Proceedings of Machine Learning Research 266:1–19, 2025 Conformal and Probabilistic Prediction with Applications

A Conformal Martingales Approach for Recurrent Concept Drift

Charalambos Eliades ST009072@STUD.FREDERICK.AC.CY
Computational Intelligence (COIN) Research Lab
Frederick University, Cyprus
Harris Papadopoulos H.PAPADOPOULOS@FREDERICK.AC.CY
Computational Intelligence (COIN) Research Lab
Frederick University, Cyprus

Editor: Khuong An Nguyen, Zhiyuan Luo, Harris Papadopoulos, Tuwe Löfström, Lars Carlsson and Henrik Boström

Abstract

In many Concept Drift scenarios, previously seen data distributions reappear. This type of drift is known as Recurrent Concept Drift and is especially common in environments with seasonality, user-behavior cycles, or regime changes. This work extends our previously proposed Inductive Conformal Martingales(ICM) concept drift approach so that it can reuse earlier models, thus saving computational resources and data. Upon drift detection, the proposed approach selects a model from a pool of all earlier models if (i) ICM fails to reject exchangeability between a recent-data window and the window immediately following that model's training set, and (ii) the model's F1 score on the current window exceeds a threshold derived from its historical performance. It only trains a new model when no stored model satisfies both criteria. Experiments on three public data streams (STAGGER, Airlines and ELEC) cut retraining events by up to 94% and reduce wasted training instances by $22\% - 33\%$, while limiting accuracy loss to less than 3 percentage points relative to always retraining.

Keywords: recurrent concept drift, conformal martingales, online learning

1. Introduction

In many real world data stream classification tasks, the data generating mechanism changes over time, a phenomenon known as *concept drift* (CD). This change might affect classifier performance and, in the conformal prediction framework, violates the exchangeability assumption (EA), causing loss of validity and calibration. Among the various types of CD, we focus on *recurrent concept drift* (RCD), where a previously seen concept reappears. RCD is common in environments with seasonality, user behavior cycles, or regime changes (e.g., network traffic, market states, or industrial sensors) (Suárez-Cetrulo et al., 2023). In these cases, the reappearance of earlier distributions enables the reuse of previously trained models, reducing computational cost and conserving resources.

Formally, CD can be defined by examining how the joint distribution of features and labels evolve over time. Consider a stream $S = \{(x_0, y_0), (x_1, y_1), \dots\}$, where (x_i, y_i) are feature label pairs. If S can be partitioned into $S_{0,t}$ and $S_{t+1,\dots}$, each generated by different distributions, then a CD has occurred at $t + 1$.

CD can be produced from three sources. By the chain rule of probability, we have $f_{X,Y,t} = f_{Y|X,t} \cdot f_{X,t}$ and $f_{X,Y,t+1} = f_{Y|X,t+1} \cdot f_{X,t+1}$, where $f_{X,Y,t}$ is the joint probability density function of a pair (x, y) at time t. A change in the joint distribution at time $t + 1$ may result from one of the following scenarios (Lu et al., 2019):

- **Virtual drift):** $f_{Y|X,t} = f_{Y|X,t+1}$ while $f_{X,t} \neq f_{X,t+1}$. In this case, the conditional distribution of labels given features remains unchanged, but the marginal distribution over features shifts. This form of drift affects the features distribution without altering decision boundaries.

- **Concept shift (actual drift):** $f_{Y|X,t} \neq f_{Y|X,t+1}$ and $f_{Y,t} = f_{Y,t+1}$ here the decision boundaries change and lead to decrease in accuracy, also referred to as actual drift.

- **Combination of the above** Both $f_{Y|X,t} \neq f_{Y|X,t+1}$ and $f_{X,t} \neq f_{X,t+1}$, representing a simultaneous shift in the conditional and marginal distributions. This is the most general form of CD and includes elements of both label and concept shift.

While earlier we categorized drift by the type of change (e.g., real vs. virtual), here we adopt the taxonomy based on how the change occurs over time, which consists of four categories (Lu et al., 2019; Bagui and Jin, 2020):

- **Sudden drift:** Abrupt, sharp changes at a single point in time.

- **Gradual drift:** The data stream contains a mix of examples from both the old and the new concept, but each example is generated entirely by either the old or the new distribution. Over time, the proportion of examples from the new distribution increases until it fully replaces the old one.

- **Incremental drift:** Each individual example is generated by a distribution that gradually changes over time, typically modeled as a mixture of the old and new concepts, with the influence of the new distribution steadily increasing until it dominates.

- **Recurrent drift:** occurs when previously seen distributions reappear over time. In this scenario, the underlying data distribution changes, but eventually returns to a previously seen state. Importantly, the transitions between distributions may occur in any of the above ways (sudden, gradual, or incremental), but what distinguishes recurrent drift is the reappearance of a previously observed concept. This scenario is particularly relevant in domains with seasonal, cyclic, or repetitive behavior, such as network intrusion detection or electricity consumption. Our study focuses on this type of drift, aiming not only to detect changes but also to recognize when a known distribution has reoccurred.

While RCD can, in principle, be addressed using general CD techniques, methods specifically designed for RCD offer distinct advantages. By taking into account the reappearance of previously seen distributions, these methods can avoid unnecessary retraining and enable faster, more efficient adaptation. According to Suárez-Cetrulo et al. (2023), approaches to handling RCD can be grouped into three broad categories:

The first category is **Supervised Learning Approaches**. These methods rely on labeled data and maintain a set of base models, each trained on previously observed concepts. Upon detecting CD, the system must decide whether to reuse an existing model or train a new one, based on its performance on recent data. The most common instantiation of this strategy involves ensemble learning, where multiple models are maintained and their

weights are adjusted dynamically. Typically, the decision to reuse a model is based on recent classification performance, rather than explicitly detecting the recurrence of a concept.

In particular, passive ensembles operate without explicit drift detection mechanisms; they update continuously and adjust model weights based on recent or accumulated predictive performance. In contrast, active ensembles incorporate drift detectors to identify significant changes and reset or replace underperforming models accordingly.

The second category consists of **Meta Learning Approaches**. These approaches collect statistics over time to detect changes in the data distribution or classifier performance. Their primary objectives are: (i) to anticipate when the next CD is likely to occur, and (ii) to predict which previously encountered concept is most likely to reappear, thereby enabling faster adaptation.

To achieve this, the meta-learner manages a pool of base models, each typically trained on a different concept. Rather than relying on ensemble voting, the meta-learner focuses on selecting the most suitable model for the current situation, often based on auxiliary signals such as time, environmental variables, or distributional trends, rather than only recent classification performance. It may load a single model, adapt or fine tune an existing one, or, in some cases, combine multiple models though model selection remains its central mechanism.

Notably, some meta-learning methods can initiate model adjustments even before the full effects of CD are evident, by anticipating recurring patterns based on historical context and learned transitions. For example, in an electricity demand forecasting scenario, the meta-learner might detect that summer is approaching—based on contextual variables such as date or temperature and activate a model previously trained under similar seasonal conditions.

The third category includes **Unsupervised Learning and Clustering Approaches**. These methods do not rely on labeled data and instead model the structure of the data stream itself. Concepts are represented as clusters or distributions, and similarity metrics (e.g., Euclidean distance, KL-divergence) are used to detect when a current data segment resembles a previously seen concept. This enables reuse of historical models based on distributional similarity rather than classification performance.

In our previous work (Eliades and Papadopoulos, 2022, 2023, 2024), a detected CD led to immediate model retraining. In contrast, our proposed method Pool ICM implements *model reuse* for RCD. It maintains a pool of past classifiers and selects the most suitable one when drift is detected, retraining only if necessary. Like most methods, we rely on supervised learning to train predictive models. In addition, we use labeled data to detect exchangeability violations through ICM, a non-parametric, distribution-free method used for CD detection. We first perform a similarity check by testing the EA to identify candidate models from the pool. Among the models that pass this test, we then apply a supervised criterion evaluating their F1 score on recent labeled data to select the most appropriate model for reuse.

Specifically, whenever ICM signal a change, we evaluate each stored model by testing the exchangeability of a recent window of incoming data against the window immediately following that model's training period. Among the models that do not reject the EA, we select the one whose F1 score on the new window does not fall below a small threshold

relative to its historical F1 performance. If no model satisfies both the exchangeability and F1 criteria, a new classifier is trained and added to the pool.

The remainder of this paper is organized as follows: Section 2 surveys related work. Section 3 outlines the ICM framework. Section 4 introduces our POOL ICM method. Section 5 presents the experimental setup and results. Section 6 concludes the paper and discusses future work.

2. Related Work

This section provides an overview of the literature on CD, with a particular focus on techniques for handling recurrent concept drift (RCD). Section 2.1 reviews key approaches, including supervised, meta-learning, and unsupervised strategies.

2.1. Concept Drift

In this subsection, we review the key contributions to CD research relevant to our work, beginning with two comprehensive surveys.

Lu et al. (2019) examines over 130 high-quality publications, showing several methods in the field of CD. Their survey also analyzes ten widely used synthetic benchmarks and fourteen publicly available real-world datasets that are essential for evaluating learning algorithms under nonstationary conditions.

Bagui and Jin (2020) presents an examination of both synthetic and real-world streams, outlining the CD types and surveying many techniques. In particular, they evaluate publicly accessible datasets for benchmarking and discuss many strategies for managing distributional changes in streaming data.

As previously mentioned, according to Suárez-Cetrulo et al. (2023), methods for addressing RCD can be categorized into three main groups: supervised learning approaches, meta-learning approaches, and unsupervised learning and clustering approaches. Supervised methods typically rely on labeled data and involve ensembles or pools of models that are evaluated and reused based on their predictive performance. Meta-learning approaches operate at a higher level, learning to manage or select models based on concept transition patterns. Unsupervised and clustering based methods, on the other hand, use distributional similarity or clustering techniques to detect the recurrence of concepts without relying on labeled data. Each of these categories offers distinct mechanisms for identifying and adapting to recurring patterns in data streams.

2.1.1. Supervised learning approaches

A significant contribution in this area is the Accuracy Weighted Ensemble (AWE) algorithm, introduced in (Wang et al., 2003), which is a supervised ensemble method designed to handle CD in data streams. It trains a new classifier on each incoming data chunk and assigns weights to all existing models based on their classification accuracy on the most recent data. Only the top performing models are retained, while those with error rates exceeding a random baseline are discarded. AWE passively adapts to changes in the data distribution through performance based model selection. Although it does not explicitly model concept

recurrence, it may handle recurrent drift implicitly, as previously trained models can regain higher weights if they perform well on new data.

Another innovative approach is the Dynamic Weighted Majority (DWM) introduced by Kolter and Maloof (2007). DWM is an ensemble method designed to adapt to CD through four core principles: it incrementally trains online learners within the ensemble, assigns weights to them based on recent predictive accuracy, removes those that consistently underperform, and introduces new experts when the overall performance of the ensemble degrades. Although DWM does not explicitly detect changes or model recurrence, it may handle recurrent concepts implicitly if previously trained models are retained in the ensemble and regain influence as their performance improves.

The Learn++.NSE algorithm, an ensemble based approach presented in (Elwell and Polikar, 2011), is designed for incremental learning in nonstationary environments. It generates a new base classifier for each incoming data batch and combines all existing classifiers using a weighted voting scheme, where the weights are dynamically adjusted based on each model's recent classification accuracy. Learn++.NSE is capable of handling various types of CD, including gradual, abrupt, and cyclic drift, due to its strategy of retaining all past classifiers. Since base learners are not updated once added to the ensemble, previously learned concepts are preserved, allowing the algorithm to implicitly handle recurring drifts.

These algorithms offer practical solutions to the challenges posed by CD and even RCD, demonstrating strong performance on both synthetic and real world datasets. They adapt to changes in data by adjusting model weights or replacing underperforming classifiers based on predictive accuracy. However, these adaptations are typically reactive and heuristic in nature; there is no guarantee that the model adjustment corresponds to an actual distributional shift. Moreover, such methods do not explicitly test the EA. In contrast, our approach uses a statistical test with provable guarantees on the false detection rate, ensuring that change detections are supported by evidence of EA violation.

In our study (Eliades and Papadopoulos, 2021), we explored the integration of ICM with a histogram betting function. This novel combination is specifically designed to detect violations of the EA and, as a result, identify CD in data streams. A key advantage of our approach is that it is distribution free, in contrast to other methods that often presuppose a specific distribution in their drift detection metrics. This aspect of our methodology directly addresses the open question raised in Lu et al. (2019) regarding reliance on assumed distributions.

In Eliades and Papadopoulos (2022), we introduced a novel betting function for use within the ICM framework. It addresses a key limitation of Conformal Martingales: when the data distribution remains stable for an extended period, the martingale value can shrink close to zero, leading to delayed or even missed detections. To mitigate this, we proposed the Cautious betting function, which suppresses bets when there is no indication of change, thereby preventing unnecessary decreases in the martingale value. This function can be layered on top of existing betting functions to enhance their robustness.

Refinements to the above approach were proposed in (Eliades and Papadopoulos, 2023, 2024), where we proposed an ICM ensemble approach to tackle CD in data-stream classification. This system comprises 10 classifiers, each trained on distinct data sizes and operating within a majority voting framework for making predictions. By analyzing unique p-value sequences generated by each classifier through ICM, our distribution free method

efficiently detects change points, triggering retraining of the affected classifier. Additionally, in the second study (Eliades and Papadopoulos, 2024), we further enhanced this framework by combining Cautious betting function with multiple density estimators. Tested on four benchmark datasets it demonstrates accuracy that matches or surpasses that of three state of the art algorithms.

2.1.2. Meta-learning approaches

The GraphPool framework, proposed by Ahmadi and Kramer (2018), addresses RCD by maintaining and refining a pool of historical concepts. It enables model reuse by selecting previously learned concepts. It introduces a merging mechanism to control pool growth when similar concepts are detected. After each batch, it extracts a concept representation based on feature correlations and compares it to stored concepts using a multivariate likelihood test. Similar concepts are merged to reduce redundancy. GraphPool also tracks concept transitions via a first order Markov chain, allowing quick adaptation to periodic drifts. Experiments on synthetic and real world datasets confirm its effectiveness in both accuracy and pool management.

The research presented in (Wu et al., 2022) introduced PEARL (Probabilistic Exact Adaptive Random Forest) a framework designed to enhance the adaptability of random forests in data streams with RCD. Instead of discarding drifted trees, PEARL reuses relevant trees built in the past through two mechanisms: an exact pattern matching technique and a probabilistic graphical model. The exact method identifies sets of trees that previously co-occurred in predictions, while the graphical model learns tree transition patterns over time. Once stabilized, the probabilistic model replaces the exact strategy to identify suitable trees more efficiently. Additionally, Lossy Counting is applied to the graphical structure, providing guarantees controlling computation and memory usage.

2.1.3. unsupervised learning and clustering approaches

The authors of (Chiu and Minku, 2022) proposed the CDCMS framework, which is an ensemble based framework that uses clustering over model behaviors to manage multiple types of CD, including recurrent drift. It stores previously trained models in memory and applies clustering in the model space, where models are grouped based on their prediction behavior on a sliding window. When a drift is detected, the current model is compared with clusters of past models. If it belongs to a cluster, the drift is considered recurrent, and the relevant past models are reused. This technique allows unsupervised recognition of recurring concepts and enables memory efficient reuse of past models without assuming a fixed number of classes or explicitly tracking concept identities.

Din and Shao (2020) proposed the EMC (Evolving Micro-Clusters) framework, which addresses data stream classification by simultaneously managing CD, concept evolution (e.g., novel classes), and outliers. It dynamically maintains a set of evolving micro-clusters, which are updated online using error based representative learning. Rather than treating CD and evolution as separate problems, EMC handles both through a unified clustering based approach. A local density based detector is employed to identify novel class instances, distinguishing them from noise or drifted data. This enables EMC to adapt to non stationary data while maintaining strong classification performance and effective novel class detection.

3. Inductive Conformal Martingales

This section presents the basic concepts of ICM and explains how nonconformity scores and p-values are computed.

3.1. Data Exchangeability

Let $(Z_1, Z_2, \dots)$ be an infinite sequence of random variables. The joint probability distribution $\mathbb{P}(Z_1, Z_2, \dots, Z_N)$ is said to be exchangeable if it remains invariant under any permutation of its variables. An infinite sequence $(Z_1, Z_2, \dots)$ is exchangeable if, for every $N \in \mathbb{N}$, its marginal distribution $(Z_1, Z_2, \dots, Z_N)$ is exchangeable. Testing for exchangeability is equivalent to testing whether the data is independent and identically distributed (i.i.d.), as stated by de Finetti's theorem (Schervish, 1995), which states that any exchangeable distribution can be represented as a mixture of i.i.d. distributions.

3.2. Exchangeability Martingale

A test exchangeability martingale is a sequence of non negative random variables $(S_1, S_2, S_3, \dots)$ that satisfies the conditional expectation property $\mathbb{E}(S_{n+1} \mid S_1, \dots, S_n) = S_n$ under the null hypothesis of exchangeability.

Ville's inequality (Ville, 1939) states that the probability of a martingale exceeding a fixed threshold C is bounded: $\mathbb{P}(\exists n : S_n \geq C) \leq 1/C$. This implies that a large martingale value provides strong evidence against exchangeability. For instance, $S_n \geq 100$ corresponds to rejecting exchangeability at the 1% significance level.

3.3. Calculating Non-conformity Scores and P-values

Let $\{z_1, z_2, \dots\}$ be a sequence of examples, where $z_i = (x_i, y_i)$ with x_i an object given in the form of an input vector, and y_i the corresponding label. The Conformal Martingales(CM) approach generates a sequence of p-values for these examples and computes the martingale as a function of these p-values. As mentioned in Section 1, this work employs CM's computationally efficient inductive version. ICM uses the first k examples $\{z_1, z_2, \dots, z_k\}$ in the sequence to train a classification algorithm, which it then used to generate the p-values for the next examples. Consequently, it starts checking for violations of the EA from example z_{k+1}, focusing on the sequence $\{z_{k+1}, z_{k+2}, \dots\}$.

Our goal is to examine how strange or unusual a new example $z_j \in \{z_{k+1}, z_{k+2}, \dots\}$ is compared to the training examples. To make this possible, we define the function $A(z_i, \{z_1, \dots, z_k\})$, where $i \in \{k+1, \dots\}$, called a nonconformity measure (NCM) that assigns a numerical value α_i to each example z_i, called nonconformity score (NCS). The NCM is based on the trained underlying classification algorithm. The bigger the NCS value of an example, the less it conforms to $\{z_1, \dots, z_k\}$ according to the underlying algorithm.

For every new example z_j we generate the sequence $H_j = \{\alpha_{k+1}, \alpha_{k+2}, \dots, \alpha_{j-1}, \alpha_j\}$ to calculate its p-value. Note that the NCSs in H_j are calculated with the underlying algorithm trained on $\{z_1, z_2, \dots, z_k\}$. Given the sequence H_j we can calculate the corresponding p-value (p_j) of the new example z_j with the function:

$$p_j = \frac{|\{\alpha_i \in H_j | \alpha_i > \alpha_j\}| + U_j \cdot |\{\alpha_i \in H_j | \alpha_i = \alpha_j\}|}{j - k}, \tag{1}$$

where α_j is the NCS of the new example and α_i is the NCS of the i^{th} element in the example sequence set and U_j is a random number from the uniform distribution $(0, 1)$ to ensure valid p-values even with ties. For more information, refer to Vovk et al. (2003).

3.4. Constructing Exchangeability Martingales

An ICM is an exchangeability test martingale (see Subsection 3.2), which is calculated as a function of p-values such as the ones described in Subsection 3.3.

Given a sequence of p-values $(p_1, p_2, \dots)$ the martingale value S_n is calculated as:

$$S_n = \prod_{i=1}^{n} f_i(p_i) \tag{2}$$

where $f_i(p_i) = f_i(p_i|p_1, p_2, \dots, p_{i-1})$ is the betting function (Vovk et al., 2003).

The betting function should satisfy the constraint: $\int_0^1 f_i(p)dp = 1$, $f_i(p) \geq 0$ and also the S_n must keep the conditional expectation: $\mathbb{E}(S_{n+1}|S_0, S_1, \dots, S_n) = S_n$.

The condition $\int_0^1 f_i(p)\, dp = 1$ holds because $f_i(p)$ is a density estimator of the empirical p-value distribution $(p_1, p_2, \dots, p_{i-1})$. Any valid density over $[0, 1]$ can serve as a betting function, but using a density estimator allows to approximate the empirical p-value distribution. As shown by Fedorova et al. (2012), if the p-value sequence is stable and the estimator consistent, the plug-in martingale asymptotically achieves at least the same growth rate as any fixed-function martingale.

We also need to prove that $\mathbb{E}(S_{n+1}|S_0, S_1, \dots, S_n) = S_n$ under any exchangeable distribution.

Proposition 1 *If $\int_0^1 f_i(p)dp = 1$ then under any exchangeable distribution it holds:*

$$\mathbb{E}(S_{n+1} \mid S_0, S_1, \dots, S_n) = S_n$$

Proof [Proof of Proposition 1] The integral $\int_0^1 f_i(p)\, dp$ equals to 1

We will now prove that the conditional expectation is preserved under any exchangeable distribution:

$$\begin{aligned} \mathbb{E}(S_{n+1} \mid S_0, S_1, \dots, S_n) &= \int_0^1 \prod_{i=1}^{n} f_i(p_i) \cdot f_{n+1}(p)\, dp \\ &= \prod_{i=1}^{n} f_i(p_i) \cdot \int_0^1 f_{n+1}(p)\, dp \\ &= \prod_{i=1}^{n} f_i(p_i) = S_n \end{aligned} \tag{3}$$

■

Using equation (2), we observe that the martingale can be updated online via the recursive formula $S_n = S_{n-1} \cdot f_n(p_n)$

If the martingale reaches a value $S_n = M > 1$, then Ville's inequality (Ville, 1939) suggests that we can reject the EA with a significance level equal to $1/M$.

To deal numerical precision issues we compute (2) in the logarithmic scale.

4. Proposed Approach

This section outlines the proposed methodology. We begin by describing how CD is detected using ICM. We then explain how ICM, combined with the F1 score, is used to evaluate whether a model from the pool is suitable for reuse or if training a new classifier is necessary.

4.1. Detecting CD using ICM

In order to detect a CD at a pre specified significance level δ, the martingale value must exceed $1/\delta$, which leads to the rejection of the EA. This process is summarized in Algorithm 1. Specifically, if the martingale value S_k at a given point k exceeds 100, a CD is detected with a significance level of 1%, where L in the algorithm denotes the number of p-values that our estimator uses.

Algorithm 1: Detect CD using ICM
Input: Training set $\{z_1, \ldots, z_k\}$, Test set $\{z_{k+1}, \ldots, z_n\}$, significance level δ
Output: Drift alarms if EA is violated
Initialize $S_1 \leftarrow 1$ **for** $i = 1$ **to** $n - k$ **do**
  $\alpha_i \leftarrow A(z_{k+i}, \{z_1, \ldots, z_k\})$;
  $p_i \leftarrow \frac{|\{j:\alpha_j > \alpha_i\}| + U_i \cdot |\{j:\alpha_j = \alpha_i\}|}{i}$
  Calculate betting function $h_i = h(p_{i-L}, \ldots, p_{i-1})$;
  $S_i \leftarrow S_{i-1} \cdot h_i(p_i)$;
  if $S_i > \frac{1}{\delta}$ **then**
    Raise an alarm
  end
end

4.2. Pool ICM

In our earlier studies (Eliades and Papadopoulos, 2022, 2023, 2024), the detection of CD led to discarding the current model and retraining it from scratch. In contrast, the approach proposed in this work maintains a pool of previously trained models. Upon detecting a CD, we re-evaluate the suitability of these models on recent data. A new model is trained only if none of the existing models is deemed suitable. Thus, (i) retraining is no longer required after every CD detection, and (ii) the total number of training instances used is significantly reduced.

The Pool ICM method operates as follows. Initially, we wait for a prespecified number of observations K to arrive and train our first classifier, which is then added to the pool. This classifier begins making predictions from timestamp $K + 1$.

As new observations arrive, we monitor for CD using ICM. During this monitoring process, we also store a prespecified number M of nonconformity scores NCMs from the current classifier, collected from timestamps $K + 1$ to $K + M$. We denote this window of nonconformity scores as set A_i. This window captures the nonconformity scores under stable conditions before any drift is suspected.

When a CD is detected by ICM at timestamp $t + K$, we evaluate each classifier C_i in the pool by forcing them to make a prespecified number L of predictions. We compute

their nonconformity scores starting from timestamp $t + K - d$ to $t + K - d + L$, where d accounts for the delay between the actual change and its detection (since CD typically begins before it is detected). This evaluation window is denoted as set B_i. This captures the nonconformity scores of each candidate under the new concept.

We then merge the two sets, A_i and B_i, while preserving the chronological order of the NCMs, and apply an exchangeability test to the combined sequence $[A_i\, B_i]$. If the EA is not rejected for a candidate classifier, we compute its F1 score. Among the classifiers for which the EA is not rejected, we select the one with the highest F1 score provided that this score remains within a small margin of its historical F1 performance. The selected classifier begins making predictions at timestamp $t + K - d + L + 1$.

If no classifier passes both the EA test and the F1 score criterion, we wait for an additional K instances and train a new classifier, which is then added to the pool. This newly trained classifier will start making predictions at timestamp $t + K + K + 1 - d$.

The gain in the number of prediction instances, compared to training a new classifier, is given by:

$$(t + K + K + 1 - d) - (t + K - d + L + 1) = K - L \tag{4}$$

To express this gain as a percentage of the total number of prediction instances if a new classifier had been trained, we compute:

$$\text{Relative Gain } (\%) = \frac{(K - L)(\text{NoDr} - \text{NoM})}{K \cdot \text{NoDr}} \times 100 \tag{5}$$

Here, d is the number of instances from the point when the martingale value of the classifier first exceeds a lower threshold r (used to anticipate drift) until the drift is officially detected. Formally,

$$d = \max\{j : S_j^{\text{classifier}} < r\} + 1,$$

where $S_j^{\text{classifier}}$ denotes the martingale value at timestamp j for the candidate classifier.

The whole process is summarized at algorithm 2

5. Experiments and Results

This section experimentally evaluates the performance of the proposed **Pool ICM** approach, which aims to reduce both the number of retrained models and the total number of training instances while maintaining competitive accuracy.

We compare the performance of Pool ICM using the **Cautious betting function**, which incorporates multiple density estimators with that of other approaches on one synthetic dataset (STAGGER) and two publicly available real world datasets (ELEC and AIRLINES).

For all three datasets, we benchmark our method against two state of the art approaches from the literature: the DWM method (Kolter and Maloof, 2007) and the AWE method (Wang et al., 2003).

Algorithm 2: Pool ICM: Concept Drift Detection and Model Selection

Input: Data stream $\{z_1, z_2, \dots\}$, thresholds K, M, L, significance level δ, early warning threshold r

Output: Updated classifier pool, prediction results

Initialize pool $\mathcal{P} \leftarrow \emptyset$ Wait until K instances arrive Train initial classifier C_1 on $\{z_1, \dots, z_K\}$ and add to pool $\mathcal{P}$ Set $t \leftarrow K + 1$; `// Start predictions from time` $K + 1$

Initialize ICM for current classifier C_{curr} Initialize Set A: collect M nonconformity scores $\alpha_{K+1}, \dots, \alpha_{K+M}$ from C_{curr}

while *stream continues* **do**

  Monitor CD using ICM with incoming z_t and update S_t

  if $S_t > 1/\delta$ **then**

    Estimate $d = \max\{j : S_j < r\} + 1$ Define Set B_i: For each classifier C_i in pool $\mathcal{P}$, predict L labels from z_{t+K-d} to $z_{t+K-d+L}$ and compute corresponding NCMs

    foreach $C_i \in \mathcal{P}$ **do**

      Merge Set A_i and Set B_i of C_i in time order $\rightarrow [A_i\, B_i]$

      if *EA is not rejected on* $[A_i\, B_i]$ **then**

        Compute $F1_i^{past}$, $F1_i$ score on window A_i and B_i respectively

      end

      else

        Mark C_i as unsuitable

      end

    end

    if *any suitable classifier exists* **then**

      Among classifiers for which the exchangeability assumption is not rejected,
      select classifier C^* with the highest F1 score,
      provided it satisfies: $\text{F1}_i > \max(\text{F1}_i^{\text{past}} - 0.05,\ 0)$;
      Set $C_{\text{curr}} \leftarrow C^*$;
      Start predicting from $t + K - d + L + 1$

    end

    else

      Wait for K new instances $\{z_{t+1}, \dots, z_{t+K}\}$
      Train new classifier C_{new} on this data Add C_{new} to pool $\mathcal{P}$ Set $C_{\text{curr}} \leftarrow C_{\text{new}}$
      Start predicting from $t + K + K + 1 - d$

    end

  end

  $t \leftarrow t + 1$; `// Move to next instance`

end

5.1. Datasets

5.1.1. Synthetic Benchmark Dataset

The STAGGER dataset (Schlimmer and Granger, 1986) is a widely used benchmark for evaluating CD detection methods. For our simulations, we generated 1,000,000 instances. Each instance is described by three categorical attributes and has a binary class label. The drift type is recurrent with sudden changes, involving three distinct concepts. A CD occurs every 10,000 instances, meaning that each concept chunk consists of 10,000 examples.

In our experiments on this dataset, we set the training set size to 200 instances. This corresponds to just 2% of the data in each chunk, making the training size relatively small compared to the length of the concept period.

We evaluate transitions through concepts in the sequence $a \rightarrow b \rightarrow c \rightarrow a$. To further assess the robustness of our approach, we also inject 10% label noise by randomly altering class labels, and we report performance both with and without noise.

5.1.2. Real World Benchmark Datasets

The Airlines dataset (Ikonomovska E, 2010) originates from the Data Expo Competition 2009 and contains flight arrival and departure records for commercial flights within the USA from October 1987 to April 2008. It consists of 539,383 instances, each described by seven features. The task is to classify whether a flight was delayed or not. In our experiments, we used a training set size of 200 instances. Note the training set size is denoted by K in Algorithm 2.

The ELEC dataset (Harries et al., 1999) is a time series consisting of 45,312 instances recorded at 30 minute intervals, with a binary class label indicating whether the electricity price rose or fell compared to the moving average of the previous 24 hours. Each instance is described by eight variables, and the data was collected from the Australian New South Wales Electricity Market. In our experiments, we excluded the variables `time`, `date`, `day`, and `period`, using only `nswprice`, `nswdemand`, `transfer`, `vicprice`, and `vicdemand`. The training set size was set to 300 instances, which corresponds to using less than one week of observations for training.

5.2. Performance Measures

To evaluate the performance of the proposed approach, we consider five evaluation measures. Some metrics cannot be reported on real-world datasets due to the lack of ground truth for change points. Here, the term "average" refers to the mean over multiple experimental runs, as explained in Section 5.3.

(a) **Accuracy(Acc):** The average classification accuracy, computed over all predictions excluding those in the training set or the ones used to check which model is more suitable.

(b) **Mean Delay(MD):** The average number of observations between the actual occurrence of a CD and its detection.

(c) **True Alarm Rate (TAR):** The average number of correctly detected CDs per chunk.

(d) **False Alarm Rate (FAR):** The average number of incorrect CD detections per chunk.

(f) **Number of CDs detected (NoDr):** The average number of CDs detected.

(g) **Number of Models (NoM):** The average number of models trained.

5.3. Experimental Setting

In this section, we present the experimental results from three sets of experiments. For the benchmark datasets, we employed a TreeBagger classifier with 40 trees. For each example, the TreeBagger (or tree based) classifier outputs a posterior probability $\tilde{p}_j$ for the true label y_j, and the corresponding nonconformity measure (NCM) is defined as $\alpha_j = -\tilde{p}_j$.

In all experiments, we use the Cautious-Multi(InterHist-NN) betting function, as proposed in Eliades and Papadopoulos (2024). This configuration combines the Cautious Betting Function with three interpolated histogram and three nearest neighbor density estimators. The parameters of the Cautious Betting Function are set to $\epsilon = 100$ and $W = 5000$. The parameters of Algorithm 2 are set to $M = 150$, $L = 150$, $\delta = 100$ and $r = 10$.

The remaining two subsections present simulations on two synthetic and two real world datasets. We focus on CD, model selection from the pool, and model retraining to recover accuracy. All reported results are averaged over five simulation runs per dataset.

We empirically demonstrate that Pool ICM reduces both the number of model retraining events and the number of training instances used, while maintaining high accuracy. Additionally, we compare the accuracy of our approach against two state of the art algorithms: AWE and DWM-NB, as described in Section 2. The reported accuracies of these baseline methods were obtained from Sarnovsky and Kolarik (2021b).

5.4. Evaluation on Synthetic and Real Datasets

Table 1 summarizes the experimental results of the proposed Pool ICM approach on both synthetic and real world datasets. The table presents performance under different settings of the cautious threshold parameter $\epsilon \in \{0, 100\}$ and varying levels of label noise.

For the synthetic STAGGER dataset, we examine the impact of noise $(0\%, 10\%)$ and the value of $\epsilon = \{0, 100\}$ on performance metrics. With $\epsilon = 100$, we observe a significant reduction in mean delay. This improvement is due to the Cautious Betting Function avoiding unnecessary bets, preventing the martingale from dropping to values close to 0 and thus allowing faster recovery while maintaining perfect TAR of 1. Even under 10% label noise, accuracy and detection performance remain robust. The increase in accuracy with $\epsilon = 100$ is attributed to earlier CD detection, which allows Pool ICM to switch to a suitable model more promptly. Additionally, the false alarm rate is reduced under $\epsilon = 100$, again due to the strategic avoidance of unnecessary betting. As mentioned in Section 5.1.1, the STAGGER dataset consists of three concepts. Theoretically, three models, each trained on a single concept, should be sufficient to maintain high accuracy. In all scenarios, the number of detected changes was slightly above 99, with the ideal being exactly 99. Nevertheless, the number of models trained remained significantly lower. On average, across all four scenarios, only 3 to 6 models were added to the pool. This indicates a significant reduction in both the

number of model retrainings and the total number of training instances required, allowing more instances to be used for prediction.

Using Equation (5), which estimates the percentage of relative gain in prediction instances achieved by reusing models instead of retraining after every drift (with $K = 200$ and $L = 150$), we compute a relative gain of approximately 24% across all scenarios. Additionally, this translates to about 5,000 extra prediction instances, or roughly 0.5% of the full dataset.

For the real world datasets (AIRLINES and ELEC), we evaluate the effect of Pool ICM with and without cautious betting ($\epsilon = 100$ and $\epsilon = 0$ respectively). Although no ground truth drift points are known for these datasets (hence MD, TAR and FAR are omitted), the Acc, NoDR and NoM provide a measure of the proposed method's performance.

While examining the AIRLINES dataset, we observe that accuracy improves when $\epsilon = 100$. This is likely due to the dataset containing long duration concepts, where the strategy of avoiding unnecessary betting allows the martingale to recover more quickly without missing a change. It also helps prevent decay, i.e., the phenomenon where the new distribution becomes the new "normal" and the martingale stops growing. For more details on decay, see Vovk et al. (2024). When we use $\epsilon = 0$, on average five CDs were detected and only three model retrainings were required. In contrast, with $\epsilon = 100$, 89 CDs were detected on average, yet only ten retrainings were needed.

Applying Equation (5), with $K = 200$ and $L = 150$ we find a relative gain of approximately $13\%, 22\%$ for $\epsilon = 0, 100$ respectively. This corresponds to about 130 and $3,950$ extra prediction instances respectively, or roughly 0.02% and 0.73% respectively of the whole dataset. The very small gain observed with $\epsilon = 0$ is due to the long duration concepts present in the dataset, which result in fewer detected drifts.

For the ELEC dataset, setting $\epsilon = 100$ results in a slight decrease in accuracy and a marginally lower number of detected CDs compared to $\epsilon = 0$. One possible explanation is that the concepts in this dataset have short durations, allowing the martingale to recover quickly even without the Cautious Betting Function. Nevertheless, as in previous cases, the number of model retrainings is significantly reduced, to approximately one third, of the total number of detected changes.

Applying Equation (5) with $K = 300$ and $L = 150$, we compute a relative gain of approximately 32% and 33% for $\epsilon = 0$ and $\epsilon = 100$, respectively. This translates to roughly $11,580$ and $11,685$ additional prediction instances, corresponding to about 26% of the entire dataset in both cases. The percentage gain over the entire dataset is higher than in the other two datasets, primarily because this dataset exhibits shorter concept durations.

Overall, the results confirm that Pool ICM effectively reduces both the retraining frequency and the number of training instances required, particularly in scenarios with frequent RCD. Setting $\epsilon = 100$ appears to be a safe and effective choice: it yields higher accuracy on the STAGGER and AIRLINES datasets, and achieves comparable performance to $\epsilon = 0$ on the ELEC dataset.

5.5. Comparison with Existing Approaches

As shown in the previous sections, the proposed Pool ICM method consistently reduces the number of model retrainings required across all datasets. This reduction, however, comes

Table 1: Performance of Pool ICM

Dataset	ϵ	Noise (%)	Acc	MD	TAR	FAR	NoDR	NoM
STAGGER	0	0	0.98742	215.09	1	0.03	103	4
STAGGER	100	0	0.99905	18.39	1	0.02	102	3
STAGGER	0	10	0.93101	359.42	1	0.06	106	6
STAGGER	100	10	0.93290	37.96	1	0.02	102	6
AIRLINES	0	unknown	0.55057	-	-	-	5.2	2.6
AIRLINES	100	unknown	0.56764	-	-	-	89	10
ELEC	0	unknown	0.74249	-	-	-	118.8	41.6
ELEC	100	unknown	0.74188	-	-	-	117.8	39.9

with a modest trade off in accuracy due to the reuse of existing models instead of training a new one after every detected change.

Table 2 presents a comparative analysis of the Pool ICM (using $\epsilon = 100$) with:

- the best performing configuration of the Cautious Betting Function from our previous work (Eliades and Papadopoulos, 2022) (corresponding to the Standard ICM column).
- two state of the art methods from the literature AWE and DWM-NB with results taken from (Sarnovsky and Kolarik, 2021a).

Note that the results for the STAGGER dataset correspond to the variant with 10% label noise.

Table 2: Comparison of Pool ICM with previous and state of the art approaches (accuracy)

Dataset	Pool ICM	Standard ICM	AWE	DWM-NB
STAGGER	0.933	0.946	0.948	0.901
ELEC	0.742	0.759	0.756	0.800
AIRLINES	0.568	0.602	0.618	0.640

While Pool ICM does not outperform the top methods in terms of raw accuracy, its strength lies in efficiency. By reusing models whose produced p-values do not provide sufficient evidence to reject the EA combined with evaluation using the F1 score it significantly reduces the number of data instances required for retraining and lowers the computational cost. We report accuracy in Table 2 to allow direct comparison with prior work, where accuracy is the standard evaluation metric used for AWE and DWM-NB. While Pool ICM selects models based on F1 score to better handle class imbalance particularly in datasets such as AIRLINES and ELEC many previous methods do not publish F1 scores, making a fair F1-based comparison infeasible. Nevertheless, Pool ICM can be easily adapted to use other selection metrics, including accuracy.

Moreover, Pool ICM offers a practical balance between accuracy and computational cost. The slightly lower accuracy particularly in more complex datasets such as AIRLINES may be an acceptable trade off in many applications, given the gains in resource conservation.

6. Conclusions

This study proposes the **Pool ICM** framework, a novel approach for handling RCD in streaming data. By leveraging ICM for drift detection and combining exchangeability testing (of existing models) with F1 score–based evaluation of model performance, Pool ICM effectively reuses previously trained models instead of retraining after every detected change.

Experimental results on both synthetic and real world datasets demonstrate that Pool ICM significantly reduces the number of model retrainings and the volume of training data required, while maintaining competitive accuracy. This makes it a promising method for resource constrained environments or applications requiring real time processing.

While the accuracy achieved is slightly below that of our previous approaches, the computational gains and efficiency highlight Pool ICM as a practical solution for real world deployment.

Future work will explore improved model selection strategies, potentially incorporating context aware features or recurrent drift modeling. We also plan to extend Pool ICM into an ensemble framework to further enhance robustness and performance across diverse drift scenarios and to examine whether the number of instances without prediction is reduced.

References

Zahra Ahmadi and Stefan Kramer. Modeling recurring concepts in data streams: a graph-based framework. *Knowledge and Information Systems*, 55(1):15–44, 2018. doi: 10.1007/s10115-017-1070-0.

Sikha Bagui and Katie Jin. A survey of challenges facing streaming data. *Transactions on Machine Learning and Artificial Intelligence*, 8(4):63–73, Aug. 2020. doi: 10.14738/tmlai.84.8579. URL https://journals.scholarpublishing.org/index.php/TMLAI/article/view/8579.

Chun Wai Chiu and Leandro L. Minku. A diversity framework for dealing with multiple types of concept drift based on clustering in the model space. *IEEE Transactions on Neural Networks and Learning Systems*, 33(3):1299–1309, 2022. doi: 10.1109/TNNLS.2020.3041684.

Salah Ud Din and Junming Shao. Exploiting evolving micro-clusters for data stream classification with emerging class detection. *Information Sciences*, 507:404–420, 2020. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2019.08.050. URL https://www.sciencedirect.com/science/article/pii/S0020025519307960.

Charalambos Eliades and Harris Papadopoulos. Using inductive conformal martingales for addressing concept drift in data stream classification. In Lars Carlsson, Zhiyuan Luo, Giovanni Cherubin, and Khuong An Nguyen, editors, *Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 152 of *Proceedings of Machine Learning Research*, pages 171–190, 08–10 Sep 2021. URL https://proceedings.mlr.press/v152/eliades21a.html.

Charalambos Eliades and Harris Papadopoulos. A betting function for addressing concept drift with conformal martingales. In Ulf Johansson, Henrik Boström, Khuong

An Nguyen, Zhiyuan Luo, and Lars Carlsson, editors, *Proceedings of the Eleventh Symposium on Conformal and Probabilistic Prediction with Applications*, volume 179 of *Proceedings of Machine Learning Research*, pages 219–238, 24–26 Aug 2022. URL https://proceedings.mlr.press/v179/eliades22a.html.

Charalambos Eliades and Harris Papadopoulos. A conformal martingales ensemble approach for addressing concept drift. In Harris Papadopoulos, Khuong An Nguyen, Henrik Boström, and Lars Carlsson, editors, *Proceedings of the Twelfth Symposium on Conformal and Probabilistic Prediction with Applications*, volume 204 of *Proceedings of Machine Learning Research*, pages 328–346, 13–15 Sep 2023. URL https://proceedings.mlr.press/v204/eliades23a.html.

Charalambos Eliades and Harris Papadopoulos. ICM ensemble with novel betting functions for concept drift. *Machine Learning*, 113(9):6911–6944, 2024. doi: 10.1007/s10994-024-06593-0. URL https://doi.org/10.1007/s10994-024-06593-0.

Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011. doi: 10.1109/TNN.2011.2160459.

Valentina Fedorova, Alex Gammerman, Ilia Nouretdinov, and Vladimir Vovk. Plug-in martingales for testing exchangeability on-line. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 923–930, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Michael Harries, U Nsw cse tr, and New South Wales. Splice-2 comparative evaluation: Electricity pricing. Technical report, 1999.

Shen-Shyang Ho. A martingale framework for concept change detection in time-varying data streams. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML 05, page 321–327, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595931805. URL https://doi.org/10.1145/1102351.1102392.

Shen-Shyang Ho and Harry Wechsler. On the detection of concept changes in time-varying data stream by testing exchangeability. *CoRR*, abs/1207.1379, 2012. URL http://arxiv.org/abs/1207.1379.

Shen-Shyang Ho, Matthew Schofield, Bo Sun, Jason Snouffer, and Jean Kirschner. A martingale-based approach for flight behavior anomaly detection. In *2019 20th IEEE International Conference on Mobile Data Management (MDM)*, pages 43–52, 2019. doi: 10.1109/MDM.2019.00-75.

Dveroski S Ikonomovska E, Gama J. Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, 23:128–168, 2010.

J. Zico Kolter and Marcus A. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *J. Mach. Learn. Res.*, 8:2755–2790, December 2007. ISSN 1532-4435.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, João Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12): 2346–2363, 2019. doi: 10.1109/TKDE.2018.2876857.

Martin Sarnovsky and Michal Kolarik. Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensemble. *PeerJ Computer Science*, 7, 04 2021a. doi: 10.7717/peerj-cs.459.

Martin Sarnovsky and Michal Kolarik. Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensemble. *PeerJ Computer Science*, 7, 04 2021b. doi: 10.7717/peerj-cs.459.

Mark J. Schervish. *Theory of Statistics*. 1995.

Jeffrey C. Schlimmer and Richard H. Granger. Incremental learning from noisy data. *Mach. Learn.*, 1(3):317–354, March 1986. ISSN 0885-6125. doi: 10.1023/A:1022810614389. URL https://doi.org/10.1023/A:1022810614389.

Andrés L. Suárez-Cetrulo, David Quintana, and Alejandro Cervantes. A survey on machine learning for recurring concept drifting data streams. *Expert Systems with Applications*, 213:118934, 2023. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2022.118934. URL https://www.sciencedirect.com/science/article/pii/S0957417422019522.

J. Ville. Étude critique de la notion de collectif. by j. ville. pp. 144. 75 francs. 1939. monographies des probabilités, calcul des probabilités et ses applications, publiées sous la direction de m. Émile borel, fascicule iii. (gauthier-villars, paris). *The Mathematical Gazette*, 23(257):490–491, 1939. doi: 10.2307/3607027.

Denis Volkhonskiy, Evgeny Burnaev, Ilia Nouretdinov, Alexander Gammerman, and Vladimir Vovk. Inductive conformal martingales for change-point detection. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Harris Papadopoulos, editors, *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, volume 60 of *Proceedings of Machine Learning Research*, pages 132–153, Stockholm, Sweden, 13–16 Jun 2017. PMLR. URL http://proceedings.mlr.press/v60/volkhonskiy17a.html.

Vladimir Vovk. Testing for concept shift online, 2020.

Vladimir Vovk, Ilia Nouretdinov, and Alex Gammerman. Testing exchangeability on-line. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*, pages 768–775. AAAI Press, 2003.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. 01 2005. doi: 10.1007/b106715.

Vladimir Vovk, Ilia Nouretdinov, and Alex Gammerman. Validity and efficiency of the conformal cusum procedure, 2024. URL https://arxiv.org/abs/2412.03464.

Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 226–235, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137370. URL https://doi.org/10.1145/956750.956778.

Ocean Wu, Yun Sing Koh, Gillian Dobbie, and Thomas Lacombe. Probabilistic exact adaptive random forest for recurrent concepts in data streams. *International Journal of Data Science and Analytics*, 13(1):17–32, 2022. doi: 10.1007/s41060-021-00273-1.

Puning Zhao and Lifeng Lai. Analysis of knn density estimation. *IEEE Transactions on Information Theory*, 68(12):7971–7995, 2022. doi: 10.1109/TIT.2022.3195870.