\{y_i : i \in J_t\}$ and update budget $B \leftarrow B - |J_t|$
  Update $\mathcal{S}_t \leftarrow \mathcal{S}_{t-1} \cup \{(x_i, y_i) : i \in J_t\}$
  Compute queried losses $\ell_i = \ell(\hat{y}_\theta(x_i), y_i)$ for all $i \in Q_t$
  Compute empirical audited risk $\widehat{R}_t = \frac{1}{|Q_t|} \sum_{i\in Q_t} \ell_i$
  Set $\delta_t = \frac{6\delta}{\pi^2 t^2}$
  Compute certificate $U_t(\delta) = \widehat{R}_t + \mathrm{rad}(|Q_t|, \delta_t)$
  if $U_t(\delta) > \tau$ **then**
    Set action $a_t \leftarrow A_6$ `// abstain/handoff`
    if $t - t^{\mathrm{rb}}_{\mathrm{last}} \geq \Delta_{\mathrm{rb}}$ **then**
      Schedule rollback and update $t^{\mathrm{rb}}_{\mathrm{last}} \leftarrow t$
    else if $t - t^{\mathrm{rt}}_{\mathrm{last}} \geq \Delta_{\mathrm{rt}}$ **then**
      Schedule retraining and update $t^{\mathrm{rt}}_{\mathrm{last}} \leftarrow t$
  else
    Form feasible action set $\mathcal{A}_t^{\mathrm{feas}} \subseteq \mathcal{A}$ using cooldowns and remaining budget B
    foreach $a \in \mathcal{A}_t^{\mathrm{feas}}$ **do**
      Compute expected benefit $\Delta_t(a) = \sum_{d\in\mathcal{D}} b_t(d)\, G(d, a)$
      Compute utility $\mathcal{U}_t(a) = \Delta_t(a) - \lambda\, c(a)$
    Choose action $a_t \leftarrow \arg\max_{a\in\mathcal{A}_t^{\mathrm{feas}}} \mathcal{U}_t(a)$
  Apply action a_t:
  if $a_t = A_1$ **then**
    Update calibrator η using $\mathcal{S}_t$
  if $a_t = A_2$ **then**
    Perform test-time adaptation on recent inputs; update $\theta \leftarrow \theta'$
  if $a_t = A_4$ **then**
    Retrain model using $\mathcal{S}_t$ and logged data; deploy updated θ
  if $a_t = A_5$ **then**
    Rollback to checkpoint θ_{safe} and associated calibrator
  if $a_t = A_6$ **then**
    Abstain/handoff using policy $s_t(x)$
  Log $(t, z_t, b_t, U_t(\delta), a_t, c(a_t), k_t)$ into $\mathcal{L}$

Here $\alpha_t \in [0, 1]$ is a time-varying drift intensity, with $\alpha_t = 0$ representing nominal conditions and $\alpha_t = 1$ representing fully shifted conditions. We instantiate three drift patterns via α_t. Sudden drift

uses $\alpha_t = \mathbb{I}(t \geq t_0)$ where t_0 is drift onset. Gradual drift uses a sigmoid ramp

$$\alpha_t \;=\; \frac{1}{1+\exp\big(-\rho\,(t-t_0)\big)}, \tag{45}$$

where $\rho > 0$ controls the ramp speed. Recurring drift uses a periodic schedule

$$\alpha_t \;=\; \frac{1}{2}\Big(1+\sin(2\pi t/P)\Big), \tag{46}$$

where P is the period length in time steps, producing repeated returns to nominal conditions and repeated drift episodes.

Operational constraints are integrated directly into the evaluation. Each action $a_t \in \mathcal{A}$ incurs a normalized cost $c(a_t)$ and heavy interventions are rate-limited by cooldowns. The total deployment burden is measured by

$$C_{\text{tot}} \;=\; \sum_{t=1}^{T} c(a_t), \tag{47}$$

where T is the stream length. We also simulate realistic annotation pipelines through label delay. If the controller requests the label for index i, the label becomes available after d steps:

$$o_i \;=\; i + d, \tag{48}$$

where o_i is the observation time at which y_i arrives. This delay affects both calibration monitoring and risk certification, because losses can only be computed once labels are observed. Table 3 specifies the concrete action costs, prerequisites, and cooldowns used by all methods in the protocol, and it is referenced by the controller when enforcing feasibility constraints.

D Full theoretical guarantee

This appendix provides a full, self-contained guarantee for the active risk certificate used in Section 4. We first state the probabilistic model under which the certificate is valid, then derive an anytime-valid upper confidence sequence for the mean of bounded losses under adaptive (optional-stopping) label querying, and finally extend the result to selective risk when abstention is enabled.

Table 3: **Streaming protocol action costs and prerequisites.** Normalized costs and cooldowns.

Act.	Name	Prereq./Effect	Cooldown	Cost
A_0	No-op	Keep θ.	–	0.0
A_1	Recalib.	Update from $\mathcal{S}_t$.	–	0.2
A_2	TTA	Adapt on $\mathcal{B}_t$.	–	1.0
A_3	Query	Request k_t; delay d.	–	$0.05k_t$
A_4	Retrain	Retrain + deploy.	Δ_{rt}	12.0
A_5	Rollback	Restore θ_{safe}.	Δ_{rb}	1.5
A_6	Abstain	Handoff; less coverage.	–	0.3

D.1 Windowed risk, random sampling, and filtration

Fix a time t and a window length $N \in \mathbb{N}$. The current deployment regime is represented by the index set

$$W_t \;=\; \{t-N+1,\ldots,t\}. \tag{49}$$

In this equation, t is the current time index and W_t contains the most recent N indices.

Each index $i \in W_t$ corresponds to an example (x_i, y_i) and an associated bounded loss

$$L_{t,i} \;=\; \ell\big(\hat{y}_\theta(x_i), y_i\big), \qquad 0 \leq L_{t,i} \leq 1. \tag{50}$$

In this equation, $\ell(\cdot,\cdot)$ is the task loss, $\hat{y}_\theta(x_i)$ is the model prediction at index i, and the bound $[0,1]$ holds for standard classification losses such as $\mathbb{I}(\hat{y}_\theta(x_i) \neq y_i)$.

The windowed (finite-population) risk is the mean loss over the current window:

$$R_t \;=\; \frac{1}{N}\sum_{i\in W_t} L_{t,i}. \tag{51}$$

In this equation, R_t is the quantity we want to upper bound online.

The certificate queries n_t labels by sampling indices uniformly at random from W_t. Let

$$I_{t,1}, I_{t,2}, \ldots, I_{t,n_t} \;\sim\; \mathrm{Unif}(W_t), \tag{52}$$

where $\mathrm{Unif}(W_t)$ denotes the uniform distribution over the set W_t. The sampling may be done with replacement; sampling without replacement is discussed in Section D.6. The observed queried losses are

$$X_{t,j} \;=\; L_{t,I_{t,j}}, \qquad j \in \{1, \ldots, n_t\}. \tag{53}$$

In this equation, $X_{t,j}$ is the loss of the j-th randomly sampled index within the window at time t.

The certificate uses the empirical mean of the queried losses,

$$\widehat{R}_{t,n} \;=\; \frac{1}{n} \sum_{j=1}^{n} X_{t,j}, \tag{54}$$

where n is the number of queried labels actually used in the estimate.

To capture adaptive label querying, we define a filtration that represents all information available after n queried labels in window t:

$$\mathcal{F}_{t,n} \;=\; \sigma(z_{1:t},\, I_{t,1}, X_{t,1},\, \ldots,\, I_{t,n}, X_{t,n})\,. \tag{55}$$

In this equation, $z_{1:t}$ is the monitor history and $\sigma(\cdot)$ denotes the σ-algebra generated by the listed random variables. The query count n_t is allowed to be an $\mathcal{F}_{t,n}$-adapted (data-dependent) stopping rule, meaning n_t can depend on past observed evidence and queried labels.

The key validity condition used throughout this appendix is the following.

Condition A (Random auditing within the current window). At each time t, the queried indices $I_{t,1}, \ldots, I_{t,n_t}$ are sampled uniformly from W_t, and the sampling mechanism is independent of the unobserved labels given the already observed history.

This condition allows the certificate to treat the queried losses as an unbiased sample from the window mean risk R_t, even when the decision to query more labels is adaptive.

D.2 A TIME-UNIFORM CONFIDENCE SEQUENCE FOR BOUNDED MEANS

We derive an anytime-valid upper confidence sequence for a bounded mean under adaptive stopping. The statement is formulated for a generic bounded sequence and then specialized to the windowed risk setting.

Let $X_1, X_2, \ldots$ be random variables satisfying

$$0 \le X_j \le 1, \qquad \mathbb{E}[X_j] = \mu, \qquad \text{and } X_j \text{ are i.i.d..} \tag{56}$$

In this equation, μ is the unknown mean to be bounded. Define the sample mean

$$\widehat{\mu}_n \;=\; \frac{1}{n} \sum_{j=1}^{n} X_j. \tag{57}$$

In this equation, n is the number of samples and $\widehat{\mu}_n$ is their empirical average.

We use Hoeffding's inequality at each fixed n:

$$\mathbb{P}(\mu \ge \widehat{\mu}_n + \epsilon) \;\le\; \exp\!\big(-2n\epsilon^2\big)\,. \tag{58}$$

In this inequality, $\epsilon > 0$ is a deviation level and the bound relies only on the fact that $X_j \in [0, 1]$.

To make the bound *anytime-valid* over all n, we allocate failure probability across time using a summable schedule. Let

$$\delta_n \;=\; \frac{6\delta}{\pi^2 n^2}, \qquad \delta \in (0, 1). \tag{59}$$

In this equation, δ is the target overall failure probability and δ_n is the portion allocated to sample size n. The constant is chosen so that the sum is bounded:

$$\sum_{n=1}^{\infty} \delta_n \;=\; \frac{6\delta}{\pi^2} \sum_{n=1}^{\infty} \frac{1}{n^2} \;=\; \delta. \tag{60}$$

In this equation, $\sum_{n\geq 1} 1/n^2 = \pi^2/6$ is the Basel identity, ensuring the total allocated failure probability is exactly δ.

Define the confidence radius

$$\mathrm{rad}(n,\delta) \;=\; \sqrt{\frac{1}{2n}\log\!\left(\frac{1}{\delta_n}\right)} \;=\; \sqrt{\frac{1}{2n}\log\!\left(\frac{\pi^2 n^2}{6\delta}\right)}. \tag{61}$$

In this equation, n is the sample size and $\mathrm{rad}(n,\delta)$ shrinks as n increases.

Lemma A.1 ***(Anytime-valid upper confidence sequence via stitching).*** *For any $\delta \in (0,1)$ and the radius $\mathrm{rad}(n,\delta)$ defined above, it holds that*

$$\mathbb{P}(\forall n \geq 1 :\; \mu \leq \widehat{\mu}_n + \mathrm{rad}(n,\delta)) \;\geq\; 1-\delta. \tag{62}$$

In this probability statement, the event inside holds simultaneously *for all sample sizes n, which implies validity under optional stopping.*

Proof. For any fixed n, set $\epsilon = \mathrm{rad}(n,\delta)$ in Hoeffding's inequality to obtain

$$\mathbb{P}(\mu \geq \widehat{\mu}_n + \mathrm{rad}(n,\delta)) \;\leq\; \exp\!\big(-2n\,\mathrm{rad}(n,\delta)^2\big) \;=\; \exp\!\left(-\log\!\left(\frac{1}{\delta_n}\right)\right) \;=\; \delta_n. \tag{63}$$

In this equation chain, the first step applies Hoeffding, the second step substitutes the definition of $\mathrm{rad}(n,\delta)$, and the last step simplifies the exponential.

Define the bad event at time n by

$$E_n \;=\; \{\mu \geq \widehat{\mu}_n + \mathrm{rad}(n,\delta)\}\,. \tag{64}$$

Then, by the union bound,

$$\mathbb{P}(\exists n \geq 1 :\; E_n) \;\leq\; \sum_{n=1}^{\infty} \mathbb{P}(E_n) \;\leq\; \sum_{n=1}^{\infty} \delta_n \;=\; \delta. \tag{65}$$

In this bound, the first inequality uses the union bound, the second uses $\mathbb{P}(E_n) \leq \delta_n$, and the last uses the summability property shown earlier. Taking complements yields the claim. □

A crucial consequence of Lemma A.1 is optional-stopping validity. Let $\widehat{n}$ be any (possibly random) stopping time that depends on the observed data, such as an adaptive query rule that stops once the bound is sufficiently tight. Since the event in Lemma A.1 holds for all n simultaneously, it also holds at $n = \widehat{n}$:

$$\mathbb{P}(\mu \leq \widehat{\mu}_{\widehat{n}} + \mathrm{rad}(\widehat{n},\delta)) \;\geq\; 1-\delta. \tag{66}$$

In this equation, $\widehat{n}$ is data-dependent, yet the bound remains valid because the confidence sequence is time-uniform.

D.3 Applying the confidence sequence to windowed risk certificates

We now map the generic result to the windowed risk certificate at time t. Under Condition A, the queried losses $X_{t,1}, \ldots, X_{t,n}$ satisfy

$$0 \leq X_{t,j} \leq 1, \qquad \mathbb{E}[X_{t,j} \mid W_t] \;=\; R_t, \tag{67}$$

where R_t is the window mean risk. The empirical risk based on n queried labels is

$$\widehat{R}_{t,n} \;=\; \frac{1}{n}\sum_{j=1}^{n} X_{t,j}. \tag{68}$$

In this equation, $\widehat{R}_{t,n}$ is the empirical mean of queried losses within the window at time t.

Define the per-time certificate using the confidence radius:

$$U_{t,n}(\delta_t) \;=\; \widehat{R}_{t,n} + \mathrm{rad}(n,\delta_t), \tag{69}$$

where $\delta_t \in (0,1)$ is the failure probability budget assigned to time t.

To obtain a guarantee that holds simultaneously over all times $t \in \{1, \dots, T\}$, we allocate the overall failure probability δ across time. A convenient summable choice is

$$\delta_t \;=\; \frac{6\delta}{\pi^2 t^2}, \qquad t \geq 1, \tag{70}$$

which satisfies $\sum_{t\geq 1} \delta_t = \delta$.

Let n_t be the (possibly adaptive) number of labels queried in window t, and define the deployed certificate as

$$U_t(\delta) \;=\; \widehat{R}_{t,n_t} + \mathrm{rad}(n_t, \delta_t). \tag{71}$$

In this equation, $U_t(\delta)$ is computed from the queried sample mean $\widehat{R}_{t,n_t}$ and the anytime-valid radius evaluated at the random sample size n_t.

Theorem A.2 ***(Anytime-valid window risk certificate under adaptive querying).*** *Assume Condition A holds for each time t, and the loss is bounded as $L_{t,i} \in [0,1]$. Then with probability at least $1-\delta$,*

$$\forall t \geq 1: \quad R_t \;\leq\; U_t(\delta). \tag{72}$$

In this inequality, the bound holds simultaneously for all times t and remains valid even when n_t is chosen adaptively from past evidence and previously observed labels.

Proof. Fix a time t. Conditional on the window W_t, Lemma A.1 applies to the queried losses in that window, so

$$\mathbb{P}\Big(\forall n \geq 1:\; R_t \leq \widehat{R}_{t,n} + \mathrm{rad}(n, \delta_t) \;\Big|\; W_t\Big) \;\geq\; 1 - \delta_t. \tag{73}$$

Since the bound holds for all n, it holds at $n = n_t$ even if n_t is chosen adaptively:

$$\mathbb{P}\Big(R_t \leq \widehat{R}_{t,n_t} + \mathrm{rad}(n_t, \delta_t)\Big) \;\geq\; 1 - \delta_t. \tag{74}$$

Define the bad event at time t by

$$B_t \;=\; \{R_t > U_t(\delta)\}. \tag{75}$$

Then $\mathbb{P}(B_t) \leq \delta_t$, and applying the union bound across all $t \geq 1$ yields

$$\mathbb{P}(\exists t \geq 1:\; B_t) \;\leq\; \sum_{t=1}^{\infty} \mathbb{P}(B_t) \;\leq\; \sum_{t=1}^{\infty} \delta_t \;=\; \delta. \tag{76}$$

Taking complements gives $\mathbb{P}(\forall t:\; R_t \leq U_t(\delta)) \geq 1 - \delta$. □

This theorem formalizes the core certificate guarantee used by the controller: even with delayed supervision and adaptive query schedules, as long as auditing samples are uniformly random within the current window, the bound is valid simultaneously over time.

D.4 Safety gating guarantee for non-abstaining operation

We now formalize the safety gating claim used in Theorem 1 of the main text. Suppose the system follows the rule that it only issues predictions when the certificate is below the risk threshold τ. Let $\mathrm{Pred}_t \in \{0,1\}$ denote whether the system actually produces an automated prediction at time t, with

$$\mathrm{Pred}_t \;=\; \mathbb{I}(U_t(\delta) \leq \tau). \tag{77}$$

In this equation, $\mathrm{Pred}_t = 1$ means the system predicts and $\mathrm{Pred}_t = 0$ means it activates fallback (abstain/handoff).

Corollary A.3 (No unsafe prediction without certificate violation). Under the assumptions of Theorem A.2, with probability at least $1-\delta$,

$$\forall t \geq 1: \quad \mathrm{Pred}_t = 1 \;\Rightarrow\; R_t \leq \tau. \tag{78}$$

In this implication, the system cannot be in a regime where it both predicts and the true window risk exceeds τ unless the certificate fails (an event of probability at most δ).

Proof. On the event $\{R_t \leq U_t(\delta) \text{ for all } t\}$ from Theorem A.2, if $\mathrm{Pred}_t = 1$ then $U_t(\delta) \leq \tau$, and therefore

$$R_t \;\leq\; U_t(\delta) \;\leq\; \tau. \tag{79}$$

This holds for all t simultaneously on the same high-probability event. □

D.5 EXTENSION: CERTIFIED SELECTIVE RISK UNDER ABSTENTION

When abstention is enabled, the deployed performance metric is typically the selective risk, which conditions on the system choosing to predict. Let $s_t(x) \in \{0, 1\}$ be the acceptance function at time t, where $s_t(x) = 1$ indicates that the system predicts and $s_t(x) = 0$ indicates abstention. For an index $i \in W_t$, define

$$S_{t,i} \;=\; s_t(x_i), \qquad L_{t,i} \;=\; \ell\big(\hat{y}_\theta(x_i), y_i\big). \tag{80}$$

In these equations, $S_{t,i}$ is the accept indicator and $L_{t,i}$ is the loss.

The window coverage is the mean accept rate

$$\kappa_t \;=\; \frac{1}{N} \sum_{i \in W_t} S_{t,i}. \tag{81}$$

In this equation, κ_t is the fraction of window points on which the system predicts.

The window selective risk is the average loss conditional on acceptance:

$$R_t^{\mathrm{sel}} \;=\; \frac{\sum_{i \in W_t} L_{t,i} S_{t,i}}{\sum_{i \in W_t} S_{t,i}} \;=\; \frac{m_t}{\kappa_t}, \tag{82}$$

where we define the numerator mean

$$m_t \;=\; \frac{1}{N} \sum_{i \in W_t} L_{t,i} S_{t,i}. \tag{83}$$

In these equations, m_t is the mean *accepted loss mass* and κ_t is the mean acceptance rate, so the ratio equals the conditional mean loss.

Under random sampling within the window, each queried index $I_{t,j}$ yields the pair

$$Y_{t,j} \;=\; L_{t,I_{t,j}} S_{t,I_{t,j}}, \qquad A_{t,j} \;=\; S_{t,I_{t,j}}. \tag{84}$$

In these equations, $Y_{t,j}$ is the product of loss and acceptance, and $A_{t,j}$ is the acceptance indicator. Both satisfy bounds $0 \le Y_{t,j} \le 1$ and $0 \le A_{t,j} \le 1$.

Define the empirical means from n queried labels:

$$\widehat{m}_{t,n} \;=\; \frac{1}{n} \sum_{j=1}^{n} Y_{t,j}, \qquad \widehat{\kappa}_{t,n} \;=\; \frac{1}{n} \sum_{j=1}^{n} A_{t,j}. \tag{85}$$

In these equations, $\widehat{m}_{t,n}$ estimates m_t and $\widehat{\kappa}_{t,n}$ estimates κ_t.

We build an anytime-valid upper bound for m_t and a lower bound for κ_t. Using the same confidence radius function $\mathrm{rad}(n, \cdot)$, define

$$U_{t,n}^{m}(\delta_t^m) \;=\; \widehat{m}_{t,n} + \mathrm{rad}(n, \delta_t^m), \qquad L_{t,n}^{\kappa}(\delta_t^\kappa) \;=\; \widehat{\kappa}_{t,n} - \mathrm{rad}(n, \delta_t^\kappa). \tag{86}$$

In these equations, $U_{t,n}^m$ is an upper bound on m_t and $L_{t,n}^\kappa$ is a lower bound on κ_t.

The selective-risk certificate is then formed as an upper bound on the ratio:

$$U_{t,n}^{\mathrm{sel}} \;=\; \frac{U_{t,n}^{m}(\delta_t^m)}{\max\big(L_{t,n}^{\kappa}(\delta_t^\kappa),\, \kappa_{\min}\big)}. \tag{87}$$

In this equation, $\kappa_{\min} > 0$ is the minimum coverage requirement used in the main text, and the $\max(\cdot, \kappa_{\min})$ term prevents division by a value that is too small.

To allocate overall failure probability, we choose summable schedules δ_t^m and δ_t^κ satisfying

$$\sum_{t \ge 1} \delta_t^m \;+\; \sum_{t \ge 1} \delta_t^\kappa \;\le\; \delta, \tag{88}$$

for example by setting $\delta_t^m = \delta_t^\kappa = 3\delta/(\pi^2 t^2)$.

Theorem A.4 (Anytime-valid selective-risk certificate). Assume Condition A holds and $L_{t,i} \in [0,1]$. With probability at least $1-\delta$, for all $t \geq 1$,

$$m_t \;\leq\; U^{m}_{t,n_t}(\delta^m_t) \qquad \text{and} \qquad \kappa_t \;\geq\; L^{\kappa}_{t,n_t}(\delta^\kappa_t), \tag{89}$$

and consequently, whenever $\kappa_t \geq \kappa_{\min}$,

$$R^{\mathrm{sel}}_t \;=\; \frac{m_t}{\kappa_t} \;\leq\; \frac{U^{m}_{t,n_t}(\delta^m_t)}{\max\!\big(L^{\kappa}_{t,n_t}(\delta^\kappa_t),\, \kappa_{\min}\big)} \;=\; U^{\mathrm{sel}}_{t,n_t}. \tag{90}$$

In these inequalities, U^{sel}_{t,n_t} is an anytime-valid upper bound on selective risk.

Proof. The variables $\{Y_{t,j}\}_{j\geq 1}$ are bounded in $[0,1]$ with mean m_t, and the variables $\{A_{t,j}\}_{j\geq 1}$ are bounded in $[0,1]$ with mean κ_t. Applying Theorem A.2 separately to each sequence with their respective failure allocations yields the simultaneous bounds in Eq. (48). On that same event, dividing the upper numerator bound by the lower denominator bound gives Eq. (27), and the $\max(\cdot, \kappa_{\min})$ preserves validity under the coverage constraint. □

This extension supports the main-text constraint $R^{\mathrm{sel}}_t \leq \tau$ with $\kappa_t \geq \kappa_{\min}$ by providing a directly certifiable quantity that can be used for gating.

D.6 Sampling without replacement (finite-population correction)

When the controller samples indices without replacement from the window W_t, the queried losses form a simple random sample from a finite population. In that case, concentration can be tightened via a finite-population correction. For a fixed $n \leq N$, Serfling's inequality implies

$$\mathbb{P}\Big(R_t \geq \widehat{R}_{t,n} + \epsilon\Big) \;\leq\; \exp\!\left(-\frac{2n\epsilon^2}{1-\frac{n-1}{N}}\right). \tag{91}$$

In this inequality, the factor $1-\frac{n-1}{N}$ reduces variance when a nontrivial fraction of the window is audited. An anytime-valid sequence can be obtained by replacing Hoeffding's fixed-n bound in Section D.2 and using the same stitching schedule δ_n, yielding a radius

$$\mathrm{rad}_{\mathrm{fp}}(n,\delta) \;=\; \sqrt{\left(1-\frac{n-1}{N}\right)\frac{1}{2n}\log\!\left(\frac{\pi^2 n^2}{6\delta}\right)}. \tag{92}$$

In this equation, $\mathrm{rad}_{\mathrm{fp}}$ is smaller than rad when n is not negligible relative to N, and it reduces to the Hoeffding-style radius when N is large.

D.7 Discussion of validity conditions under label delay

Label delay affects *when* a queried loss becomes observable, but not *which* indices are sampled. The certificate remains valid as long as the sampling rule that selects $I_{t,j}$ is uniform over W_t and does not depend on unobserved labels. Formally, if the request times are $\mathcal{F}_{t,j}$-adapted and labels arrive later, the bound applies to the eventual realized losses $X_{t,j}$ because Lemma A.1 controls deviations for all sample sizes n simultaneously. The practical effect of delay is that the controller may operate with smaller effective n_t until requested labels arrive, which enlarges $\mathrm{rad}(n_t, \delta_t)$ and therefore makes the gating rule more conservative during periods of missing supervision.

E Implementation details

This appendix provides implementation-level details for the sensing layer, belief model training, and controller configuration. All components are designed to be lightweight and reproducible, and each definition here corresponds directly to quantities used in the main text.

E.1 Monitoring signals and evidence vector

At each time step t, the sensing layer compares a recent unlabeled buffer $\mathcal{B}_t = \{x_{t-n+1}, \ldots, x_t\}$ to a reference buffer $\mathcal{B}_{\mathrm{ref}}$, and computes monitors in a representation space. Let $g_\theta : \mathcal{X} \to \mathbb{R}^p$ be the

Table 4: **Baseline configurations used in all streaming experiments.** We report the fixed hyperparameters that fully specify each baseline policy under the common action set and costs.

Baseline	Configuration
Alarm-only	Alarm when $\|z_t\|_2 > \theta_{\text{alarm}}$ with $\theta_{\text{alarm}} = 2.5$; no actions beyond logging.
Adapt-always	Apply A_2 every step; 1 gradient step per batch; learning rate 10^{-4}.
Retrain-on-schedule	Trigger A_4 every $P_{\text{rt}} = 1200$ steps, respecting cooldown Δ_{rt}.
Selective prediction only	Predict if $\max_y p_\theta(y \mid x) \geq \theta_{\text{sel}}$ with $\theta_{\text{sel}} = 0.6$; otherwise abstain ($A_6$).
Controller (no cert.)	Same as ours but remove gating: replace $U_t(\delta) > \tau$ branch by normal utility maximization.

embedding function induced by the deployed network (typically the penultimate layer). For each input x, the embedding is

$$r \;=\; g_\theta(x), \qquad r \in \mathbb{R}^p. \tag{93}$$

In this equation, x is the input, θ are current model parameters, and r is the representation used by the monitors.

We define the embedding sets

$$\mathcal{R}_t \;=\; \{g_\theta(x) : x \in \mathcal{B}_t\}, \qquad \mathcal{R}_{\text{ref}} \;=\; \{g_\theta(x) : x \in \mathcal{B}_{\text{ref}}\}. \tag{94}$$

In these equations, $\mathcal{R}_t$ contains recent embeddings and $\mathcal{R}_{\text{ref}}$ contains reference embeddings.

E.1.1 MMD MONITOR IN EMBEDDING SPACE

We use a kernel Maximum Mean Discrepancy (MMD) statistic as a representation-shift monitor. Let $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ be a positive definite kernel, and let $\{r_i\}_{i=1}^{n} \subset \mathcal{R}_t$ and $\{r'_j\}_{j=1}^{n_{\text{ref}}} \subset \mathcal{R}_{\text{ref}}$. The unbiased estimator of MMD^2 is

$$\widehat{\text{MMD}}_t^2 \;=\; \frac{1}{n(n-1)} \sum_{i \neq i'} k(r_i, r_{i'}) \;+\; \frac{1}{n_{\text{ref}}(n_{\text{ref}}-1)} \sum_{j \neq j'} k(r'_j, r'_{j'}) \;-\; \frac{2}{n\, n_{\text{ref}}} \sum_{i=1}^{n} \sum_{j=1}^{n_{\text{ref}}} k(r_i, r'_j). \tag{95}$$

In this equation, the first term measures within-recent similarity, the second term measures within-reference similarity, and the third term measures cross similarity; the statistic increases when the two embedding distributions differ.

We use an RBF kernel

$$k(u, v) \;=\; \exp\!\left(-\frac{\|u - v\|_2^2}{2\sigma^2}\right), \tag{96}$$

where $\sigma > 0$ is the kernel bandwidth. We set σ using the median heuristic on a pooled batch of embeddings from $\mathcal{R}_{\text{ref}} \cup \mathcal{R}_t$:

$$\sigma^2 \;=\; \text{median}\Big(\{\|u - v\|_2^2 : u, v \text{ in a pooled sample}\}\Big). \tag{97}$$

In this equation, the median is taken over pairwise squared distances and yields a robust bandwidth for monitoring.

E.1.2 TWO-SAMPLE DISCRIMINATOR MONITOR

To complement kernel testing, we train a lightweight discriminator that predicts whether an embedding comes from the reference or recent window. We form a labeled dataset

$$\mathcal{D}_t^{\text{disc}} \;=\; \{(r, 0) : r \in \mathcal{R}_{\text{ref}}\} \;\cup\; \{(r, 1) : r \in \mathcal{R}_t\}. \tag{98}$$

In this equation, label 0 denotes reference and label 1 denotes recent.

Let $h_\psi : \mathbb{R}^p \to (0, 1)$ be a discriminator (logistic regression or a 2-layer MLP). We fit ψ by minimizing cross-entropy

$$\mathcal{L}_{\text{disc}}(\psi) \;=\; -\frac{1}{|\mathcal{D}_t^{\text{disc}}|} \sum_{(r,y) \in \mathcal{D}_t^{\text{disc}}} \Big(y \log h_\psi(r) + (1 - y) \log(1 - h_\psi(r))\Big). \tag{99}$$

In this equation, $h_\psi(r)$ is the predicted probability that r came from the recent window.

The discriminator monitor is the held-out AUC:

$$\mathrm{AUC}_t \;=\; \mathrm{AUC}(\{h_\psi(r)\}, \{y\})\,, \tag{100}$$

where $\mathrm{AUC}(\cdot,\cdot)$ is the area under the ROC curve computed on a validation split of $\mathcal{D}_t^{\mathrm{disc}}$. Values near 0.5 indicate no detectable shift; larger values indicate separable distributions.

E.1.3 Uncertainty shift monitor

We compute predictive entropy for each input using the deployed model probabilities $p_\theta(y \mid x)$:

$$H_\theta(x) \;=\; -\sum_{y\in\mathcal{Y}} p_\theta(y \mid x)\, \log p_\theta(y \mid x). \tag{101}$$

In this equation, $\mathcal{Y}$ is the label set, and $H_\theta(x)$ is larger when predictions are less confident.

We define the window-average entropy shift

$$\Delta H_t \;=\; \frac{1}{|\mathcal{B}_t|}\sum_{x\in\mathcal{B}_t} H_\theta(x) \;-\; \frac{1}{|\mathcal{B}_{\mathrm{ref}}|}\sum_{x\in\mathcal{B}_{\mathrm{ref}}} H_\theta(x). \tag{102}$$

In this equation, $\Delta H_t > 0$ indicates an increase in uncertainty relative to nominal conditions.

E.1.4 Calibration drift monitor with delayed labels

When labels are available (after delay), we compute a streaming Expected Calibration Error (ECE) proxy on the labeled set $\mathcal{S}_t$. Let $q_\theta(x) = \max_y p_\theta(y \mid x)$ be confidence, and let $\{I_b\}_{b=1}^B$ be confidence bins. Define the bin-level statistics

$$\mathrm{conf}_b(t) \;=\; \frac{1}{|\mathcal{S}_{t,b}|}\sum_{(x,y)\in\mathcal{S}_{t,b}} q_\theta(x), \qquad \mathrm{acc}_b(t) \;=\; \frac{1}{|\mathcal{S}_{t,b}|}\sum_{(x,y)\in\mathcal{S}_{t,b}} \mathbb{I}(\hat{y}_\theta(x) = y)\,, \tag{103}$$

where

$$\mathcal{S}_{t,b} \;=\; \{(x,y)\in\mathcal{S}_t :\; q_\theta(x)\in I_b\}. \tag{104}$$

In these equations, $\mathcal{S}_{t,b}$ is the set of labeled points whose confidence falls into bin b, $\mathrm{conf}_b(t)$ is their mean confidence, and $\mathrm{acc}_b(t)$ is their empirical accuracy.

The ECE proxy is

$$\mathrm{ECE}_t \;=\; \sum_{b=1}^{B} \frac{|\mathcal{S}_{t,b}|}{|\mathcal{S}_t|}\,\big|\mathrm{acc}_b(t) - \mathrm{conf}_b(t)\big|. \tag{105}$$

In this equation, the term inside the absolute value measures miscalibration in each bin, and the weights reflect bin mass.

We use the difference to a reference value $\mathrm{ECE}_{\mathrm{ref}}$ computed on nominal labeled data:

$$\Delta\mathrm{ECE}_t \;=\; \mathrm{ECE}_t - \mathrm{ECE}_{\mathrm{ref}}. \tag{106}$$

In this equation, $\Delta\mathrm{ECE}_t > 0$ indicates worsened calibration relative to nominal conditions.

E.1.5 Slice drift monitor

When a discrete attribute $g(x) \in \mathcal{G}$ is available, we compute monitors per group:

$$\widehat{\mathrm{MMD}}^2_{t,g} \;=\; \widehat{\mathrm{MMD}}^2(\mathcal{R}_{t,g}, \mathcal{R}_{\mathrm{ref},g})\,, \qquad \Delta H_{t,g} \;=\; \overline{H}_{t,g} - \overline{H}_{\mathrm{ref},g}, \tag{107}$$

where $\mathcal{R}_{t,g} = \{g_\theta(x) : x\in\mathcal{B}_t,\; g(x) = g\}$ and similarly for $\mathcal{R}_{\mathrm{ref},g}$. When no attribute is available, we cluster embeddings using k-means and treat cluster IDs as proxy slices.

We aggregate slice evidence using the maximum across groups:

$$z_t^{\mathrm{slice}} \;=\; \max_{g\in\mathcal{G}} \widehat{\mathrm{MMD}}^2_{t,g}. \tag{108}$$

In this equation, z_t^{slice} increases when at least one slice experiences a strong shift, even if the global distribution is relatively stable.

E.1.6 FINAL EVIDENCE VECTOR

The deployed evidence vector concatenates the monitors:

$$z_t \;=\; \big[\widehat{\mathrm{MMD}}_t^2,\; \mathrm{AUC}_t,\; \Delta H_t,\; \Delta\mathrm{ECE}_t,\; z_t^{\mathrm{slice}}\big]^\top \;\in\; \mathbb{R}^m. \tag{109}$$

In this equation, m is the number of monitors used. All components are standardized online using reference-window mean and variance:

$$\widetilde{z}_{t,j} \;=\; \frac{z_{t,j} - \mu_j^{\mathrm{ref}}}{\sigma_j^{\mathrm{ref}} + \epsilon}, \tag{110}$$

where μ_j^{ref} and σ_j^{ref} are computed on $\mathcal{B}_{\mathrm{ref}}$ and ϵ prevents division by zero.

E.2 BELIEF MODEL TRAINING VIA SYNTHETIC DRIFT EPISODES

The belief model estimates $b_t(d) = \mathbb{P}(D_t = d \mid z_{1:t})$ for $d \in \{\mathrm{none}, \mathrm{covariate}, \mathrm{concept}, \mathrm{subgroup}\}$. We learn the evidence likelihoods $p(z \mid d)$ and (optionally) the transition matrix T using synthetic drift episodes generated from the base dataset.

E.2.1 EPISODE STRUCTURE

An episode is a stream of length L with one drift onset at t_0 and a drift intensity schedule $\alpha_t \in [0, 1]$. For each time step,

$$(x_t, y_t) \sim (1 - \alpha_t)\, \mathcal{D}^{\mathrm{ref}} + \alpha_t\, \mathcal{D}^{\mathrm{shift}}. \tag{111}$$

In this equation, $\mathcal{D}^{\mathrm{ref}}$ is the nominal distribution and $\mathcal{D}^{\mathrm{shift}}$ is a constructed shifted distribution.

We generate three patterns:

$$\alpha_t \;=\; \mathbb{I}(t \geq t_0), \tag{112}$$

$$\alpha_t \;=\; \frac{1}{1 + \exp(-\rho(t - t_0))}, \tag{113}$$

$$\alpha_t \;=\; \frac{1}{2}\Big(1 + \sin(2\pi t / P)\Big), \tag{114}$$

where t_0 is drift onset, ρ is the gradual-drift slope, and P is the period for recurring drift.

E.2.2 COVARIATE DRIFT CONSTRUCTION

Covariate drift changes $p(x)$ while holding $p(y \mid x)$ approximately stable. For vision tasks, we apply transformations T_{cov} such as corruption severity, style shifts, blur/noise, color jitter, or background changes:

$$x^{\mathrm{shift}} \;=\; T_{\mathrm{cov}}(x;\, \alpha_t), \qquad y^{\mathrm{shift}} \;=\; y. \tag{115}$$

In this equation, $T_{\mathrm{cov}}(\cdot;\, \alpha_t)$ is a drift-intensity-controlled transformation and labels remain unchanged.

For datasets with discrete domains (e.g., DomainNet), we define $\mathcal{D}^{\mathrm{shift}}$ by swapping the domain environment while preserving the same label space.

E.2.3 CONCEPT DRIFT CONSTRUCTION

Concept drift changes $p(y \mid x)$. We implement a class-conditional label remapping that varies over time. Let π_t be a permutation over classes, and define

$$y^{\mathrm{shift}} \;=\; \pi_t(y), \qquad x^{\mathrm{shift}} \;=\; x. \tag{116}$$

In this equation, x is unchanged but the mapping from inputs to labels is altered.

To model gradual concept drift, we mix remapped and original labels:

$$y_t \;=\; \begin{cases} y, & \text{with probability } 1 - \alpha_t, \\ \pi(y), & \text{with probability } \alpha_t, \end{cases} \tag{117}$$

where α_t follows one of the schedules above. This produces a controlled change in class-conditional structure and calibration behavior.

E.2.4 SUBGROUP DRIFT CONSTRUCTION

Subgroup drift changes behavior only for a subset. Let $g(x) \in \{0, 1\}$ denote membership in a drifted slice, with $\mathbb{P}(g(x) = 1) = \rho_g$ under nominal conditions. We apply a covariate or concept transformation only to the subgroup:

$$x^{\text{shift}} \;=\; \begin{cases} T_{\text{sub}}(x), & g(x) = 1, \\ x, & g(x) = 0, \end{cases} \tag{118}$$

and optionally

$$y^{\text{shift}} \;=\; \begin{cases} \pi(y), & g(x) = 1, \\ y, & g(x) = 0. \end{cases} \tag{119}$$

In these equations, only the subgroup experiences the shift, enabling training of slice-aware monitors.

E.2.5 FITTING LIKELIHOODS AND TRANSITIONS

For each synthetic episode, we compute monitor vectors z_t and record the true drift type label D_t. We fit a multinomial likelihood model with diagonal covariance:

$$p(z \mid D = d) \;=\; \mathcal{N}\big(z;\, \mu_d,\, \text{diag}(\sigma_d^2)\big)\,, \tag{120}$$

where μ_d and σ_d^2 are estimated by maximum likelihood over training samples for drift type d.

We estimate the transition matrix from episode counts:

$$T_{d'd} \;=\; \frac{\sum_{\text{episodes}} \sum_t \mathbb{I}(D_{t-1} = d',\, D_t = d)}{\sum_{\text{episodes}} \sum_t \mathbb{I}(D_{t-1} = d')}. \tag{121}$$

In this equation, the numerator counts transitions from d' to d, and the denominator normalizes by the total transitions out of d'.

Discriminative evidence model and its integration with the filter. The Bayesian update requires an evidence likelihood $p(z_t \mid D_t = d)$. Rather than fitting a fully generative density in $\mathbb{R}^m$, we fit a discriminative model $q_\phi(d \mid z_t)$ via multinomial logistic regression on synthetic drift data. Concretely,

$$q_\phi(d \mid z) \;=\; \frac{\exp(w_d^\top z + b_d)}{\sum_{d' \in \mathcal{D}} \exp(w_{d'}^\top z + b_{d'})}, \tag{122}$$

where $w_d \in \mathbb{R}^m$ and $b_d \in \mathbb{R}$ are learned parameters and m is the number of monitors. We use $q_\phi(d \mid z_t)$ as a *proxy emission potential* in the filtering step:

$$\psi_t(d) \;=\; \left(q_\phi(d \mid z_t)\right)^\beta, \tag{123}$$

where $\beta > 0$ is a temperature that controls evidence sharpness (we set $\beta = 1$ in all experiments). The belief update becomes

$$\tilde{b}_t(d) \;=\; \sum_{d' \in \mathcal{D}} T_{d'd}\, b_{t-1}(d'), \qquad b_t(d) \;=\; \frac{\psi_t(d)\, \tilde{b}_t(d)}{\sum_{d'' \in \mathcal{D}} \psi_t(d'')\, \tilde{b}_t(d'')}. \tag{124}$$

In Eq. equation 124, $\tilde{b}_t(d)$ is the predicted belief under the transition matrix T, and $\psi_t(d)$ plays the same role as the likelihood $p(z_t \mid D_t = d)$ up to a z_t-dependent normalization constant. Because the update renormalizes over d, any constant factor shared across drift types cancels, making the discriminative potential $\psi_t(d)$ sufficient for reproducible filtering.

E.3 SYNTHETIC DRIFT GENERATION FOR BELIEF TRAINING AND GAIN CALIBRATION

We generate synthetic drift episodes by applying controlled transformations to nominal streams to induce each drift type $d \in \{\text{covariate}, \text{concept}, \text{subgroup}\}$. Each episode consists of a pre-drift segment drawn from the nominal environment and a post-drift segment modified by a transformation. We generate $M = 200$ episodes per dataset, balanced across drift types and drift intensities.

Camelyon17 (histopathology). Covariate drift applies stain and scanner perturbations using (i) color jitter (brightness/contrast/saturation in ± 0.2), (ii) Gaussian blur with $\sigma \in [0.5, 1.2]$, and (iii) H&E stain normalization perturbation (random perturbation of stain concentration vectors). Concept drift is induced by label noise localized to the drift segment: we flip labels with probability $p \in \{0.10, 0.20\}$ to emulate changing annotation criteria. Subgroup drift targets a minority slice defined by low-tissue-content patches; only that slice receives severe stain perturbations (same operators but doubled magnitude).

DomainNet (multi-domain objects). Covariate drift is instantiated by switching input style operators within a domain: random JPEG compression (quality in $[20, 50]$), random grayscale with prob. 0.3, and edge enhancement. Concept drift is induced by remapping a subset of classes: we permute labels within a semantically related superclass group (e.g., within `vehicle` or `instrument`) using a fixed permutation π applied only after drift onset. Subgroup drift targets a minority cluster discovered by k-means in embedding space; only that cluster is transformed into a different style via strong edge-only rendering.

SyntheticDrift-CIFAR. Covariate drift uses CIFAR-C corruption types (e.g., blur, noise, brightness) with severity $\in \{3, 5\}$. Concept drift uses a class-conditional permutation π applied gradually by α_t. Subgroup drift applies the corruption only to a minority slice (10%) defined by a fixed set of classes or a cluster in embedding space.

E.4 Dataset-specific synthetic drift transformations

Training the belief model and calibrating the gain table require synthetic drift episodes with known drift-type labels. This section specifies the exact transformations used for each benchmark, including the covariate transform T_{cov}, subgroup transform T_{sub}, and the concept permutation schedule π_t.

E.4.1 General episode construction

Each synthetic episode has length L and a drift onset time t_0. We sample a drift type $d \in \{\text{covariate}, \text{concept}, \text{subgroup}\}$ uniformly and generate a time-varying drift intensity $\alpha_t \in [0, 1]$ using the schedules in Eqs. (42)–(44). For each time step t, we draw a base example (x_t, y_t) from the nominal stream distribution and apply a drift operator that depends on d.

E.4.2 Covariate drift: T_{cov}

For covariate drift, we transform inputs only:

$$x'_t \;=\; T_{\text{cov}}(x_t; \alpha_t), \qquad y'_t \;=\; y_t. \tag{125}$$

Here α_t controls drift intensity. We specify T_{cov} per dataset below.

Camelyon17 (WILDS). Camelyon17 inputs are histopathology patches. We use a stain-style covariate transform implemented as a convex combination between the original image and a fixed stain-perturbed target style:

$$T_{\text{cov}}(x; \alpha) \;=\; (1 - \alpha)\, x \;+\; \alpha\, \text{StainAug}(x; \kappa_{\text{stain}}), \tag{126}$$

where $\text{StainAug}(\cdot; \kappa_{\text{stain}})$ applies HED color-space perturbations and gamma/brightness jitter with strength $\kappa_{\text{stain}} = 0.6$. This models site-specific acquisition shifts without changing class semantics.

DomainNet. DomainNet already contains discrete domains. For synthetic covariate drift within a domain, we use a corruption-based transform:

$$T_{\text{cov}}(x; \alpha) \;=\; \text{Corrupt}(x; s(\alpha)), \tag{127}$$

where $\text{Corrupt}(\cdot; s)$ is a standard image corruption operator (Gaussian blur or shot noise chosen uniformly) with severity $s(\alpha) = \lceil 5\alpha \rceil \in \{0, \ldots, 5\}$.

SyntheticDrift-CIFAR. We use CIFAR-10-C style corruptions:

$$T_{\text{cov}}(x; \alpha) \;=\; \text{CIFARCorrupt}(x; s(\alpha)), \tag{128}$$

with severity $s(\alpha) = \lceil 5\alpha \rceil$ and corruption type fixed per episode (Gaussian noise, motion blur, or brightness).

E.4.3 CONCEPT DRIFT: LABEL PERMUTATION π_t

For concept drift, we keep inputs unchanged but modify the conditional mapping by permuting labels after onset:

$$x'_t \;=\; x_t, \qquad y'_t \;=\; \pi_t(y_t). \tag{129}$$

We sample a class permutation $\pi^\star$ uniformly from all permutations of the label set that move at least two classes. We then apply it with intensity α_t using a mixture between identity and $\pi^\star$:

$$\pi_t(y) \;=\; \begin{cases} \pi^\star(y), & \text{with probability } \alpha_t, \\ y, & \text{with probability } 1-\alpha_t. \end{cases} \tag{130}$$

In Eq. equation 130, α_t controls how frequently the permuted labeling rule is active, producing gradual concept shift when α_t ramps and sudden concept shift when α_t jumps. This construction is used for DomainNet and SyntheticDrift-CIFAR. For Camelyon17, where label semantics are binary tumor/non-tumor and permutation is not meaningful, we implement concept drift as class-conditional prior shift by flipping labels with probability α_t on a fixed subset of patches near the decision boundary (identified by low confidence under the nominal model).

E.4.4 SUBGROUP DRIFT: T_{sub} APPLIED TO A MINORITY SLICE

For subgroup drift, we define a slice indicator $g(x) \in \{0,1\}$ with $\mathbb{P}(g(x)=1) = p_{\text{sub}}$ under nominal conditions (we set $p_{\text{sub}} = 0.15$). We apply a subgroup-only covariate transform:

$$x'_t \;=\; \begin{cases} T_{\text{sub}}(x_t;\alpha_t), & \text{if } g(x_t)=1, \\ x_t, & \text{otherwise,} \end{cases} \qquad y'_t \;=\; y_t. \tag{131}$$

Camelyon17 (WILDS). We define $g(x) = 1$ for a fixed subset of patches selected by tissue-type clustering in embedding space (minority cluster). We set

$$T_{\text{sub}}(x;\alpha) \;=\; (1-\alpha)\,x \;+\; \alpha\,\text{StainAug}(x;\kappa_{\text{sub}}), \tag{132}$$

with stronger stain shift $\kappa_{\text{sub}} = 0.9$ than the global covariate transform.

DomainNet. We define $g(x) = 1$ as one of the object categories (minority category) sampled per episode. We set

$$T_{\text{sub}}(x;\alpha) \;=\; \text{Corrupt}(x;s(\alpha)), \tag{133}$$

using the same corruption family but applied only to the subgroup.

SyntheticDrift-CIFAR. We define $g(x) = 1$ as images from two fixed classes (e.g., classes 0 and 1), chosen once per episode. We apply a style-like transform implemented as color quantization and contrast shift with severity $s(\alpha) = \lceil 5\alpha \rceil$.

E.4.5 REPRODUCIBILITY SETTINGS

For each dataset we generate $M = 200$ episodes of length $L = 1500$ with drift onset t_0 sampled uniformly from $[300, 600]$, and we use 5 random seeds per dataset. All transforms are applied on-the-fly with fixed transform parameters and seed-controlled randomness. The same generators are used for belief-model training and for gain-table calibration.

E.5 CONTROLLER CONFIGURATION AND OPERATIONAL CONSTRAINTS

This section specifies hyperparameters and operational logic used by Algorithm 1.

E.5.1 WINDOW SIZES AND BUFFERS

We set the monitor window length n and certificate window length N as

$$n \;=\; 256, \qquad N \;=\; 1024. \tag{134}$$

In these equations, n controls responsiveness of drift monitors and N controls the regime represented by the risk certificate.

The reference buffer $\mathcal{B}_{\text{ref}}$ is formed from a clean initial deployment segment of length n_{ref}:

$$|\mathcal{B}_{\text{ref}}| \;=\; n_{\text{ref}} \;=\; 2048. \tag{135}$$

In this equation, n_{ref} is chosen to provide stable reference statistics.

E.5.2 LABEL DELAY AND AUDITING BUDGET

Requested labels arrive after a fixed delay d:

$$o_i \;=\; i + d, \qquad d \;=\; 50. \tag{136}$$

In this equation, o_i is the observation time for label y_i.

The labeling budget is

$$B_{\mathrm{lab}} \;=\; 3000, \tag{137}$$

where B_{lab} is the total number of labels that may be requested over a stream.

E.5.3 COOLDOWNS AND FEASIBILITY

Retraining and rollback are rate-limited by cooldowns:

$$\Delta_{\mathrm{rt}} \;=\; 800, \qquad \Delta_{\mathrm{rb}} \;=\; 400. \tag{138}$$

In these equations, Δ_{rt} and Δ_{rb} are time steps during which A_4 or A_5 cannot be executed again.

At time t, the feasible action set is

$$\mathcal{A}_t^{\mathrm{feas}} \;=\; \{a \in \mathcal{A} :\; \mathrm{Cool}(a,t) = 1,\; \mathrm{Bud}(a,t) = 1\}\,, \tag{139}$$

where $\mathrm{Cool}(a,t)$ enforces cooldown rules and $\mathrm{Bud}(a,t)$ enforces remaining budget rules.

For retrain feasibility:

$$\mathrm{Cool}(A_4,t) = \mathbb{I}\big(t - t^{\mathrm{rt}}_{\mathrm{last}} \geq \Delta_{\mathrm{rt}}\big)\,, \tag{140}$$

and for rollback:

$$\mathrm{Cool}(A_5,t) = \mathbb{I}\big(t - t^{\mathrm{rb}}_{\mathrm{last}} \geq \Delta_{\mathrm{rb}}\big)\,. \tag{141}$$

In these equations, $t^{\mathrm{rt}}_{\mathrm{last}}$ and $t^{\mathrm{rb}}_{\mathrm{last}}$ record the last execution time of retrain and rollback.

E.5.4 QUERY POLICY FOR CERTIFICATE LABELS

The controller selects a query size k_t as a function of drift evidence and certificate uncertainty. Let the current certificate be

$$U_t(\delta) \;=\; \widehat{R}_t + \mathrm{rad}(n_t, \delta_t). \tag{142}$$

Define a safety margin

$$m_t \;=\; \tau - U_t(\delta). \tag{143}$$

In this equation, $m_t > 0$ indicates certified safety slack, while $m_t < 0$ indicates a certificate violation.

We set k_t using a piecewise rule:

$$k_t \;=\; \begin{cases} k_{\max}, & U_t(\delta) > \tau, \\ k_{\mathrm{high}}, & m_t \leq m_{\mathrm{low}} \text{ or } \max_j \widetilde{z}_{t,j} \geq \zeta, \\ k_{\mathrm{low}}, & \text{otherwise}, \end{cases} \tag{144}$$

with

$$k_{\max} = 64, \qquad k_{\mathrm{high}} = 32, \qquad k_{\mathrm{low}} = 8, \tag{145}$$

and thresholds $m_{\mathrm{low}} = 0.02$ and $\zeta = 2.0$. In these equations, $k_{\max}$ is used during potential unsafe regimes, k_{high} is used when evidence is strong or safety margin is small, and k_{low} is used during stable operation.

E.6 GAIN TABLE CALIBRATION AND VALUES

The controller selects actions via a belief-weighted gain table $G(d,a)$, which encodes the expected benefit of taking action $a \in \mathcal{A}$ under drift type $d \in \mathcal{D}$. This appendix specifies (i) the exact definition of "gain" used in our experiments, (ii) the offline calibration protocol, and (iii) the full numeric gain table values.

E.6.1 DEFINITION OF GAIN AS WINDOWED RISK REDUCTION

Gain is defined in terms of reduction in certified windowed risk after applying an intervention. Let R_t denote the windowed risk evaluated on ground-truth labels in simulation, and let $R_t^{(a)}$ denote the counterfactual risk trajectory obtained by applying action a at time t and then executing the fixed deployment loop for a short horizon of length H (including the same monitoring windows, label delay model, and update rules). For each drift type d, we estimate the expected risk reduction at horizon H:

$$\Delta_R(d, a) \;=\; \mathbb{E}\Big[R_{t+H}^{\text{base}} \;-\; R_{t+H}^{(a)} \;\big|\; D_t = d\Big], \tag{146}$$

where R_{t+H}^{base} is the risk at $t + H$ under the baseline "no-op" continuation (A_0), $R_{t+H}^{(a)}$ is the risk at $t + H$ when applying action a at time t, D_t is the drift-type state, and the expectation is over random drift episode realizations, window sampling, and label delay. We convert risk reduction into a dimensionless gain by normalizing by a fixed risk scale $\sigma_R > 0$:

$$G(d, a) \;=\; \frac{\Delta_R(d, a)}{\sigma_R}. \tag{147}$$

In our experiments we set $\sigma_R = 0.10$ so that $G(d, a) = 1$ corresponds to an average 0.10 absolute reduction in windowed risk at horizon H.

E.6.2 OFFLINE CALIBRATION PROTOCOL

We estimate $G(d, a)$ by generating synthetic drift episodes in each benchmark stream and evaluating counterfactual one-step interventions. For each dataset, we construct M synthetic drift episodes of length L by sampling a drift type $d \in \mathcal{D}$, sampling a drift onset time t_0, and applying the corresponding drift generator. We then roll out the deployed model to time $t = t_0 + \Delta$ (with Δ sampled uniformly in a short window after onset), and compute R_{t+H}^{base} and $R_{t+H}^{(a)}$ for each candidate action $a \in \mathcal{A}$ using the exact update rules used at test time (calibrator update for A_1, adaptation steps for A_2, retrain trigger for A_4, rollback for A_5, and abstention policy for A_6). All counterfactuals share the same drift realization, label delay, and sampling RNG state so that differences are attributable to the action. We set $M = 200$ episodes, $L = 1500$ steps, $H = 150$ steps, and use 5 random seeds per dataset; reported gains average across seeds.

We emphasize that gains are calibrated once offline and then fixed for all streaming evaluations. The gain table does not use test labels from the evaluation stream; it is computed on separate synthetic episodes built from the same dataset distributions and augmentation generators.

E.6.3 NUMERIC GAIN TABLES USED IN EXPERIMENTS

Table 5 reports the exact gain values used in all experiments. Values reflect average risk reduction at horizon $H = 150$ normalized by $\sigma_R = 0.10$ (Eq. equation 147). Higher values indicate larger expected improvement. Entries for actions that are not applicable to a drift type (e.g., recalibration under pure covariate shift when labels are not yet available) are set to small positive values reflecting modest indirect benefits (e.g., stabilizing confidence), which we observed empirically in the calibration rollouts.

Table 5: **Gain table $G(d, a)$ used by the controller.** Gains are computed by offline synthetic-drift calibration as normalized windowed risk reduction (Eqs. equation 146–equation 147) at horizon $H = 150$ with normalization $\sigma_R = 0.10$. Rows correspond to drift types $d \in \{\text{none}, \text{covariate}, \text{concept}, \text{subgroup}\}$ and columns correspond to actions $a \in \{A_0, \ldots, A_6\}$.

Drift type d	A_0	A_1 (recalib.)	A_2 (TTA)	A_3 (query)	A_4 (retrain)	A_5 (rollback)	A_6 (abstain)
none	0.00	0.10	0.05	0.08	0.12	0.10	0.15
covariate	0.00	0.35	0.70	0.25	0.85	0.40	0.55
concept	0.00	0.20	0.30	0.75	1.05	0.60	0.65
subgroup	0.00	0.25	0.35	0.85	0.95	0.55	0.80

Finally, we found that controller behavior is robust to moderate rescaling of $G(d, a)$: multiplying all gains by a constant does not change the argmax in Eq. (25), and small perturbations preserve the preferred action ordering under each drift type.

E.6.4 ESCALATION LOGIC

Safety gating is applied using the certificate threshold τ:

$$U_t(\delta) \le \tau \;\Rightarrow\; a_t \in \{A_0, A_1, A_2, A_3\}, \qquad U_t(\delta) > \tau \;\Rightarrow\; a_t = A_6 \text{ and schedule } \{A_5, A_4\}. \tag{148}$$

In this rule, A_6 is an immediate protective mode, while $\{A_5, A_4\}$ are heavy interventions executed when cooldown constraints allow.

We define a deterministic heavy-action priority:

$$\text{If } U_t(\delta) > \tau: \quad \begin{cases} \text{rollback if feasible,} \\ \text{else retrain if feasible,} \\ \text{else continue fallback and increase } k_t. \end{cases} \tag{149}$$

This ordering is used to prefer a known-safe checkpoint when available, while still enabling adaptation through retraining when rollback is not feasible.

E.6.5 RECALIBRATION AND TEST-TIME ADAPTATION DETAILS

Recalibration uses temperature scaling with parameter $T > 0$ applied to logits:

$$p_{\theta,T}(y \mid x) \;=\; \mathrm{softmax}\!\left(\frac{h_\theta(x)}{T}\right), \tag{150}$$

where T is optimized on the labeled set $\mathcal{S}_t$ by minimizing negative log-likelihood.

Test-time adaptation updates θ using entropy minimization on recent unlabeled inputs:

$$\theta \leftarrow \theta - \eta_{\mathrm{tta}} \nabla_\theta \left(\frac{1}{|\mathcal{B}_t|} \sum_{x \in \mathcal{B}_t} H_\theta(x) \right), \tag{151}$$

where η_{tta} is a learning rate and $H_\theta(x)$ is predictive entropy. We use $s = 5$ gradient steps when executing A_2.

E.7 UTILITY FUNCTION AND SAFETY-VIOLATION PENALTY

The controller ranks feasible actions,

$$\mathcal{U}_t(a) \;=\; \Delta_t(a) \;-\; \lambda\, c(a) \;-\; \gamma\, \mathrm{Viol}_t(a), \tag{152}$$

where $\Delta_t(a)$ is the belief-weighted gain, $c(a)$ is the operational cost, and $\mathrm{Viol}_t(a)$ penalizes actions that would lead to uncertified operation. We now define $\mathrm{Viol}_t(a)$ explicitly.

Let $U_t(\delta)$ be the current certificate value computed in Section 4, and let τ be the safety target. For each candidate action a at time t, we form a one-step-ahead conservative prediction of the post-action certificate,

$$\widehat{U}_t^{(a)}(\delta) \;=\; U_t(\delta) \;-\; \widehat{\Delta U}_t(a), \tag{153}$$

where $\widehat{\Delta U}_t(a)$ is an action-dependent expected reduction in the certificate bound. We set

$$\widehat{\Delta U}_t(a) \;=\; \sigma_U \cdot \Delta_t(a), \tag{154}$$

with $\Delta_t(a) = \sum_{d \in \mathcal{D}} b_t(d)\, G(d, a)$ and a fixed scale $\sigma_U > 0$ that converts gain units into certificate units. In all experiments we set $\sigma_U = 0.10$, so $\Delta_t(a) = 1$ corresponds to a predicted 0.10 absolute reduction in the certificate.

The safety-violation penalty is then defined as the (hinge) excess of the predicted certificate above the target:

$$\mathrm{Viol}_t(a) \;=\; \max\!\Big\{0,\; \widehat{U}_t^{(a)}(\delta) - \tau\Big\}. \tag{155}$$

In this definition, $\mathrm{Viol}_t(a) = 0$ when action a is predicted to keep the system certified safe ($\widehat{U}_t^{(a)}(\delta) \le \tau$), and $\mathrm{Viol}_t(a)$ increases linearly with the margin by which safety would be violated.

We set the penalty weight to

$$\gamma \;=\; 50, \tag{156}$$

which makes uncertified operation strongly dominated unless it yields substantial expected gains. This value is fixed across all datasets and streams.

Table 6: **Ablations on the full streaming protocol.** Metrics match the main paper: cumulative cost C_{tot}, safety violations V, detection delay T_{det}, recovery time T_{rec}, and minimum worst-group accuracy $\min \mathrm{Acc}^{\mathrm{wg}}$. The last column reports the false-intervention rate (FIR): the fraction of time steps in which the method intervenes ($a_t \neq A_0$) while the deployed risk is already below the safety target.

Method variant	C_{tot}	V	T_{det}	T_{rec}	$\min \mathrm{Acc}^{\mathrm{wg}}$	FIR
Camelyon17 stream						
Alarm-only	3.2(3)	46(4)	22(2)	210(18)	0.71(2)	0.05(1)
Adapt-always (TTA)	58.0(21)	3(1)	18(2)	74(6)	0.79(1)	0.82(3)
Retrain-on-schedule	41.5(18)	9(2)	35(3)	96(9)	0.77(1)	0.21(3)
Selective prediction only	9.8(6)	14(3)	24(2)	160(14)	0.74(2)	0.14(2)
Controller (no certificate)	18.7(11)	7(2)	19(2)	98(8)	0.78(1)	0.16(2)
Ours (full)	24.6(14)	0	20(2)	62(5)	0.81(1)	0.10(2)
DomainNet stream						
Alarm-only	4.1(4)	63(5)	28(3)	260(21)	0.49(2)	0.06(1)
Adapt-always (TTA)	64.0(25)	6(2)	21(2)	110(10)	0.58(2)	0.85(3)
Retrain-on-schedule	45.0(20)	14(3)	40(4)	132(11)	0.56(2)	0.23(3)
Selective prediction only	12.4(9)	27(4)	30(3)	205(17)	0.52(2)	0.18(2)
Controller (no certificate)	22.5(13)	11(3)	23(2)	141(12)	0.57(2)	0.19(2)
Ours (full)	29.8(16)	1(1)	24(2)	88(7)	0.61(1)	0.13(2)
SyntheticDrift-CIFAR stream						
Alarm-only	2.6(2)	52(4)	16(2)	185(16)	0.68(2)	0.04(1)
Adapt-always (TTA)	52.0(20)	4(1)	13(2)	66(5)	0.79(1)	0.80(3)
Retrain-on-schedule	36.0(17)	10(2)	25(3)	92(8)	0.77(1)	0.20(3)
Selective prediction only	8.6(6)	19(3)	18(2)	148(12)	0.72(2)	0.15(2)
Controller (no certificate)	16.8(10)	8(2)	14(2)	84(7)	0.78(1)	0.17(2)
Ours (full)	21.2(12)	0	15(2)	58(5)	0.82(1)	0.09(2)

F Extra experiments

This appendix provides additional empirical evidence that each component of the proposed system matters, and that the safety–cost advantages observed in Table 1 and Figures 2a–2b persist across operational settings. We emphasize interpretability and operational diagnostics: each subsection reports a compact set of tables (kept intentionally larger for readability) followed by detailed analysis of what the results mean in deployment terms.

F.1 Ablations that isolate each component

We test ablations that remove one design choice at a time while keeping the same action space, cost model, and streaming protocol (Table 3). The goal is to show which ingredients drive safety, which drive efficiency, and which prevent unnecessary intervention.

Interpretation. Table 6 highlights three practical takeaways.

First, **certification is what prevents unsafe operation under delayed labels**. The controller without certificate has substantially more safety violations (V increases from 0 to 7 on Camelyon17 and from 0 to 8 on SyntheticDrift-CIFAR). In streaming deployments, drift evidence alone cannot prevent long unsafe stretches because the system can mistakenly "believe" it has recovered while risk remains elevated. The certificate acts as a fail-safe: it forces fallback and escalation whenever verified risk is not under control.

Second, **adaptation-only strategies buy speed at a high operational price**. Adapt-always is consistently among the fastest to recover, but its cost is extreme and its FIR is close to one: it intervenes almost continuously even during stable periods. This is the operational signature of methods that treat every input as a potential drift signal. In production, such behavior is often

unacceptable due to compute, latency, and governance constraints, even when safety outcomes are good.

Third, **the full method reduces safety violations while remaining selective about interventions**. Compared to retrain-on-schedule and selective-prediction-only baselines, the proposed system improves worst-group robustness and recovery time while keeping FIR low. Operationally, this means the controller is not "trigger-happy": it intervenes primarily when needed, and it can justify interventions with both drift evidence and certified risk.

F.2 Component-specific ablations of Drift2Act

To isolate how the *internal* pieces contribute, we keep the full streaming protocol but modify one internal module at a time.

Table 7: **Internal component ablations of Drift2Act (same controller interface).** Each row removes one internal module: belief (replace with single drift-score thresholds), active labeling (replace with fixed label rate), certificate gating (do not force fallback when risk may be unsafe), and monitor stack (use MMD-only rather than multi-monitor).

Variant	C_{tot}	V	T_{det}	T_{rec}	$\min \mathrm{Acc}^{\mathrm{wg}}$	FIR
Camelyon17 stream						
Drift2Act (full)	24.6(14)	0	20(2)	62(5)	0.81(1)	0.10(2)
No belief (single score)	23.3(13)	4(2)	21(2)	90(8)	0.78(1)	0.17(2)
No active labeling (fixed rate)	24.1(13)	2(1)	20(2)	76(7)	0.80(1)	0.21(3)
No certificate gating	23.7(12)	6(2)	20(2)	88(8)	0.79(1)	0.12(2)
MMD-only monitors	25.0(14)	3(2)	24(3)	83(8)	0.79(1)	0.16(2)
DomainNet stream						
Drift2Act (full)	29.8(16)	1(1)	24(2)	88(7)	0.61(1)	0.13(2)
No belief (single score)	28.5(15)	6(2)	25(2)	128(11)	0.58(2)	0.19(2)
No active labeling (fixed rate)	29.2(16)	3(2)	24(2)	106(9)	0.60(1)	0.24(3)
No certificate gating	28.9(15)	9(3)	24(2)	142(12)	0.59(1)	0.14(2)
MMD-only monitors	30.5(17)	5(2)	29(3)	121(10)	0.58(2)	0.18(2)

Interpretation. Table 7 explains *why* the full system looks strong in the main paper.

Removing belief collapses drift understanding into a single score; the controller then cannot distinguish when to use inexpensive fixes (recalibration/TTA) versus when to acquire labels and retrain. The result is a systematic pattern: longer recovery (T_{rec} increases sharply) and more violations (V increases), because actions become mismatched to the underlying drift mechanism.

Removing active labeling spreads audits uniformly across time. In stable segments this wastes budget (FIR increases), while during drift the certificate tightens too slowly, leaving longer unsafe windows. This is exactly the operational failure that motivates active verification: labels are most valuable when uncertainty is high and risk is near the safety threshold.

Removing certificate gating is the clearest safety regression. Even with a good belief model, the controller will sometimes choose actions that *look* beneficial under drift evidence but do not immediately restore safety under label delay. Without the gating rule, the system can keep predicting in that "gray zone," driving violations upward.

Finally, using only MMD reduces the breadth of drift evidence. On DomainNet, MMD-only delays detection and increases violations because some shifts manifest more strongly in uncertainty and calibration than in pure representation discrepancy. The multi-monitor design is therefore not an aesthetic choice; it is what stabilizes decisions across heterogeneous drifts.

F.3 False alarms and unnecessary interventions

We report how often methods intervene when no intervention was needed. We compute the false-intervention rate (FIR) on time steps where the deployed risk is already below the safety target,

and we also report a heavy false-intervention rate restricted to retrain/rollback actions (which are the most disruptive operationally).

Table 8: **False-intervention diagnostics.** FIR measures unnecessary interventions during already-safe operation; Heavy-FIR counts unnecessary retrain/rollback events. The certified controller remains selective, while adapt-always exhibits high unnecessary intervention by design.

Method	Camelyon17		DomainNet		SyntheticDrift-CIFAR	
	FIR	Heavy-FIR	FIR	Heavy-FIR	FIR	Heavy-FIR
Alarm-only	0.05(1)	0.00	0.06(1)	0.00	0.04(1)	0.00
Adapt-always (TTA)	0.82(3)	0.00	0.85(3)	0.00	0.80(3)	0.00
Retrain-on-schedule	0.21(3)	0.05(1)	0.23(3)	0.06(1)	0.20(3)	0.05(1)
Selective prediction only	0.14(2)	0.00	0.18(2)	0.00	0.15(2)	0.00
Controller (no certificate)	0.16(2)	0.02(1)	0.19(2)	0.03(1)	0.17(2)	0.02(1)
Controller + certificate (ours)	0.10(2)	0.01(1)	0.13(2)	0.02(1)	0.09(2)	0.01(1)

Interpretation. Table 8 shows that the certificate reduces unnecessary churn in two ways. First, it prevents "panic actions" triggered by noisy monitors during stable periods, because the controller can demand a small number of verification labels and confirm that risk is still under τ. Second, it reduces heavy interventions in safe periods by making rollback/retrain conditional on certified risk rather than raw drift evidence. In contrast, schedule-based retraining produces heavy false interventions by construction: it retrains even when the stream is stable, which increases Heavy-FIR and inflates cost.

F.4 Sensitivity to label delay and labeling budget

We vary label delay d and labeling budget B_{lab} to simulate different operational environments. Longer delays slow down verification, while smaller budgets restrict auditing and reduce the controller's ability to tighten certificates during drift.

Table 9: **Sensitivity to label delay d (Camelyon17)** Increasing delay slows recovery because verification arrives later, which delays safe exit from fallback and slows escalation decisions.

Delay d	C_{tot}	V	T_{det}	T_{rec}
0	23.1(12)	0	20(2)	54(5)
25	23.8(13)	0	20(2)	58(5)
50	24.6(14)	0	20(2)	62(5)
100	26.2(16)	1(1)	21(2)	78(7)

Interpretation. Table 9 matches operational intuition. When d increases, the controller receives confirmation of current performance later, so it remains longer in conservative modes (fallback and higher auditing). This increases cost and recovery time. Importantly, violations remain controlled: even at $d = 100$, unsafe operation remains rare because the system reacts conservatively when verification is delayed.

Table 10: **Sensitivity to labeling budget B_{lab} (Camelyon17)** Larger budgets tighten certificates faster around drift events, reducing recovery time and violations at moderate added cost.

Budget B_{lab}	C_{tot}	V	T_{det}	T_{rec}
1000	21.4(12)	2(1)	20(2)	86(8)
2000	23.0(13)	1(1)	20(2)	72(7)
3000	24.6(14)	0	20(2)	62(5)
5000	26.8(16)	0	20(2)	55(5)

Interpretation. Table 10 shows that labeling budget primarily affects *how quickly* the system can certify safety and exit fallback after drift. With small budgets, the certificate remains wider for longer, producing longer recovery and occasional violations. With larger budgets, the controller can concentrate audits when needed, tighten the bound quickly, and recover faster. The key point is

that the controller uses additional labels efficiently: cost increases are mild relative to the gains in recovery and safety.

F.5 More drift shapes: gradual drift and recurring drift

We evaluate two additional drift patterns beyond sudden shift: gradual drift (slowly increasing intensity) and recurring drift (periodic return of a previously seen shift). These settings stress-test whether the controller avoids permanent overreaction.

Table 11: **Gradual and recurring drift (SyntheticDrift-CIFAR)** Gradual drift mainly increases recovery time, while recurring drift mainly tests whether the controller avoids repeated heavy interventions when the stream revisits regimes.

Drift pattern	C_{tot}	V	T_{det}	T_{rec}	FIR	$\min \mathrm{Acc}^{\mathrm{wg}}$
Sudden drift (baseline setting)	21.2(12)	0	15(2)	58(5)	0.09(2)	0.82(1)
Gradual drift (slow ramp)	20.6(12)	0	19(2)	72(6)	0.10(2)	0.81(1)
Gradual drift (fast ramp)	21.5(12)	0	16(2)	63(5)	0.09(2)	0.82(1)
Recurring drift (short period)	22.8(13)	0	14(2)	61(5)	0.11(2)	0.81(1)
Recurring drift (long period)	21.9(12)	0	15(2)	60(5)	0.10(2)	0.82(1)

Interpretation. Table 11 shows that gradual drift is not primarily a detection problem; it is a *verification pacing* problem. Because degradation accumulates slowly, the certificate approaches the threshold more gradually, and the controller increases auditing and corrective actions later, extending recovery time slightly. Recurring drift tests whether controllers "thrash" by retraining repeatedly. The results show stable safety with only mild increases in FIR and cost, indicating that the combination of cooldowns, belief, and certificate gating prevents repeated overreaction when regimes reappear.

F.6 Cost settings: cheap vs expensive labels

We vary the per-label annotation cost while keeping the same labeling budget ceiling and stream. This captures deployments where labels are either cheap (lightweight human verification) or expensive (expert review).

Table 12: **Effect of per-label cost (Camelyon17)** When labels become expensive, the controller shifts toward adaptation and conservative fallback rather than frequent audits, increasing recovery time but preserving safety.

Per-label cost	C_{tot}	V	T_{det}	T_{rec}	FIR	$\min \mathrm{Acc}^{\mathrm{wg}}$
0.01	23.5(13)	0	20(2)	58(5)	0.11(2)	0.81(1)
0.05	24.6(14)	0	20(2)	62(5)	0.10(2)	0.81(1)
0.10	25.2(15)	1(1)	21(2)	70(6)	0.12(2)	0.80(1)

Interpretation. Table 12 shows a predictable and desirable adaptation in controller behavior. When labels are cheap, the controller audits more aggressively, tightens certificates quickly, and exits fallback sooner. When labels are expensive, the controller leans more on recalibration and TTA, and it tolerates longer fallback periods to avoid heavy label spending. Safety remains strong, indicating that the certificate gating continues to enforce conservative operation even when audits are costly.

F.7 Robustness to monitor noise

We test robustness by corrupting monitor values (e.g., due to logging failures, representation drift noise, or unstable uncertainty estimates). We inject moderate noise into the monitor vector and measure how often this causes unnecessary actions.

Interpretation. Table 13 illustrates a key advantage of separating drift evidence from safety verification. Monitor noise increases action frequency and cost because the controller reacts to noisier signals. However, safety degrades slowly because the certificate prevents the controller from

Table 13: **Robustness to monitor noise (Camelyon17)** With noisier monitors, drift evidence becomes less reliable and interventions become slightly more frequent; certificate gating prevents large safety regressions by requiring verification before unsafe operation persists.

Noise level	C_{tot}	V	T_{det}	T_{rec}	FIR	$\min \mathrm{Acc}^{\mathrm{wg}}$
0.00	24.6(14)	0	20(2)	62(5)	0.10(2)	0.81(1)
0.25	25.1(15)	0	21(2)	64(5)	0.12(2)	0.81(1)
0.50	26.0(16)	1(1)	22(2)	70(6)	0.15(3)	0.80(1)
0.75	27.3(17)	2(1)	24(3)	78(7)	0.18(3)	0.79(1)

continuing normal operation in regimes that cannot be verified as safe. In practical terms, monitor noise mostly causes *efficiency loss* rather than catastrophic safety loss, which is the desired failure mode in safety-constrained operation.

F.8 Optional extra benchmark: temporal drift in Amazon Reviews

To test a non-vision stream, we evaluate on a temporal Amazon Reviews stream, where time induces changing user behavior and class balance. We use the same evaluation metrics.

Table 14: **Amazon Reviews temporal stream.** The certified controller maintains low violations and strong recovery while keeping costs moderate compared to always-adapt and schedule-based retraining.

Method	C_{tot}	V	T_{det}	T_{rec}	$\min \mathrm{Acc}^{\mathrm{wg}}$	FIR
Alarm-only	3.0(3)	41(5)	20(2)	180(15)	0.73(1)	0.05(1)
Adapt-always (TTA)	56.0(25)	4(2)	16(2)	70(7)	0.80(1)	0.79(3)
Retrain-on-schedule	38.0(20)	9(3)	28(3)	96(9)	0.78(1)	0.20(2)
Controller (no certificate)	17.9(12)	6(2)	18(2)	88(9)	0.79(1)	0.15(2)
Controller + certificate (ours)	23.5(14)	0	19(2)	60(6)	0.82(1)	0.11(2)

Interpretation. Table 14 shows that the same system design transfers to a temporally evolving text domain. The certified controller attains strong safety and worst-group robustness without resorting to continuous adaptation. The cost remains moderate because the controller spends labels and heavy actions primarily during periods where certified risk indicates genuine danger.

G Discussion, Limitations, Future Work and Broader Impact

Discussion This work proposes a practical reframing of drift monitoring: rather than treating drift as an alarm that operators must interpret, the system maintains a certified view of current risk and converts evidence into cost-aware interventions. The empirical results in Table 1 together with the safety–cost trade-off in Figure 2a indicate that coupling active verification with belief-driven action selection reduces unsafe operation without requiring continuous adaptation. The recovery dynamics in Figure 2b further suggest that safety gating enables rapid escalation when needed while preserving cost-efficiency during stable regimes.

Limitations The primary limitation is that the certificate validity relies on random label sampling from the current evaluation window. Concretely, the guarantee in Theorem 1 requires that the queried indices are drawn uniformly from W_t (or uniformly within strata for slice-level certification). If labels are obtained through heavily biased selection, the empirical risk $\widehat{R}_t$ may no longer represent the true window risk R_t, weakening the bound. In deployments where uniform sampling is difficult, one mitigation is to reserve a small fraction of queries for uniform auditing, using the remaining budget for targeted diagnostics.

A second limitation concerns the choice of window length N and how W_t tracks the evolving regime. Under extreme non-stationarity, a short window reduces lag but increases variance, while a long window stabilizes estimates but can mix regimes and delay detection of rapid changes. Although the controller can partially compensate by increasing label queries when uncertainty is high, selecting N

remains a domain-dependent design choice. A promising direction is adaptive windowing, where N is adjusted online based on evidence stability and the certificate radius.

Finally, our action space is intentionally small to match common production interventions. Richer action sets may further improve performance, including targeted data collection for specific slices, dynamic thresholding for abstention, and model patching via lightweight adapters. Future controllers could replace the belief-weighted gain model with learned policies, such as contextual bandits or reinforcement learning, while retaining the certificate as a safety constraint. Extending the framework to broader tasks, including regression, ranking, and structured prediction, requires task-specific loss definitions and may benefit from task-tailored monitoring signals.

Future Work Several directions can strengthen and broaden the framework. First, the certificate can be extended beyond a single global risk target to provide slice-conditional safety certificates, enabling explicit guarantees for vulnerable subgroups and supporting fairness-aware escalation when subgroup drift is detected. Second, richer action sets could improve recovery without relying on expensive retraining, including lightweight adapters, targeted data collection for specific slices, dynamic abstention thresholds, and retrieval of safe checkpoints conditioned on drift type. Third, the current receding-horizon controller can be replaced by learned policies (contextual bandits or reinforcement learning) that optimize long-run utility under explicit safety constraints, with the certificate serving as a hard guardrail. Fourth, improving robustness to practical monitoring noise remains important: combining representation monitors with causal or graph-based change detectors, and incorporating uncertainty about monitor reliability into the belief state, may reduce unnecessary interventions without sacrificing safety. Finally, applying the same principles to additional tasks—regression, ranking, and structured prediction—requires task-specific risk definitions and certificates, as well as benchmark protocols that reflect the corresponding operational costs and failure modes. Overall, the key message is that reliability under drift is not just a detection problem. By integrating drift evidence, costed interventions, and an explicit online safety certificate, we move from 'detecting drift" to operating models responsibly in non-stationary environments.

Broader impact From a broader impact perspective, certified monitoring can reduce silent failures and improve accountability by producing auditable evidence of when and why interventions were triggered. At the same time, abstain/handoff can increase human workload, and naive deployment may disproportionately route uncertain cases from specific subgroups. For this reason, subgroup monitoring and coverage constraints should be reported alongside overall performance. Overall, this reframes monitoring as decision-making with safety.