Online Drift Detection with Maximum Concept Discrepancy

Ke Wan*
University of Illinois at Urbana-Champaign
Illinois, USA
kewan2@illinois.edu

Yi Liang*
Fudan University
Shanghai, China
yliang23@m.fudan.edu.cn

Susik Yoon§
Korea University
Seoul, Korea
susik@korea.ac.kr

ABSTRACT

Continuous learning from an immense volume of data streams becomes exceptionally critical in the internet era. However, data streams often do not conform to the same distribution over time, leading to a phenomenon called concept drift. Since a fixed static model is unreliable for inferring concept-drifted data streams, establishing an adaptive mechanism for detecting concept drift is crucial. Current methods for concept drift detection primarily assume that the labels or error rates of downstream models are given and/or underlying statistical properties exist in data streams. These approaches, however, struggle to address high-dimensional data streams with intricate irregular distribution shifts, which are more prevalent in real-world scenarios. In this paper, we propose MCD-DD, a novel concept drift detection method based on maximum concept discrepancy, inspired by the maximum mean discrepancy. Our method can adaptively identify varying forms of concept drift by contrastive learning of concept embeddings without relying on labels or statistical properties. With thorough experiments under synthetic and real-world scenarios, we demonstrate that the proposed method outperforms existing baselines in identifying concept drifts and enables qualitative analysis with high explainability.

CCS CONCEPTS

• **Information systems** → **Data stream mining**.

KEYWORDS

Concept Drift Detection; Maximum Concept Discrepancy

ACM Reference Format:
Ke Wan*, Yi Liang*, and Susik Yoon§. 2024. Online Drift Detection with Maximum Concept Discrepancy. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3672016

1 INTRODUCTION

1.1 Background and Motivation

Continuously learning from evolving data streams is crucial for numerous online services to derive real-time insights [5, 50, 51]. However, in many real-world scenarios, data streams from different times may exhibit distinct characteristics [25, 41, 49]. For instance, a previously stable weather pattern might incrementally change due to global warming with unprecedented high temperatures, leading to unpredictable fluctuations in temperature, wind speed, and humidity. This phenomenon is called *concept drift* in data streams [30], indicating that data at different times follows distinct probability distributions. Developing methods for continuously detecting whether a data stream has undergone concept drift is imperative, since it is impractical to employ consistent modeling to the concept-drifted data streams (e.g., a weather prediction model needs to be updated after unprecedented temperature changes are observed).

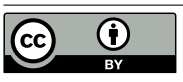
Current methods for detecting concept drift online fall into two categories: error rate-based or data distribution-based [2, 13, 39]. A common tactic involves constructing a hypothesis test statistic to determine whether error rates of downstream models or data samples from different periods adhere to the same probability distribution under a certain significance level. While this approach is favored for its interpretability and strong statistical foundation, distinguishing between natural fluctuations and actual drifts poses challenges, especially in the context of complex, evolving data streams. The sparsity, noise, and high dimensionality commonly observed in real-world data streams can make statistical approaches ineffective. Additionally, obtaining error rates of downstream models is not always feasible, as true labels may not be readily available.

Meanwhile, in machine learning, kernel methods are commonly used to map data into high-dimensional spaces [11], improving its representation for downstream tasks such as classification and clustering with more distinct separations in the projected space. Likewise, kernel methods can be used to transform a set of sampled data into a space where the existing concepts can be effectively represented, facilitating the detection of potential concept drifts. However, traditional kernels like the Gaussian kernel [20] are limited in their ability to detect concept drifts. They are designed with a deterministic mapping function, making them ill-suited for the ever-changing distributions of data streams. While deep kernels offer more flexibility [29, 47], adapting them to address concurrently evolving concepts, especially in unsupervised settings, remains a significant challenge. The computational costs associated with updating these kernels repeatedly can be prohibitive.

1.2 Main Idea and Challenges

Detecting concept drifts from data streams presents numerous challenges, mainly centered around the representation of ever-changing data distributions (i.e., concept representation) and the measurement of their differences (i.e., drift quantification). It also necessitates the continuous monitoring of the dynamic shifts in data distributions as they evolve, which is crucial for accurately identifying

*Equal contributions. § Corresponding author.



KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3672016

arbitrary drifts (i.e., online updates). Furthermore, in real-world scenarios, there is often a lack of ground truth labels for concept drifts as well as downstream tasks, making an unsupervised approach (i.e., data distribution-based) preferable to a supervised approach (i.e., error rate-based) in practice. To address these objectives, we propose a novel method for continuously identifying concept drifts in an *unsupervised* and *online* manner, that can effectively handle arbitrary data distributions with high interpretability.

The main idea of this work is to employ a new measure *Maximum Concept Discrepancy* for concept drift detection, inspired by the maximum mean discrepancy [38] with a kernel function. Through a deep neural network, we encode a set of sample data points in a short time period into a compact representation that captures the concept observed during the period. We leverage contrastive learning accompanied by time-aware sampling strategies to learn the embedding space of concepts. This entails the generation of positive sample pairs drawn from temporally proximate distributions and that of negative sample pairs from temporally distant distributions, while also introducing controlled perturbations. The embedding space is continuously updated to bring positive samples closer together and push negative samples further apart. Concept drifts are then identified by evaluating the discrepancy between the representations of concepts in consecutive time periods. In addition, the maximum concept discrepancy between two concepts can be bounded with a statistical significance. It can function as a theoretical threshold for detecting concept drifts, providing high interpretability and practicality for our method. In consequence, our method is capable of continuously identifying various types of concept drift from data streams without any supervision. It effectively addresses the aforementioned challenges for concept drift detection while keeping the advantages of both online statistical approaches and offline deep kernel-based approaches.

1.3 Summary

As a concrete implementation of our main idea, we propose an algorithm *MCD-DD (Maximum Concept Discrepancy-based Drift Detector)*, aiming at unsupervised online concept drift detection from data streams. The main contributions of this work can be summarized as follows:

- To the best of our knowledge, this is the first work to propose a dynamically updated measure, *maximum concept discrepancy*, for unsupervised online concept drift detection.
- We propose a novel method MCD-DD equipped with the sample set encoder and drift detector, optimized by contrastive objective with time-aware sampling strategies. For reproducibility, the source code of MCD-DD is publicly available[1].
- Theoretical analysis of learning the maximum concept discrepancy provides its statistical interpretation and complexity.
- Comprehensive experiments are conducted on 11 data sets with varying complexities of drifts. MCD-DD achieves state-of-the-art results in three performance metrics and demonstrates better interpretability in qualitative analysis, compared with baselines.

[1] https://github.com/LiangYiAnita/mcd-dd

2 RELATED WORK

2.1 Concept Drift Detection

Concept drift detection is essential in employing a model robustly in data streams [1, 2, 15, 30]. Error rate-based drift detection is the most commonly used supervised method for detecting concept drift [4, 12, 28, 36, 48]. This approach continuously monitors the performance of downstream models in data streams. It relies on a trained predictive model and assesses whether concept drift has occurred by examining the consistency of the model's predictive performance over different time intervals [2]. For an unsupervised approach [16], it is common to conduct statistical tests on the two samples from different periods to determine whether they originate from the same concept [6, 19, 24, 29, 34], called data distribution-based detection, which is the scope of this work. While some error rate-based detectors [4, 33] can be adopted for this setting, it is not straightforward to apply them to multivariate data streams. It is also worth noting that some recent works try variants for concept drift detection with a pre-trained model [7, 54], active learning [55], imbalanced [26] or resource-constrained [44] streaming settings.

2.2 Contrastive Learning in Data Streams

Contrastive learning, as an effective self-supervised learning paradigm [8], is widely applied in various detection tasks in data streams [44, 46, 53]. The nature of data streams with scarce or delayed labels and lack of external supervision leads to the adoption of continual learning with contrastive losses. The pseudo-labeling for preparing positive and negative samples is a critical design factor and its strategy ranges from model confidence-based [53], learnable focuses [46], to class prototype [44] tailored for downstream tasks. Despite the advancements in contrastive learning, current techniques have yet to be explored for learning separable embeddings of probability distributions representing varying concepts or for application in two-sample tests with statistical bounds, both of which are addressed in this study.

2.3 Maximun Mean Discrepancy

The utilization of Maximum Mean Discrepancy (MMD) has been widespread, primarily serving to map data into high-dimensional spaces and thereby enhancing separability for downstream tasks [23]. MMD has been also actively applied for designing generative models [10, 27] and detecting whether two samples originate from the same distribution [18] with the Gaussian kernel function [19] or the deep kernels [29] to achieve greater flexibility and expressiveness. The idea of Maximum Concept Discrepancy (MCD) in this study draws inspiration from MMD-based approaches but is specifically tailored for unsupervised online concept drift detection by integrating a deep encoder for sample sets to represent data distributions and continuous learning strategies to dynamically optimize the projected space encompassing varying concepts.

3 PRELIMINARIES

3.1 Concept Drift

Concept drift is a phenomenon referring to the arbitrary changes in the statistical properties of a target domain of data over time. Formally, concept drift at time t is defined as the change in the

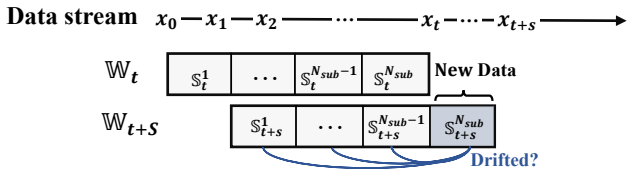


Figure 1: Unsupervised online concept drift detection over sliding window $\mathbb{W}$ with sub-windows $\mathbb{S}$.

joint probability of data points X and labels y at time t, denoted as $P_t(X, y) \neq P_{t+1}(X, y)$. Concept drift primarily originates from one of the following three sources [30]: (i) $P_t(Y|X) \neq P_{t+1}(Y|X)$, when the conditional distribution of the target variable Y given the covariate X undergoes drift; (ii) $P_t(X) \neq P_{t+1}(X)$, when the distribution of the covariate experiences drift; and (iii) combination of (i) and (ii). In addition to varying sources, concept drift can also be distinguished into four types based on the specific nature of the drift occurrence: sudden, reoccurring, gradual, and incremental. For additional references, we direct readers to recent surveys [1, 2, 30]. This work aims to develop an unsupervised method for detecting various types of drift caused by the source described in (ii).

3.2 Problem Setting

Given a continuously evolving data stream $\mathcal{X} = \{x_t\}_{t=0}^{\infty}$, we maintain the latest context of the data stream by employing a *sliding window* $\mathbb{W}_t$ of size W updated by a slide of size S (i.e., $\mathbb{W}_t = \{x_{t-i}\}_{i=0}^{W-1}$ and $\mathbb{S}_t = \{x_{t-i}\}_{i=0}^{S-1}$). The window and slide sizes can be defined either in terms of the number of data points or a time period.

Then, a window $\mathbb{W}_t$ consists of non-overlapping slides indexed by $j = 1, \ldots, N_{sub}$ where $N_{sub} = W/S$ is the number of slides in a window (i.e., $\mathbb{W}_t = \bigcup_{j=1}^{N_{sub}} \mathbb{S}_t^j$ and $\mathbb{S}_t^j = \{x_{t-(N_{sub}-j)*S-i}\}_{i=0}^{S-1}$). In the rest of the paper, we use the term *sub-window* instead of slide for consistency. It is worth noting that the context within a sub-window is set to be sufficiently compact to ensure that the data points it contains adhere to the same underlying distribution.

For every sliding window in $\mathcal{X}$, the problem of unsupervised online concept drift detection is to identify whether the concept drift has occurred in a new sub-window $\mathbb{S}_t^{N_{sub}}$ compared with the existing data points in the current window $\mathbb{W}_t \setminus \mathbb{S}_t^{N_{sub}}$, without using any labels for drifts and downstream tasks (see Figure 1).

3.3 Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) [38] is a statistical measure that compares two probability distributions through a kernel, especially when the distributions are unknown and only samples are available. MMD evaluates the distance between the mean embeddings of two distributions in the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ [3]. Given $X \sim P$, $Y \sim Q$, and a kernel $f(\cdot)$, we have:

$$\mathrm{MMD}(P, Q) = \sup_{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq 1} \|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\|_2. \quad (1)$$

When we have samples $\{X_i\}_{i=1}^n$ from probability distribution P and $\{Y_i\}_{i=1}^n$ from probability distribution Q, the empirical Maximum Mean Discrepancy (MMD) can be written as:

$$\mathrm{MMD}(P, Q) = \sup_{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq 1} \|\frac{1}{n}\sum_{i=1}^{n} f(X_i) - \frac{1}{n}\sum_{i=1}^{n} f(Y_i)]\|_2. \quad (2)$$

Algorithm 1 Overall Procedure of MCD-DD

INPUT: Data stream $\mathcal{X} = \{x_t\}_{t=0}^{\infty}$, a sample set encoder set-enc() with an encoding function $f(\cdot)$.

```
Initialize the parameter θ of f(·) and the MCD threshold σ
for every sliding window 𝕎_t in 𝒳 do
    /* 1. Drift Detection */
    Obtain sample sets M^j for 𝕊_t^j ⊂ 𝕎_t
    Obtain concept representations {h^j = set-enc(M^j)}_{j=1}^{N_sub}
    Report drifted if MCD(P_t(𝕊_t^{N_sub}), P_t(𝕊_t^{N_sub-1})) > σ
    /* 2. Encoder Update */
    Obatin positive, weak/strong negative samples
    Calculate the loss ℒ by Eq. (14)
    Update θ by ℒ
    Update σ from the positive samples
end for
```

4 PROPOSED METHOD

4.1 Overview

The proposed method MCD-DD exploits *maximum concept discrepancy* for detecting concept drift, enabled by a deep encoder for data distribution embedding and contrastive learning for effective unsupervised training. The overall procedure of MCD-DD is illustrated in Figure 2 and outlined in Algorithm 1. For each new sliding window, MCD-DD follows the prequential evaluation scheme (i.e., test-and-train) [14]. First, MCD-DD samples sets of data points in sub-windows and embeds them through a sample set encoder. The discrepancy between sample sets from adjacent sub-windows is calculated, and if it exceeds a threshold, MCD-DD asserts that concept drift has occurred between them. Second, the sample set encoder is updated by considering the sliding window as the context for learning the latest concepts. Specifically, we treat sample sets from the same sub-window as positive pairs and construct negative pairs by leveraging temporal gaps between sub-windows and noise augmentation to distort the original distribution. In the meantime, MCD-DD dynamically adjusts the threshold for drift detection in the next sliding window by analyzing the positive sample pairs. Finally, the encoder is updated by aiming to minimize the distance between embeddings of positive sample pairs while simultaneously maximizing the distance between those of negative pairs.

4.2 Sample Set Encoder

We use a set of data points sampled in each sub-window to estimate its data distribution that represents a concept in the sub-window. Specifically, for a given sub-window $\mathbb{S}_t^j \subset \mathbb{W}_t$, we perform sampling without replacement from it to obtain a *sample set* of size m:

$$M^j = \{x_i \in \mathbb{S}_t^j\}_{i=1}^m. \quad (3)$$

The sample set is passed to a sample set encoder, denoted as set-enc(). It translates the probability distribution $P_t(\mathbb{S}_t^j)$ of the sub-window into the compact representation in the projected space, which we call *concept representation* $\mathbf{h}^j$ of $\mathbb{S}_t^j$:

$$\mathbf{h}^j = \text{set-enc}(M^j) = \frac{1}{m}\sum_{i}^{m} f(x_i \in M^j), \quad (4)$$

where $f(\cdot)$ is an encoding model with a deep neural network.

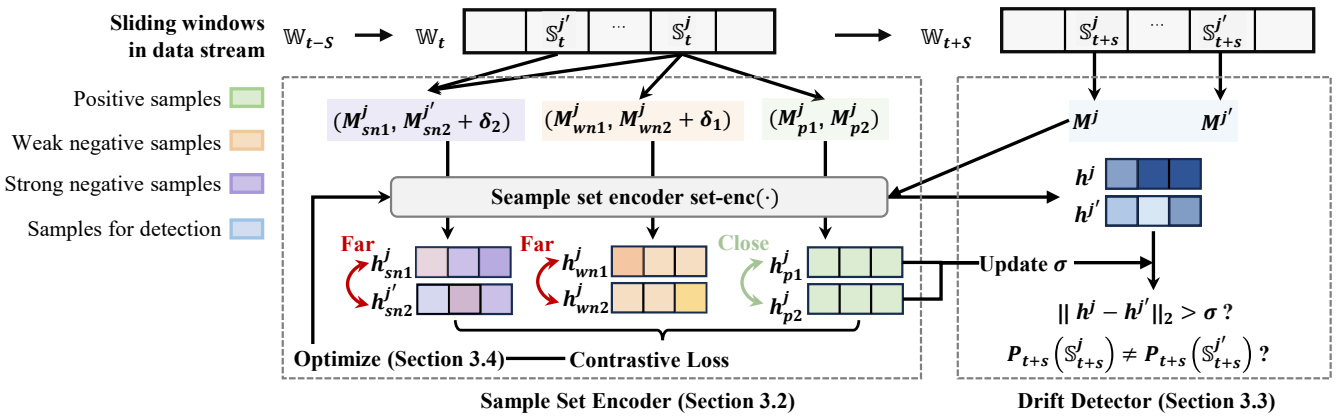


Figure 2: Overall procedure of MCD-DD. Sub-windows are encoded and compared to derive MCD for drift detection.

4.3 Drift Detector

For each sub-window $\mathbb{S}_t^j$ in a sliding window $\mathbb{W}_t$, we obtain the concept representations $\{h^j\}_{j=1}^{N_{sub}}$ by the sample set encoder. Then, to effectively quantify the distance between the concept representations from two adjacent sub-windows, we introduce a new measure *Maximum Concept Discrepancy (MCD)* inspired by MMD:

$$\mathrm{MCD}(P,Q) = \sup_{\|f\|_2 \le L} \|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\|_2, \tag{5}$$

where the function $f(\cdot)$ is approximated by a deep neural network and is constrained to be Lipschitz continuous [17]. This ensures the convergence of optimizing f and prevents MCD from becoming infinitely large. Moreover, it makes MCD between two sets of independent data from the same distribution bounded, providing the theoretical foundation for our drift detection.

When we have samples $M^j \sim P$ and $M^{j'} \sim Q$ by Eq. (3), the empirical MCD can be derived from Eq. (5) and Eq. (4) as:

$$\begin{aligned}\mathrm{MCD}(P,Q) &= \sup_{\|f\|_2 \le L} \|\frac{1}{m}\sum_{i=1}^{m} f(x_i \in M^j) - \frac{1}{m}\sum_{i=1}^{m} f(x_i \in M^{j'})]\|_2. \\ &= \sup_{\|f\|_2 \le L} \|\mathbf{h}^j - \mathbf{h}^{j'}\|_2\end{aligned} \tag{6}$$

Based on MCD, if the discrepancy between the data distributions of two adjacent sub-windows exceeds a given threshold σ,

$$\mathrm{MCD}(P_t(\mathbb{S}_t^j), P_t(\mathbb{S}_t^{j-1})) = \|\mathbf{h}^j - \mathbf{h}^{j-1}\|_2 > \sigma, \tag{7}$$

then we determine that a concept drift happened between them:

$$P_t(\mathbb{S}_t^j) \neq P_t(\mathbb{S}_t^{j-1}). \tag{8}$$

The threshold σ can be controlled dynamically using a bootstrapping strategy, allowing it to adjust to the varying distances between sample sets within the same concept. By analyzing the historical collection of MCD values between sample sets in the same sub-windows (i.e., those with identical data distributions), we adjust the threshold σ for each sliding window within a predefined statistical significance level (e.g., 0.05). It is worth noting that this historical MCD information can be obtained during the optimization of the encoder without necessitating additional computations.

4.4 Optimization

To optimize the sample set encoder, we employ contrastive learning [22] to enhance the separability of various concepts. The primary challenge lies in determining positive and negative sample pairs that are to be closer and further apart, respectively. This challenge is particularly pronounced in unsupervised scenarios, where we lack any prior information regarding the underlying concepts and true drifts. In MCD-DD, we exploit the concept of temporal coherence [37, 45], which is a fundamental characteristic observed in temporal data. It suggests that data points that are close in time are more likely to exhibit similar characteristics. We exploit this insight to create positive and negative samples used for optimization.

4.4.1 ***Preparing Positive Samples.*** MCD-DD generates a positive sample pair by choosing two sets of data points from the same sub-window, assuming that these sets follow the same distributions.

In each sub-window $\mathbb{S}t^j$, MCD-DD conducts sampling similar to Eq. (3) to select two sets of data points, which we treat as positive sample pairs. These pairs are denoted as M_{p1}^j and M_{p2}^j. To generate a diverse set of positive sample pairs, we repeat this sampling process k times, resulting in $\{M_{p1}^{j,i}\}_{i=1}^k$ and $\{M_{p2}^{j,i}\}_{i=1}^k$. Subsequently, the sample set encoder in Eq. (4) computes the embeddings for the k positive sample pairs, yielding $\{\mathbf{h}_{p1}^{j,i}\}_{i=1}^k$ and $\{\mathbf{h}_{p2}^{j,i}\}_{i=1}^k$.

4.4.2 ***Preparing Negative Samples.*** MCD-DD generates a negative sample pair to learn diversity in concept drift. Specifically, MCD-DD selects two sets of data points respectively from the different sub-windows that are temporally distant, whose distributions are likely to differ. MCD-DD also employs an efficient data augmentation technique to improve the generalization of negative pairs by introducing noise to each sampled data point:

$$x' = x + \delta, \tag{9}$$

where the noise δ is generated from a standard probability distribution (e.g., the Gaussian distribution $G(\mu, \epsilon)$). By incorporating these two principles, *temporal gap* (i.e., drift diversity) and *noise augmentation* (i.e., concept diversity), we prepare two types of negative sample pairs.

Weak negative samples: The first type of negative samples involves sampling pairs of data points from the same sub-windows but adding a small degree of noise into one of the sets. Specifically,

given a sub-window $\mathbb{S}t^j$, the k weak negative sample pairs are prepared:

$$\{M^{j,i}_{wn1}\}^k_{i=1} \text{ and } \{M^{j,i}_{wn2} + \delta_1\}^k_{i=1}, \tag{10}$$

where $\delta_1 \sim G(0, \epsilon_{small})$ and G is a Gaussian distribution. Finally, the corresponding two set sample embeddings are derived:

$$\{\mathbf{h}^{j,i}_{wn1}\}^k_{i=1} \text{ and } \{\mathbf{h}^{j,i}_{wn2}\}^k_{i=1}. \tag{11}$$

Strong negative samples: The second type of negative samples involves sampling pairs of data points from the two sub-windows that are temporally distant and more substantial noise is added to one of the sets. Specifically, given a sub-window $\mathbb{S}t^j$ and $\mathbb{S}t^{j'}$, the k strong negative sample pairs are prepared:

$$\{M^{j,i}_{sn1}\}^k_{i=1} \text{ and } \{M^{j',i}_{sn2} + \delta_2\}^k_{i=1}, \tag{12}$$

where $\delta_2 \sim G(0, \epsilon_{big})$. Similarly, the corresponding two set sample embeddings are derived:

$$\{\mathbf{h}^{j,i}_{sn1}\}^k_{i=1} \text{ and } \{\mathbf{h}^{j',i}_{sn2}\}^k_{i=1}. \tag{13}$$

Note that the temporal gap between two sub-windows can be adjusted within the context of a window (i.e., $1 \le |j - j'| \le N_{sub} - 1$). By default, MCD-DD adopts the largest temporal gap by setting $j = N_{sub}$ and $j' = 1$ so as to maximize the likelihood that the samples within each window exhibit distinct data distributions.

4.4.3 ***Learning Objective***. By putting the positive, weak negative, and strong negative samples altogether, the final loss for the sample set encoder is formulated in the form of InfoNCE loss [32]:

$$\mathcal{L} = \log \sum_{j=1}^{N_{sub}} \frac{\sum_{i=1}^k \exp(\text{MCD}^{j,k}_p)}{\sum_{i=1}^k (\exp(\text{MCD}^{j,k}_p) + \exp(\text{MCD}^{j,k}_{wn}) + \exp(\text{MCD}^{j,k}_{sn}))} + \lambda \sum_{i=1}^m (||\nabla_{x_i} f(x_i)||_2 - L)^2, \tag{14}$$

where $\text{MCD}^k_p = ||\mathbf{h}^{j,k}_{p1} - \mathbf{h}^{j,k}_{p2}||_2$, $\text{MCD}^k_{wn} = ||\mathbf{h}^{j,k}_{wn1} - \mathbf{h}^{j,k}_{wn2}||_2$, and $\text{MCD}^k_{sn} = ||\mathbf{h}^{j,k}_{sn1} - \mathbf{h}^{j',k}_{sn2}||_2$ are MCD values between the probability distributions of two sub-windows chosen for each sample type. The last term is the gradient penalty to ensure the L-Lipschitz continuity [21] of the encoding model $f(\cdot)$ where L is a constant and the coefficient λ is the regularization parameter.

5 THEORETICAL ANALYSIS

5.1 Upper Bound of MCD

Given two sample sets of data points drawn from the same distribution, we study the upper bound of MCD between the two sets, which can serve as a theoretical threshold for detecting concept drifts in two sub-windows for MCD-DD.

Theorem 1. Assume that the sets $\{X_i\}^n_{i=1}$ and $\{Y_i\}^n_{i=1}$ are independently and identically distributed (i.i.d.), both drawn from the probability distribution $p(x)$ with a mean μ and variance σ. If f is a Lipschitz continuous function with Lipschitz constant L, we have:

$$P(|\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)|) > G(1 - \frac{\alpha}{2})\sqrt{\frac{2}{n}}L\sigma) \le \alpha, \tag{15}$$

where $G(\cdot)$ is the standard Gaussian distribution function. Therefore, for a given significance level α, $G(1 - \frac{\alpha}{2})\sqrt{\frac{2}{n}}L\sigma$ is an upper bound for the MCD $|\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)|$.

Proof. By the Central Limit Theorem, $\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)$ converges to the Guassain distribution. Moreover, we have:

$$E(\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)) = 0. \tag{16}$$

Since f is a L-Lipschitz continuous function, we have:

$$Var(\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)) \le \frac{2L^2\sigma^2}{n}. \tag{17}$$

When we approximate the distribution of $\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)$ as the Gaussian distribution, we have:

$$P(|\frac{1}{n}\sum_{i=1}^n f(X_i) - \frac{1}{n}\sum_{i=1}^n f(Y_i)| > G(1 - \frac{\alpha}{2})\sqrt{\frac{2}{n}}L\sigma) \le \alpha. \tag{18}$$

□

Therefore, for the null hypothesis H_0: $\{X_i\}^n_{i=1}$ and $\{Y_i\}^n_{i=1}$ are drawn from the same probability distribution, given a significance level α, $G(1 - \frac{\alpha}{2})\sqrt{\frac{2}{n}}L\sigma$ can serve as the threshold for rejecting H_0. In the scenario where $\{X_i\}^n_{i=1}$ and $\{Y_i\}^n_{i=1}$ are multivariate random variables, hypothesis testing can similarly be conducted using the chi-squared distribution.

This theoretical bound can serve as a guide to set the threshold in a hypothesis testing framework. However, deriving the exact rejection threshold analytically may not always be feasible. As suggested in Section 4.3, the empirical threshold for rejecting the null hypothesis can be used by estimating statistics of historical MCD values meeting the hypothesis with a pre-defined significance.

5.2 Complexity of MCD-DD

We analyze the time complexity of MCD-DD mainly for sampling, training, and inference. Recall that we use a sliding window with N_{sub} sub-windows, k sets of m samples, and an encoder with the parameter size p and training epochs e. Since we sample in each sub-window, the time complexity for constructing positive and negative samples is $O(mkN_{sub})$. The time complexity for training the encoder is $O(mkep)$. For inference, since we need to calculate the differences of sample sets in each sub-window sequentially, the time complexity is $O(mkN^2_{sub})$. Finally, the total complexity is $O(mk(N^2_{sub} + ep))$. Since typically $p \gg e, m, k, N_{sub}$, the time complexity of MCD-DD is mostly controlled by the encoder complexity.

6 EXPERIMENTS

We conducted thorough experiments to evaluate the performance of MCD-DD on 7 synthetic data sets and 4 real-world data sets. The results are briefly summarized as follows.

- MCD-DD outperforms existing baselines in detecting concept drifts in terms of Precision, F1, and MCC scores and shows high interpretablity with varying drift types (Section 6.2).
- Through ablation analysis, the three sampling strategies introduced in contrastive learning for MCD are demonstrated to be effective (Section 6.3).

Table 1: Description of data streams used for evaluating drift detection performance.

Stream Category	Data Set	Composition	Instances	Dimension	Drift Type(s)
Synthetic (Primary)	GM_Sud	Gaussian Mixture	30,000	5	Sudden
	GM_Rec	Gaussian Mixture	30,000	5	Reoccurring
	GM_Grad	Gaussian Mixture	30,000	5	Gradual
	GM_Inc	Gaussian Mixture	30,000	5	Incremental
Synthetic (Complex)	GamLog_Sud	Gamma, Lognormal	30,000	5	Sudden
	LogGamWei_Sud	Lognormal, Gamma, Weibull	30,000	20	Sudden
	GamGM_SudGrad	Gamma,Gaussian Mixture	30,000	20	Sudden, Gradual
Real-World	INSECTS_Sud	Mosquito sensors with varying temperatures	52,848	33	Sudden
	INSECTS_Grad	Mosquito sensors with varying temperatures	24,150	33	Gradual
	INSECTS_IncreRec	Mosquito sensors with varying temperatures	79,986	33	Incremental, Reoccurring
	EEG	EEG readings for open/closed eye states	14,980	14	Sudden, Reoccurring

- The hyperparameters for MCD-DD are reasonably set by default and robust to the performance in most cases (Section 6.4).
- Visualizations of concept embeddings (Section 6.5) and threshold changes (Section 6.6) show the empirical efficacy of MCD-DD.

6.1 Experiment Setting

6.1.1 ***Data Sets.*** As summarized in Table 1, we used both *synthetic* and *real-world* data sets, each with known drift points.

The synthetic data sets are further divided into *primary drift tasks* and *complex drift tasks*, based on the complexity of drift detection. The primary drift tasks involve drifts created by altering the weights in Gaussian mixture models, leading to distinct distribution changes in low-dimensional data streams. Conversely, complex drift tasks include more challenging, higher-overlap distributions like Gamma, Lognormal, and Weibull, and consider multiple drift occurrences in higher-dimensional streams. A more comprehensive data generation process is provided in Appendix A.1.1.

Concept drift in real-world data sets tends to occur with more arbitrary and complex causes and thus presents greater predictive challenges. We selected real data sets known for their drift locations and types. *INSECTS* [40] introduces data sets with multiple types of drift through varying environmental temperatures to collect sensor reading values monitoring mosquitos. *EEG* [35] is a collection of EEG measurements of eye states recorded via camera, where open and closed eye states represent two different EEG distributions causing concept drift. Both data sets are widely used in relevant work considering concept drifts in real-world scenarios [31, 52]. Further details on the real-world data sets can be found in Appendix A.1.2.

For all data sets, a sliding window is employed from the beginning of each data set to simulate data streams. The sub-windows where the true drift initiates or is present are assigned a label of "Drift", while others are designated as "No Drift", formulating drift detection to a binary classification to facilitate evaluation.

6.1.2 ***Compared Algorithms.*** We chose popular drift detection algorithms that can be adopted for *unsupervised* and *online* concept drift detection; (1) *Kolmogorov-Smirnov Test (KS Test)* [34]: assessing whether one-dimensional distributions differ by calculating the supremum of the differences between their empirical distributions. For the multi-dimensional data, p-values were aggregated using the Bonferroni correction. (2) *Maximum Mean Discrepancy with a Gaussian Kernel (MMD-GK)* [19]: a kernel-based method for multi-dimensional two-sample testing that quantifies the disparity between two distributions by measuring their mean embeddings within the RKHS. In our experiments, we employ a Gaussian kernel to compute an unbiased estimate of the disparity. (3) *Least-Squares Density Difference (LSDD)* [6]: employing a linear-in-parameters Gaussian kernel function to estimate the distance between the probability density functions of two samples. (4) *Maximum Mean Discrepancy with a Deep Kernel (MMD-DK)* [29]: enhancing the traditional MMD approach by incorporating a deep kernel, where the kernel function is optimized using a subset of the data to maximize the test power. The detailed implementation and evaluation setting for the compared algorithms are provided in Appendix A.2

6.1.3 ***Evaluation Metrics.*** We used Precision, F1-score, and the Matthews Correlation Coefficient (MCC) [9] as metrics to gauge performance. Precision evaluates the proportion of true drifts correctly identified among all detected drifts, serving as a critical metric for assessing the effectiveness of a detector in avoiding false positives. The F1-score, a harmonic mean of precision and recall, serves as a balanced measure of a detector's effectiveness in identifying true drifts while considering both false positives and false negatives. The MCC[9], which encompasses all quadrants of the confusion matrix, is particularly reliable in scenarios involving rare but critical events, notably in cases of concept drift. We conducted 20 runs for each experiment and reported the average with standard deviations.

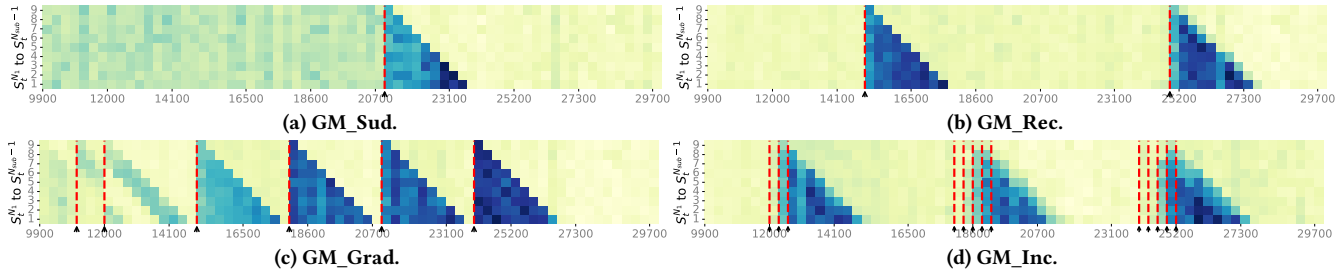
6.2 Drift Detection Accuracy

6.2.1 ***Synthetic Data Sets.*** As shown in Table 2, MCD-DD achieved the highest precision across all simulated data sets, with the scores being almost all 1 or very close to 1, demonstrating its *Eagle Eye* capability in drift point detection. Moreover, except for data sets with incremental drift, our method outperformed others in terms of F1-score and MCC as well.

Visualization of MCD in each window: In Figure 3, We further analyzed the concept drift capability of MCD-DD. We calculated MCD values between the most recent data distributions in each new sub-window ($S_t^{N_{sub}}$) and all preceding data distributions within the same window ($S_t^{N_1}$ to $S_t^{N_{sub}-1}$). The horizontal axis represents the locations of $S_t^{N_{sub}}$ at each time step t, and the vertical axis encompasses preceding sub-windows in the same window. Then, each cell indicates the learned MCD between sub-window pairs. For various types of concept drift, the heatmap patterns exhibit distinctive shapes: sudden drift manifests as lower triangular patterns starting

Table 2: Overall performance comparison (the best and second best results are in bold and underlined, respectively).

	MCD-DD			KS			MMD-GK			LSDD			MMD-DK		
Data set	*Pre.*	*F1*	*MCC*	*Pre.*	*F1*	*MCC*	*Pre.*	*F1*	*MCC*	*Pre.*	*F1*	*MCC*	*Pre.*	*F1*	*MCC*
GM_Sud	**1.00** (±0.00)	**1.00** (±0.00)	**1.00** (±0.00)	0.25 (±0.00)	0.40 (±0.00)	0.49 (±0.00)	0.32 (±0.07)	0.48 (±0.08)	0.56 (±0.06)	0.17 (±0.04)	0.30 (±0.05)	0.40 (±0.05)	0.25 (±0.12)	0.38 (±0.15)	0.47 (±0.12)
GM_Rec	**1.00** (±0.00)	**0.79** (±0.11)	**0.81** (±0.10)	0.40 (±0.00)	0.50 (±0.00)	0.50 (±0.00)	0.64 (±0.08)	0.78 (±0.06)	0.79 (±0.05)	0.60 (±0.16)	0.74 (±0.12)	0.76 (±0.10)	0.37 (±0.11)	0.51 (±0.12)	0.54 (±0.13)
GM_Grad	**1.00** (±0.00)	0.79 (±0.06)	0.79 (±0.06)	0.78 (±0.00)	0.78 (±0.00)	0.76 (±0.00)	0.80 (±0.05)	**0.89** (±0.03)	**0.88** (±0.03)	0.79 (±0.06)	0.78 (±0.03)	0.77 (±0.03)	0.70 (±0.11)	0.79 (±0.08)	0.78 (±0.09)
GM_Inc	**0.99** (±0.04)	0.44 (±0.09)	0.51 (±0.07)	0.38 (±0.00)	0.33 (±0.00)	0.27 (±0.00)	0.64 (±0.05)	**0.71** (±0.03)	**0.67** (±0.04)	0.57 (±0.06)	0.65 (±0.05)	0.61 (±0.06)	0.44 (±0.18)	0.37 (±0.13)	0.31 (±0.15)
GamLog_Sud	**1.00** (±0.00)	**1.00** (±0.00)	**1.00** (±0.00)	0.13 (±0.00)	0.22 (±0.00)	0.34 (±0.00)	0.11 (±0.01)	0.20 (±0.02)	0.31 (±0.02)	0.11 (±0.02)	0.20 (±0.03)	0.32 (±0.03)	0.18 (±0.06)	0.30 (±0.08)	0.40 (±0.07)
LogGamWei_Sud	**0.98** (±0.07)	**0.94** (±0.12)	**0.95** (±0.11)	0.40 (±0.00)	0.57 (±0.00)	0.62 (±0.00)	0.31 (±0.06)	0.46 (±0.06)	0.54 (±0.05)	0.26 (±0.10)	0.41 (±0.10)	0.49 (±0.09)	0.29 (±0.07)	0.44 (±0.08)	0.52 (±0.06)
GamGM_SudGrad	**0.98** (±0.04)	**0.99** (±0.02)	**0.99** (±0.02)	0.88 (±0.00)	0.93 (±0.00)	0.93 (±0.00)	0.66 (±0.05)	0.79 (±0.04)	0.79 (±0.03)	0.57 (±0.07)	0.72 (±0.06)	0.73 (±0.05)	0.61 (±0.11)	0.75 (±0.09)	0.76 (±0.08)
INSECTS_Sud	**0.55** (±0.06)	**0.59** (±0.07)	**0.55** (±0.08)	0.38 (±0.00)	0.53 (±0.00)	0.53 (±0.00)	0.38 (±0.02)	0.53 (±0.02)	0.53 (±0.02)	0.33 (±0.02)	0.49 (±0.02)	0.48 (±0.02)	0.38 (±0.06)	0.51 (±0.07)	0.48 (±0.08)
INSECTS_Grad	**0.18** (±0.15)	**0.28** (±0.23)	**0.32** (±0.28)	0.06 (±0.00)	0.11 (±0.00)	0.22 (±0.00)	0.06 (±0.00)	0.12 (±0.01)	0.23 (±0.01)	0.05 (±0.01)	0.10 (±0.01)	0.21 (±0.01)	0.06 (±0.04)	0.11 (±0.08)	0.18 (±0.14)
INSECTS_IncreRec	**0.26** (±0.05)	**0.37** (±0.07)	**0.40** (±0.07)	0.08 (±0.00)	0.15 (±0.00)	0.24 (±0.00)	0.08 (±0.00)	0.15 (±0.00)	0.24 (±0.00)	0.08 (±0.00)	0.14 (±0.00)	0.23 (±0.00)	0.10 (±0.01)	0.18 (±0.02)	0.25 (±0.04)
EEG	0.43 (±0.14)	0.23 (±0.10)	**0.12** (±0.09)	0.25 (±0.00)	0.40 (±0.00)	-0.05 (±0.00)	0.47 (±0.00)	0.63 (±0.00)	-0.11 (±0.00)	0.25 (±0.00)	0.40 (±0.00)	-0.00 (±0.00)	**0.48** (±0.00)	**0.64** (±0.00)	0.03 (±0.04)



(a) GM_Sud. (b) GM_Rec. (c) GM_Grad. (d) GM_Inc.

Figure 3: Heatmaps of MCD between sub-windows for primary synthetic data sets with drift indicators (red lines and arrows).

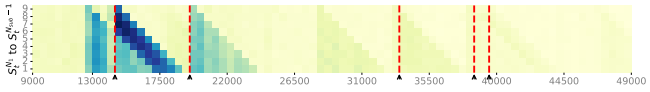


Figure 4: Heatmap for INSECTS_Sud with drift indicators.

at the drift point, reoccurring drift as two lower triangles, gradual drift progressively forms lower triangular patterns from top to bottom, and incremental drift results in fainter lower triangles due to more subtle distributional changes. We further investigated the model's performance with even more subtle and slower incremental drifts and real-world data sets in Appendix A.3. These demonstrations highlight the interpretive strength of MCD-DD in capturing the dynamics of drift occurrences.

6.2.2 ***Real-world Data Sets.*** As shown in Table 2, MCD-DD demonstrated superior performances over other baseline algorithms across a variety of drift scenarios within the *INSECTS*, including INSECTS_Sud (sudden drift), INSECTS_Grad (gradual drift), and INSECTS_IncreRec (incremental drift and reoccurring). Figure 4 shows the heatmap of MCD for INSECTS_Sud. These results highlight MCD-DD's robust capability in effectively detecting and adapting to different types of drift phenomena, ranging from abrupt changes to slow evolutions and cyclic variations. Despite exhibiting slightly weaker performance than MMD-DK when applied to the *EEG*, known for its high frequency of sudden drifts, it is noteworthy that MCD-DD still consistently achieved the highest MCC value.

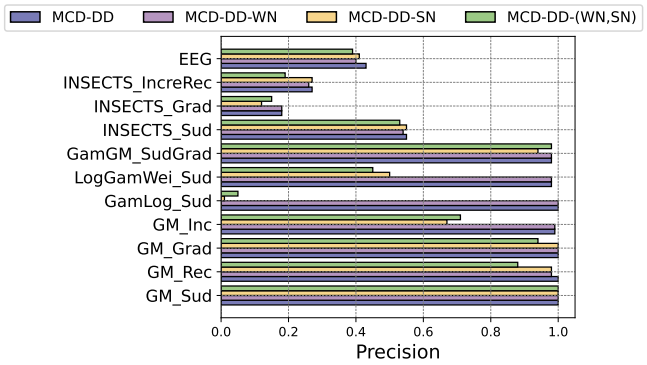
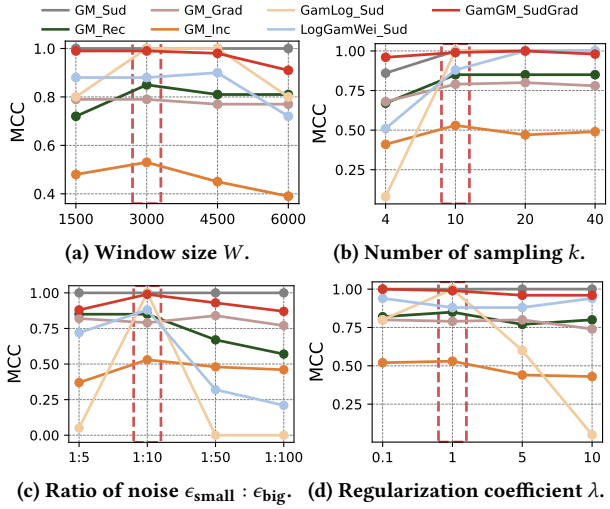


Figure 5: Ablation study of contrastive learning strategies.

6.3 Ablation Study

We conducted ablation studies on the strategies for obtaining positive and negative sample pairs. Specifically, we evaluated the performance of MCD-DD under various configurations: the complete MCD-DD setup, MCD-DD without weak negative sample pairs (i.e., MCD-DD-WN), MCD-DD without strong negative sample pairs (i.e., MCD-DD-SN), and MCD-DD utilizing only positive sample pairs, eliminating all negative samples (i.e., MCD-DD-(WN,SN)).

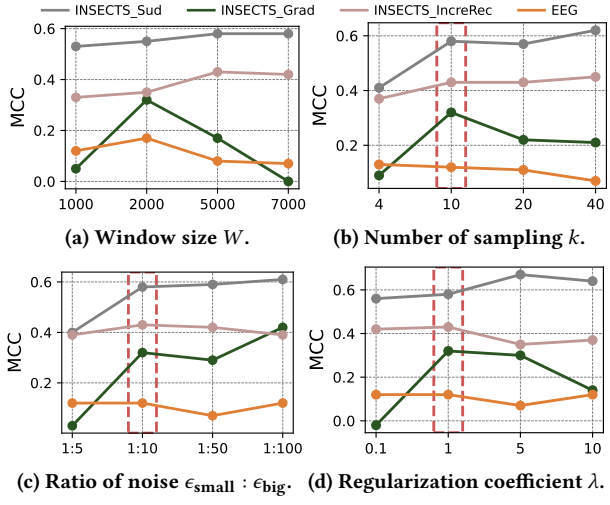
Figure 5 shows the results of precision, while the results of other metrics showed similar trends (Appendix A.3). In summary, MCD-DD achieved the highest precision across all data sets and the



(a) Window size W. (b) Number of sampling k.

(c) Ratio of noise $\epsilon_{\text{small}} : \epsilon_{\text{big}}$. (d) Regularization coefficient λ.

Figure 6: Sensitivity analysis results (default in red box).



(a) Window size W. (b) Number of sampling k.

(c) Ratio of noise $\epsilon_{\text{small}} : \epsilon_{\text{big}}$. (d) Regularization coefficient λ.

Figure 7: Sensitivity analysis results for real-world data sets.

absence of each type of negative sample pairs leads to lower performances in most cases. Specifically, removing weak negative sample pairs resulted in diminished effectiveness in detecting gradual, subtle concept drifts, like GM_Rec and EEG. Eliminating strong negative sample pairs adversely affected performance in more challenging drift detection scenarios (e.g., high overlap distributions, incremental drift), even failing to identify any drifts GamLog_Sud, along with notably poor performances in GM_Inc and INSECTS_Grad. Retaining only positive sample pairs yielded comparable results in simpler tasks but significantly underperformed in more complex simulated and real-world data sets compared to strategies incorporating negative sample pairs. These ablation study findings demonstrate the efficacy of the sampling strategies proposed particularly in dealing with complex drift scenarios.

6.4 Sensitivity Analysis

6.4.1 ***Effects of main Hyperparameters.*** We conducted a sensitivity analysis of the main hyperparameters used in MCD-DD: the *sliding window size* W, the *number of samples* k, the *degree of noise* ϵ, and the *regularization coefficient* λ. Figure 6 shows the MCC results on the synthetic data sets, and the results of other metrics showed similar trends. For W we varied it from half to two times the default value. Figure 6a demonstrates that MCD-DD shows comparable performances over varying window sizes with the peak performances at window sizes around the default value. Since the window size determines the context of current concepts used for training the encoder, the smaller size is preferable in practice for efficiency, as long as it includes temporally distant different distributions. Regarding the number k of sampling in each sub-window, Figure 6b shows that the higher number of sampling leads to increased performance. Nevertheless, sampling frequencies beyond the default value ($k = 10$) lead to only marginal improvements. In constructing negative sample pairs, the perturbation degree δ with ϵ_{small} and ϵ_{big} controls the relative degrees of noise for weak and strong negative samples. Figure 6c indicates that too low or too high ratios of small and big noise perform poorly, particularly in challenging cases (e.g., GamLog_Sud and LogGamWei_Sud). The default ratios of $1 : 10$ were demonstrated to be the most optimal in most cases. Finally, regarding the regularization coefficient λ for gradient penalty to ensure L-Lipschitz continuity in the optimization, Figure 6d shows that either too light or too heavy penalties can reduce the model's effectiveness on challenging cases, making $\lambda = 1$ a suitable choice.

Figure 7 shows the sensitivity analysis results on real-world data sets, demonstrating similar trends with the synthetic data sets. While the default values for W are not annotated in the figure given the varying lengths of each data set, it is found that setting W to 10% of the total length of each data set is a suitable choice.

Table 3: Window processing time (sec) over encoder sizes.

Hidden Size	Sampling Time	Training Time	Inference Time
50	0.0067	0.2106	0.0112
100	0.0065	0.2248	0.0117
150	0.0065	0.2297	0.0103
200	0.0069	0.2302	0.0120
250	0.0071	0.2456	0.0120
300	0.0067	0.2518	0.0121

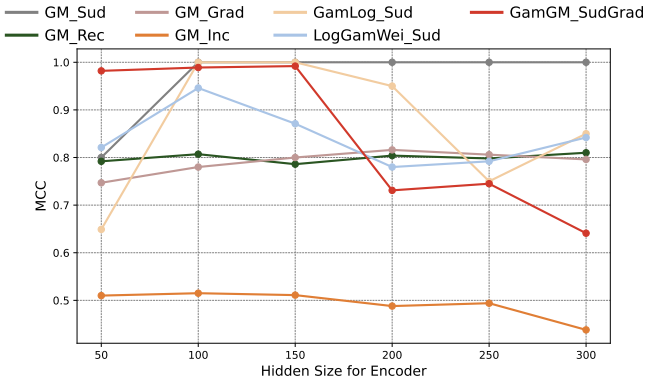
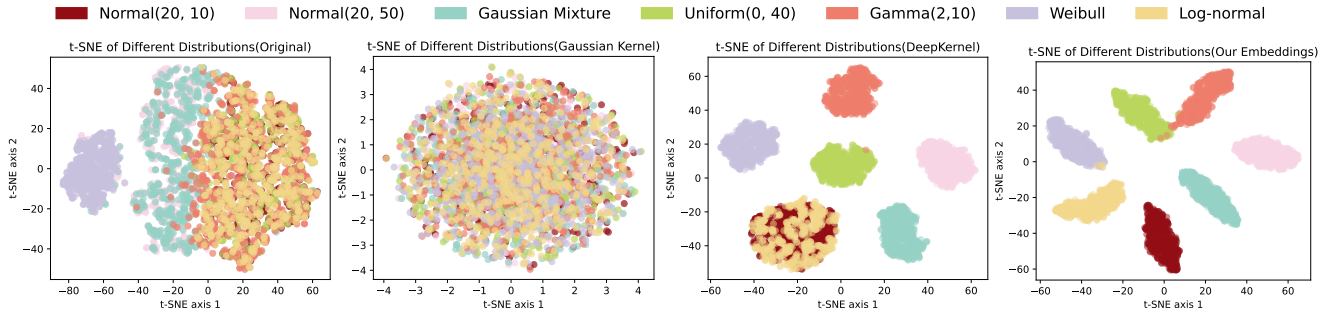


Figure 8: Detection accuracy over encoder sizes.

6.4.2 ***Effects of Encoder Size.*** We also conducted a sensitivity analysis of the encoder on processing time (Table 3) and detection accuracy (Figure 8) by varying its hidden size from 50 to 300. The results indicate that, in general, the encoder's hidden size does



(a) Orginal. (b) Gaussian Kernel. (c) Deep Kernel. (d) MCD-DD.

Figure 9: Comparative visualization of discriminative embedding capabilities for complex distributions.

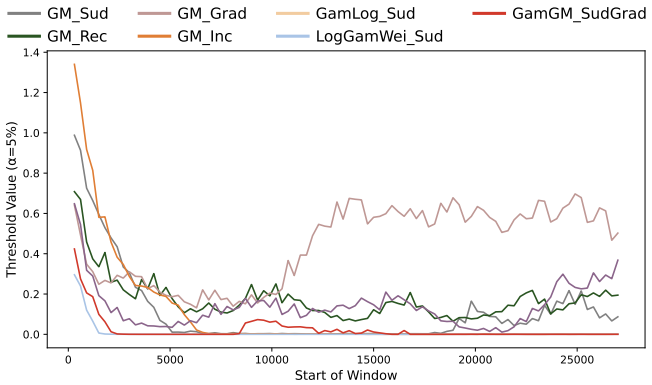


Figure 10: Changes of drift detection threshold σ over time.

not have a significant impact on performance. However, particularly in the data streams with complex drifts (e.g., GamLog_Sud or GamGM_SudGrad), the encoder with too low or too high hidden sizes degrades the detection accuracy owing to lacking expressiveness or overfitting in representing the complex concepts.

6.5 Qualitative Analysis

To more vividly showcase how MCD-DD maps original data distributions to the appropriate embedding space, facilitating the identification of varying distribution changes, we visualized the embedding spaces generated from the compared algorithms in two-dimensional space by t-SNE [42]. The seven challenging distributions used to generate data points are detailed in Appendix A.4. In Figure 9, points in each t-SNE plot demonstrate the representations of a set of data, including 500 data points. Different colors represent their different underlying probability distributions. Figure 9a demonstrates the difficulty of distinguishing distributions in the original space, showing highly intermingled clusters. Figure 9b shows that the Gaussian kernel mapping used by MMD-GK uniformly distributes all distributions across the space, aligning with the outcomes of prior studies [29]. While the deep kernel by MMD-DK achieved more separable embeddings, as shown in Figure 9c, MCD-DD exhibits even better separation, particularly in handling more complex distributions. This aspect justifies the superior performance of MCD-DD in concept drift detection across various scenarios.

6.6 Drift Detection Threshold Analysis

As discussed in Section 4.3, the drift detection threshold σ of MCD-DD is dynamically controlled by tracking the historical MCD values with a predefined statistical significance level (e.g., 0.05). To further understand its dynamicity, we analyzed the changes in the threshold over sliding windows in synthetic data sets with different drift types. Figure 10 shows that the thresholds were consistent over time in most cases, indicating that MCD is optimized robustly to varying drift types. GM_Grad notably exhibited a significant increase in the threshold values after the first drift was observed around 10,000, but it quickly converged to a constant value and the corresponding detection accuracy remained high as shown in Section 6.2.

7 CONCLUSION

We proposed MCD-DD, an unsupervised online concept drift detection method, exploiting a new measure called maximum concept discrepancy. MCD-DD leverages contrastive learning to obtain quality concept representations from sampled data points and optimize the maximum concept discrepancy. It is facilitated by sampling strategies based on temporal consistency and perturbations for robust optimization. The theoretical analysis demonstrated that MCD-DD can also be used within the hypothesis testing framework. Experimental results on multiple synthetic and real-world data sets showed that the proposed method achieves superior detection accuracy and higher interpretability than existing baselines.

For future work, we consider exploiting MCD histories learned throughout data streams. It can provide a systematic understanding of the patterns, duration, and strengths of concept drifts, as glimpsed in the heatmap visualization analyses. Further, assuming partial concept labels are available, adopting weak supervision philosophy can be promising. Estimating the degree of differences between partially labeled concepts can function as pseudo-labels to help us optimize the encoder more effectively.

ACKNOWLEDGMENTS

This work was supported by ICT Creative Consilience Program through the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government (MSIT) (IITP-2024-2020-0-01819) and the New Faculty Settlement Research Fund by Korea University. We thank Professor Yifan Cui from Zhejiang University for his valuable advice.

REFERENCES

[1] S. Agrahari and A. K. Singh. Concept drift detection in data stream mining: A literature review. *Journal of King Saud University-Computer and Information Sciences*, 34(10):9523–9540, 2022.

[2] F. Bayram, B. S. Ahmed, and A. Kassler. From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, 245:108632, 2022.

[3] A. Berlinet and C. Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011.

[4] A. Bifet and R. Gavalda. Learning from time-changing data with adaptive windowing. In *SDM*, pages 443–448. SIAM, 2007.

[5] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer. *Machine learning for data streams: with practical examples in MOA*. MIT press, 2023.

[6] L. Bu, C. Alippi, and D. Zhao. A pdf-free change detection test based on density difference estimation. *IEEE TNNLS*, 29(2):324–334, 2018.

[7] V. Cerqueira, H. M. Gomes, A. Bifet, and L. Torgo. STUDD: A student–teacher method for unsupervised concept drift detection. *Machine Learning*, 112(11):4351–4378, 2023.

[8] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607. PMLR, 2020.

[9] D. Chicco and G. Jurman. The advantages of the matthews correlation coefficient (MCC) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.

[10] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *UAI*, pages 258–267, 2015.

[11] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. *NeurIPS*, 14, 2001.

[12] I. Frias-Blanco, J. del Campo-Ávila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Díaz, and Y. Caballero-Mota. Online and non-parametric drift detection methods based on hoeffding's bounds. *IEEE TKDE*, 27(3):810–823, 2014.

[13] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Advances in Artificial Intelligence–SBIA 2004*, pages 286–295. Springer, 2004.

[14] J. Gama, R. Sebastião, and P. P. Rodrigues. On evaluating stream learning algorithms. *Machine Learning*, 90(3):317–346, 2013.

[15] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *CSUR*, 46(4):1–37, 2014.

[16] R. N. Gemaque, A. F. J. Costa, R. Giusti, and E. M. Dos Santos. An overview of unsupervised drift detection methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(6):e1381, 2020.

[17] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110:393–416, 2021.

[18] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two-sample-problem. *NeurIPS*, 19, 2006.

[19] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

[20] A. Gretton, K. Fukumizu, Z. Harchaoui, and B. K. Sriperumbudur. A fast, consistent kernel two-sample test. *NeurIPS*, 22, 2009.

[21] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *NeurIPS*, 30, 2017.

[22] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[23] T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. 2008.

[24] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *VLDB*, volume 4, pages 180–191. Toronto, Canada, 2004.

[25] D. Kim, H. Min, Y. Nam, H. Song, S. Yoon, M. Kim, and J.-G. Lee. Covid-EENet: Predicting fine-grained impact of covid-19 on local economies. In *AAAI*, volume 36, pages 11971–11981, 2022.

[26] Ł. Korycki and B. Krawczyk. Concept drift detection from multi-class imbalanced data streams. In *ICDE*, pages 1068–1079. IEEE, 2021.

[27] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos. Mmd gan: Towards deeper understanding of moment matching network. *NeurIPS*, 30, 2017.

[28] A. Liu, G. Zhang, and J. Lu. Fuzzy time windowing for gradual concept drift adaptation. In *FUZZ-IEEE*, pages 1–6. IEEE, 2017.

[29] F. Liu, W. Xu, J. Lu, G. Zhang, A. Gretton, and D. J. Sutherland. Learning deep kernels for non-parametric two-sample tests. In *ICML*, 2020.

[30] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE TKDE*, 31(12):2346–2363, 2018.

[31] K. Miyaguchi and H. Kajino. Cogra: concept-drift-aware stochastic gradient descent for time-series forecasting. In *AAAI*, 2019.

[32] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[33] C. Raab, M. Heusinger, and F.-M. Schleif. Reactive soft prototype computing for concept drift streams. *Neurocomputing*, 416:340–351, 2020.

[34] S. Rabanser, S. Günnemann, and Z. Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *NeurIPS*, 32, 2019.

[35] O. Roesler. EEG Eye State. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C57G7J.

[36] J. Shao, Z. Ahmadi, and S. Kramer. Prototype-based learning on concept-drifting data streams. In *KDD*, pages 412–421, 2014.

[37] Y. Shin, S. Yoon, S. Kim, H. Song, J.-G. Lee, and B. S. Lee. Coherence-based label propagation over time series for accelerated active learning. In *ICLR*, 2021.

[38] A. J. Smola, A. Gretton, and K. Borgwardt. Maximum mean discrepancy. In *ICONIP*, pages 3–6, 2006.

[39] X. Song, M. Wu, C. Jermaine, and S. Ranka. Statistical change detection for multi-dimensional data. In *KDD*, pages 667–676, 2007.

[40] V. M. A. Souza, D. M. Reis, A. G. Maletzke, and G. E. A. P. A. Batista. Challenges in benchmarking stream learning algorithms with real-world data. *Data Mining and Knowledge Discovery*, 34:1805–1858, 2020.

[41] P. Trirat, S. Yoon, and J.-G. Lee. MG-TAR: multi-view graph convolutional networks for traffic accident risk prediction. *IEEE Transactions on Intelligent Transportation Systems*, 24(4):3779–3794, 2023.

[42] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[43] A. Van Looveren, J. Klaise, G. Vacanti, O. Cobb, A. Scillitoe, R. Samoilescu, and A. Athorne. Alibi detect: Algorithms for outlier, adversarial and drift detection, 2019.

[44] Z. Wang, Y. Chen, C. Zhao, Y. Lin, X. Zhao, H. Tao, Y. Wang, and L. Khan. Clear: Contrastive-prototype learning with drift estimation for resource constrained stream mining. In *TheWebConf*, pages 1351–1362, 2021.

[45] Z. Wang, Z. Gao, L. Wang, Z. Li, and G. Wu. Boundary-aware cascade networks for temporal action segmentation. In *ECCV*, pages 34–51. Springer, 2020.

[46] Z. Wang, L. Liu, Y. Kong, J. Guo, and D. Tao. Online continual learning with contrastive vision transformer. In *ECCV*, pages 631–650. Springer, 2022.

[47] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning. In *AISTATS*, pages 370–378. PMLR, 2016.

[48] S. Xu and J. Wang. Dynamic extreme learning machine for data stream classification. *Neurocomputing*, 238:433–449, 2017.

[49] S. Yoon, H. P. Chan, and J. Han. PDSum: Prototype-driven continuous summarization of evolving multi-document sets stream. In *Proceedings of the ACM Web Conference 2023*, pages 1650–1661, 2023.

[50] S. Yoon, D. Lee, Y. Zhang, and J. Han. Unsupervised story discovery from continuous news streams via scalable thematic embedding. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 802–811, 2023.

[51] S. Yoon, J.-G. Lee, and B. S. Lee. NETS: extremely fast outlier detection from a data stream via set-based processing. *Proceedings of the VLDB Endowment*, 12(11):1303–1315, 2019.

[52] S. Yoon, Y. Lee, J.-G. Lee, and B. S. Lee. Adaptive model pooling for online deep anomaly detection from a complex evolving data stream. In *KDD*. ACM, Aug. 2022.

[53] S. Yoon, Y. Meng, D. Lee, and J. Han. SCStory: Self-supervised and continual online story discovery. In *TheWebConf*, pages 1853–1864, 2023.

[54] H. Yu, J. Li, J. Lu, Y. Song, S. Xie, and G. Zhang. Type-LDD: A type-driven lite concept drift detector for data streams. *IEEE TKDE*, 2023.

[55] H. Yu, T. Liu, J. Lu, and G. Zhang. Automatic learning to detect concept drift. *arXiv preprint arXiv:2105.01419*, 2021.

A APPENDIX

A.1 Data Sets

A.1.1 ***Synthetic Data Sets.*** The synthetic data sets used for model performance evaluation feature two types of drifts: **primary** and **complex**. Primary drift tasks involve concept drifts that are relatively easy to distinguish within the data stream. Complex drift tasks, however, present higher distribution similarity, increased dimensionality, and a greater number of drift events, making them more challenging to discern.

For primary tasks, drift is simulated by adjusting the weighting of two Gaussian distributions, G_1 and G_2. Both distributions conform to a 5-dimensional Gaussian distribution with a mean vector $\mu = [20, 20, 20, 20, 20]$ and covariance matrices $\Sigma_1 = 10^2\mathbf{I}$ and $\Sigma_2 = 50^2\mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix. These distributions form the basis for diverse Gaussian mixtures, with the weighting factor p controlling the proportion of each distribution in the mixture. The mixture model is represented by the equation: $\mathcal{N}(\mu, \Sigma_1) \times p + \mathcal{N}(\mu, \Sigma_2) \times (1 - p)$. Each primary drift task involves a data set of 30,000 instances. Within this framework, different types of primary drift tasks are simulated by varying p:

Primary Task 1-GM_Sud: Initially, p is set to 0.2. To induce a sudden drift at the 21,000th instance, p is shifted to 0.8, significantly changing the mixture's composition and simulating a sudden change in the underlying data structure.

Primary Task 2-GM_Rec: Initially, p is set to 0.8, giving prominence to G_1. At the 15,000th instance, p changes to 0.2, thus shifting the mixture's balance towards G_2. Finally, at the 25,000th instance, p returns to 0.8, reinstating the initial distribution emphasis. This pattern creates a reoccurring drift in the data stream.

Primary Task 3-GM_Inc: Initially set at 0.2, the weighting factor p undergoes specific adjustments to introduce incremental drifts at predetermined intervals within the data set. Specifically, p linearly increases from 0.2 to 0.8 between the 12,000th and 12,600th instances, then decreases back to 0.2 between the 18,000th and 19,200th instances, and finally increases again to 0.8 between the 24,000th and 25,200th instances. Outside these intervals, p remains constant, ensuring no drift occurs in the intervening segments. This design results in multiple incremental drifts across the data set.

These linear transitions facilitate incremental drifts, modifying the data distribution across two Gaussian components.

Primary Task 4-GM_Grad: In the gradual drift task, p fluctuates between 0.2 and 0.8. The drift periods where $p = 0.8$ occur during the intervals (10000, 11000), (12001, 15000), and (18000, 21000). Conversely, in the intervening periods (11001,12000), (15001,18000), and (21001,24000), p reverts to 0.2. This oscillation creates a gradual drift pattern by alternating phases of drift and stability.

Complex drift tasks, conversely, involve a mixture of distributions like Gamma, Lognormal, and Weibull. These distributions exhibit substantial overlap in their probability density functions and possess similar statistical characteristics, leading to lower discriminability and making the detection of drifts more challenging. These tasks are further compounded by introducing multiple drifts of different natures within the data stream. Also, each complex drift task involves a data set of 30,000 instances, where every dimension conforms to the same distribution pattern, ensuring consistency across the multidimensional data space.

Complex Task 1-GamLog_Sud: Initially, data is generated from a Gamma distribution (Gamma(1.5, 20)) across 5 dimensions. At the 21,000th instance, there is a sudden shift to a 5-dimensional Log-normal distribution with parameters $\mu = \log(30) - 0.5$ and $\sigma = 0.5$. This transition represents a complex and sudden drift, with the overlap between the two distributions making the drift challenging to identify.

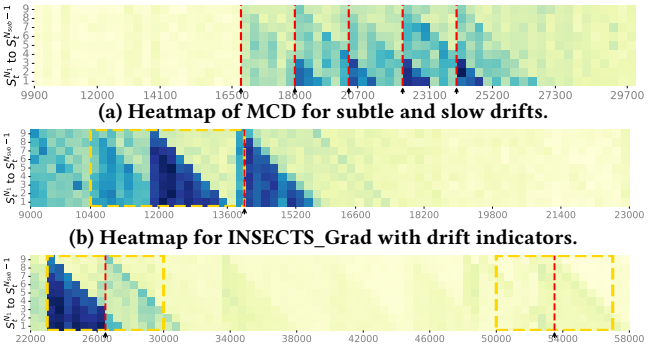
Complex Task 2-LogGamWei_Sud: The data stream, consisting of 20 dimensions, initially follows a Log-normal distribution (Lognormal($\log(30) - 0.5, 0.5$)). At the 15,000th instance, there is a sudden drift to a Gamma distribution (Gamma(1.5, 20)), and at the 24,000th instance, it transitions to a Weibull distribution (Weibull(1.5, 20)). These successive drifts add higher complexity.

Complex Task 3-GamGM_SudGrad: This data stream consists of 20 dimensions, each following the same distribution. Initially, the data follows a Gamma distribution (Gamma(2, 10)). At the 11,000th instance, a sudden drift occurs, transitioning the data to a Gaussian mixture. Subsequently, the task experiences gradual drifts, where the weighting factor p alternates between 0.2 and 0.8 during specific intervals. This alternation leads to shifting dominance between two Gaussian distributions ($\mathcal{N}(20, 10^2)$ and $\mathcal{N}(20, 50^2)$), creating a complex pattern of both sudden and gradual drifts.

A.1.2 ***Real-World Data Sets.*** Here we provide detailed descriptions of the real-world data sets employed to evaluate the performance of our detector: *INSECTS* and *EEG*. These data sets are instrumental in assessing how effectively the detector adapts to real-world concept drift scenarios.

INSECTS: The INSECTS data sets consist of optical sensor readings obtained from monitoring mosquitoes. Concept drifts are induced by varying temperatures, which affect the insects' activity levels in alignment with their circadian rhythms. This data set offers a dynamic setting of concept drifts. We utilized three specific data sets from the collection, each representing one or more distinct types of drift: (i) **INSECTS_Sud:** This subset exhibits five sudden drifts, initiated at a temperature of 30°C, with a sudden shift to 20°C, and subsequently to approximately 35°C among other changes. The stream captures several rapid transitions in temperature, illustrating sudden concept drifts throughout. (ii) **INSECTS_Grad:** Illustrating both gradual and incremental drifts, this data set simulates a scenario where temperatures slowly transition over time, presenting a nuanced evolution of environmental conditions. (iii) **INSECTS_IncreRec:** Features a unique pattern of reoccurring incremental drifts, where temperature gradually increases or decreases in cycles. This data set is pivotal for studying the model's performance in scenarios where drift patterns repeat over time, challenging the detection mechanism with both incremental and reoccurring drift phenomena.

EEG: The EEG data set encompasses multivariate time-series data from a single continuous EEG recording using the Emotiv EEG Neuroheadset over a span of 117 seconds. Eye states were captured through video recording concurrent with the EEG data collection and were subsequently annotated manually to indicate moments of eye closure ("1") and eye opening ("0"). This data set provides a sequential record of neurological activity, with values arranged in the order they were measured, presenting a unique challenge for detecting shifts in physiological states over time.



(a) Heatmap of MCD for subtle and slow drifts.

(b) Heatmap for INSECTS_Grad with drift indicators.

(c) Heatmap for INSECTS_IncreRec with drift indicators.

Figure 11: Heatmaps of MCD for other data sets.

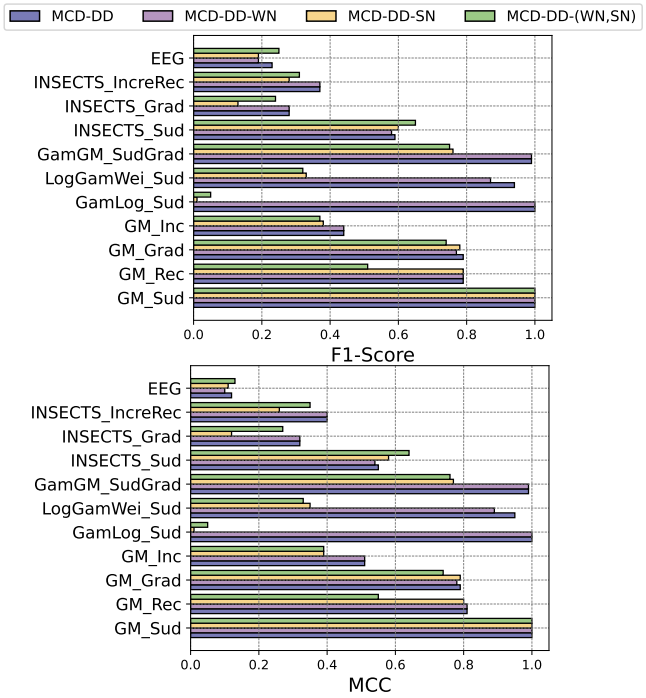


Figure 12: Ablation study results of F1 and MCC.

Table 4: Performance with unidimensional streams.

	MCD-DD			ADWIN		
Data set	*Pre.*	*F1*	*MCC*	*Pre.*	*F1*	*MCC*
INSECTS_Sud	0.585	0.503	0.472	0.667	0.244	0.332

A.2 Implementation Details

A.2.1 ***Implementation of compared algorithms.*** For MCD-DD, we set $m = 30$ (or 50 for a larger sub-window size such as in INSECTS_IncreRec), $k = 10$, $\lambda = 1$, $\epsilon_{small} = 1$, $\epsilon_{big} = 10$, and $L = 1$. The encoder function was implemented as a two-layer MLP with a ReLU activation function. For synthetic data sets, the two-layer encoder features 100 units in both hidden and output layers. For the INSECTS data set, reflecting its more complex drift types and higher dimensionality, the encoder dimensions are increased to 200 (hidden) and 150 (output). For the EEG data set, which involves smaller window sizes, the dimensions are adjusted to 150 (hidden) and 100 (output). In all data sets, the encoder function has been trained for a single epoch with a learning rate of 0.005 for every sliding window. For the implementation of baselines, we referred to *alibi-detect* [43] and used sufficiently high permutations (200) for the statistical method and the default projection and the best epochs for the learning-based method.

A.2.2 ***Evaluation Scheme.*** For a fair comparison, all algorithms followed the prequential evaluation scheme [14] and used the same sliding window with the window size W of 10% of the total length of each data set and $N_{sub} = 10$. Each algorithm compares every new sub-window with the preceding one in a window to identify concept drift with a significance level of 0.05 if applicable.

A.2.3 ***Computing Platform.*** All experiments were conducted on a Linux server equipped with an Intel Xeon CPU @ 2.20GHz, 12GB RAM, and 226GB of storage where Ubuntu 22.04 LTS, Python 3.10, and PyTorch 2.1.0+cu122 were installed. An NVIDIA Tesla V100-SXM2-16GB GPU was used for the deep learning-based algorithms.

A.3 Additional Performance Analysis Results

For subtle and slow drift types, we have considered adaptively adjusting the training process to achieve better detection outcomes. In this regard, an incremental drift was introduced by linearly transitioning the weighting factor p from 0.0 to 1.0 between the 15,000th and 24,000th instances. Specifically, for this incremental drift scenario, a specialized training strategy was implemented that alternates between exclusively using positive sample pairs and a mix of both positive and negative sample pairs, aimed at better adapting to and learning the nuanced shifts present. Keeping other parameters constant, MCD-DD achieved a precision of 0.80, significantly outperforming other baseline algorithms with a maximum precision of 0.55. Similar to the visualization discussed in Section 6.2, Figure 11a illustrates the progression of MCD in this scenario.

In addition, Figures 11b and 11c show the heatmaps of MCD for the other two types of INSECTS data sets. While it is challenging to accurately identify the exact starting points for gradual and incremental drifts in real data streams, the heatmap shows higher MCD values around the true drift indicators (in the yellow boxes). Figure 12 shows the ablation study results with F1 and MCC. Table 4 compares the performance of MCD-DD with ADWIN [4] adopted for an unsupervised setting with unidimensional data stream.

A.4 Details of Qualitative Analysis

For the seven distributions, Dist1 is a normal distribution $\mathcal{N}(20, 10^2)$, generating samples in a 5-dimensional space. Dist2 is a normal distribution with increased variance $\mathcal{N}(20, 50^2)$. Dist3 is a mixture of two normal distributions, giving each sample a 50% probability of originating from either $\mathcal{N}(20, 10^2)$ or $\mathcal{N}(20, 50^2)$. Dist4 is a uniform distribution spanning from 0 to 40, $\mathcal{U}(0, 40)$. Dist5 follows a gamma distribution with shape and scale parameters set to 2 and 10, respectively, Gamma$(2, 10)$. Dist6 utilizes a Weibull distribution with shape and scale parameters of 2 and 20, Weibull$(2, 20)$. Lastly, Dist7 is a log-normal distribution chosen to approximate a mean close to 20 by setting a standard deviation of 0.5 and a location parameter μ to $\log(20) - \frac{0.5^2}{2}$, Lognormal$(\mu, 0.5^2)$.