Addressing Intermediate Verification Latency in Online Learning Through Immediate Pseudo-labeling and Oriented Synthetic Correction

Zixin Zhong[1] , Liyan Song[2*] , Fengzhen Tang[3] , Bo Yuan[1]
12232431@mail.sustech.edu.cn, songly@hit.edu.cn, tangfengzhen@sia.cn, yuanb@sustech.edu.cn
[1] Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China
[2] Faculty of Computing, Harbin Institute of Technology, Harbin, China
[3] State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China

***Abstract*—In non-stationary data streams, the challenges of concept drift are further compounded by the issue of Intermediate Verification Latency (IVL), which can impede timely model adaptation. IVL refers to the finite delay between the arrival of data features and their corresponding labels. This delay could pose a significant challenge in adapting models to new concepts, ultimately hindering predictive performance. However, existing IVL approaches exhibit certain limitations. Some approaches passively wait for delayed labels, thereby overlooking temporarily unlabeled data. Other approaches employ pseudo-labeling for immediate model updates, but may risk losing valuable information when reverting model states to rectify previous pseudo-labeling mistakes. To overcome these limitations, we propose a novel approach called Micro-cluster based Immediate Pseudo-Labeling with Oriented Synthetic Correction (MIPLOSC). MIPLOSC leverages micro-cluster systems to effectively capture data distributions, thus facilitating its two core components: immediate pseudo-labeling and oriented synthetic correction. The immediate pseudo-labeling mechanism facilitates immediate utilization of temporarily unlabeled data, and the oriented synthetic correction mechanism enables finer-grained rectification from previous erroneous pseudo-labels and concept drift, minimizing the loss of learned information. Experimental studies validated the effectiveness of MIPLOSC in addressing IVL, demonstrating its superiority over competing methods in both space consumption and predictive performance across varying degrees of label delay.**

***Index Terms*—online learning, concept drift, verification latency, label delay**

I. Introduction

Numerous real-world applications such as stock prediction, weather forecasting, and Internet of Things (IoT) systems have encountered a substantial surge in data volume. This influx of data is characterized by its continuous and sequential arrival over time, reflecting the inherent dynamism of real-world activities. These ongoing data sequences are commonly referred to as a *data stream* and can be represented as $S = \{(\mathbf{x}_t, y_t)|\ t = 1, 2, \ldots\}$, where $\mathbf{x}_t \in \mathbb{R}^d$ is a d-dimensional data feature, and $y_t \in \mathcal{Y}$ denotes the corresponding label [1]. Each data $(\mathbf{x}_t, y_t)$ is drawn from an underlying joint probability distribution $P_t(\mathbf{x}, y)$ [2]. The dynamic nature of data streams will lead to changes in the underlying distribution, known as *concept drift* [3]. In the presence of concept drift, observations generated at different time steps, denoted as t and $t + \Delta$, may originate from different joint probability distributions, represented as $P_t(\mathbf{x}, y) \neq P_{t+\Delta}(\mathbf{x}, y)$ [2]. Concept drift poses a significant challenge for data modeling systems as it requires models to adapt to the changing distributions [4].

Many real-world applications also suffer from a temporal gap between the arrival of data features and their corresponding labels, known as *verification latency* [5], [6]. This latency can be denoted as $\delta_t = T(y_t) - T(\mathbf{x}_t)$, where $T(\cdot)$ represents the respective arrival time [7]. Verification latency arises due to factors such as labeling cost, data transmission cost, and inherent characteristics of the applications. This paper specifically focuses on Intermediate Verification Latency (IVL), where $0 < \delta_t < \infty$ [6]. An example of IVL is evidenced in the context of Tool Condition Monitoring (TCM) within IoT systems. TCM aims to detect equipment tool issues by modeling the relationship between tool conditions and sensor data, such as cutting force, vibration, and power signals [8]. However, the manual labeling process for tool condition data is expensive and time-consuming. Consequently, acquiring labels of tool condition data would suffer from significant delays, thereby resulting in IVL [9]. The presence of IVL would impede the timely adaptation of models to new concepts, resulting in performance decline. In the context of TCM, IVL inhibits the model's ability to promptly response to real-time changes in tool conditions during the manufacturing process. As a result, the timely decision-making capabilities of IoT systems would be hindered, emphasizing the importance of addressing the issue of IVL.

Existing approaches for addressing IVL can be classified into two categories: the wait-until-arrival approach [10]–[12] and the pseudo-labeling with rollback approach [9], [13]. Both approaches involve a trade-off between encouraging immedi-

* The corresponding author.

ate model updates and minimizing the mistakes introduced by those updates. The wait-until-arrival strategy defers model updates until the delayed labels become available. This approach prioritizes the availability of true labels before making any model updates. Conversely, the pseudo-labeling with rollback strategy assigns pseudo-labels to temporarily unlabeled data, allowing for immediate model updates. However, this strategy may introduce errors in the process of pseudo-labeling, primarily due to the estimation of labels for unlabeled data. To mitigate the impact of these erroneous updates, a rollback mechanism is typically accompanied, which identifies and corrects mistakes by reverting the model to its previous state.

However, these approaches have their own limitations. The wait-until-arrival strategy delays the update process and overlooks the valuable information contained in unlabeled data. On the other hand, the pseudo-labeling with rollback strategy may discard valuable data information during the rollback procedure and requires substantial storage for historical data and model states. To overcome these limitations, three key questions need to be answered:

1) How to effectively use temporarily unlabeled data?
2) How to identify and rectify mistaken updates with less loss of learned information?
3) How to achieved these goals while reducing spatial cost?

We propose a novel technique called MIPLOSC (Micro-cluster based Immediate Pseudo-Labeling with Oriented Synthetic Correction) to response to these questions. MIPLOSC builds upon the wait-until-arrival strategy, and leverages the micro-clustering technique to support model updates. Micro-clustering systems offer a way to summarize and trace the data distribution while maintaining a lower spatial cost compared to storing all data points [14], [15], thus addressing question 3. Regarding question 1, MIPLOSC utilizes information from the micro-cluster system to pseudo-label unlabeled data and assess their reliability based on their geometric location. Only reliable pseudo-labeled data will be used for immediate model updates. As for question 2, micro-cluster systems enable the identification of mistakes introduced by previous pseudo-labels and potential concept drift. This is accomplished by scrutinizing discrepancies between delayed true labels and the class label information maintained by the micro-clusters. When mistakes in previous pseudo-labeling are detected, an oriented synthetic correction approach is activated. This correction procedure incorporates synthetic data to facilitate model correction while preserving valuable information that remains unaffected by erroneous pseudo-labels and concept drift.

The main contributions of this paper are summarized below:

1) We propose a novel approach called MIPLOSC to address IVL in online learning scenarios. MIPLOSC incorporate online micro-clustering to minimize spatial cost and provide a concise summary of the data distribution. On top of that, immediate pseudo-labeling and oriented synthetic correction mechanisms are implemented. Together, these techniques enhance model performance.
2) Based on a diverse range of synthetic and real-world datasets, we conduct a systematic experimental study to evaluate the performance of MIPLOSC and the effectiveness of its individual component in addressing IVL.

II. Related Work

A. Methods Directly Dealing with IVL

As briefly mentioned in Section I, there have been a limited number of existing approaches tackling IVL, which can be categorized into two types: the wait-until-arrival strategy and the pseudo-labeling with rollback strategy.

The wait-until-arrival strategy has been employed by several studies including ECSMiner [10], FuzzCND [11], and EFuzzCND [12]. This strategy is favored for its simplicity when dealing with class evolution in the presence of IVL. For these reasons, our method is built upon this strategy. By adopting the wait-until-arrival strategy, these studies prioritize the importance of true labels, ensuring that model updates occur only upon the arrival of delayed labels. This approach effectively avoids introducing additional noise into the model. However, this approach has its limitations. While the strategy postpones adaptation to new concepts, it may overlook potentially valuable information inherent in the unlabeled data. In summary, the wait-until-arrival strategy strikes a balance by eliminating labeling noise that may delay the adaptation to new concepts.

The pseudo-labeling with rollback strategy was proposed in SkipE-RNN [13] and SERMON [9]. This strategy effectively leverages the information contained within unlabeled data by incorporating pseudo-labels. In SkipE-RNN and SERMON, the label mapping component exploits the temporal dependency of the data stream to aligns unlabeled data with the closest historical data in hidden space. Subsequently, the label of the matched historical data is assigned to unlabeled data as a pseudo-label, enabling an immediate model update. However, the adoption of pseudo-labels may introduce label noise. To mitigate this concern, SkipE-RNN implements a rollback mechanism, which is further enhanced in SERMON. The core idea is as follows: when detecting a previous erroneous update, a penalty is incorporated into the loss function. This encourages the model parameters to revert to a state prior to the erroneous update, thereby preventing the propagation of inaccuracies. While the strategy ensures the immediate model updates, it requires additional storage for detecting erroneous updates, historical data matching, and implementing model state rollback. Consequently, substantial space consumption is necessary. Furthermore, a blunt rollback of the model state may lead to the loss of valuable information contained in the data that is not affected by erroneous pseudo-labels and concept drift. Overall, the pseudo-labeling with rollback strategy requires careful reconsideration in handling detected erroneous updates and managing substantial space consumption.

B. Potential Methods Indirectly Dealing with IVL

In addition to the methods specifically designed for IVL, active learning and online semi-supervised learning techniques have the potential to be adapted and utilized to address IVL in specific scenarios [7]. These techniques, originally developed for other purposes, can potentially be modified and tailored to address the issues of IVL.

Active learning approaches [16], [17] involve selecting unlabeled data from the data stream based on heuristics and acquiring their true labels from an oracle. Subsequently, these labeled instances are used to update the model. However, it is worth noting that active learning techniques would require additional labeling costs and may not be feasible if the demand for requested labeling cannot be fulfilled.

Online semi-supervised learning approaches also have the potential in addressing IVL [7]. These methods leverage temporarily unlabeled data in various ways to improve model performance. Tree-based methods, such as REDLLA [18] and SUN [19], employ incremental decision trees for model construction. In these methods, data within a leaf node are clustered, and labels of unlabeled data are inferred through majority voting. Graph-based methods, such as [20]–[22], incorporate both labeled and temporarily unlabeled data in the graph maintenance process. Label propagation algorithms are then employed to infer the labels of unlabeled test examples. Micro-cluster based methods, such as the approach proposed by Ud Din et al. [15], maintain a collection of labeled and unlabeled micro-clusters. The labels of unlabeled data are then inferred using k-NN based on these micro-clusters. The existing online semi-supervised learning methods are primarily designed for partially labeled scenarios rather than IVL. These methods operate under the assumption that true labels can either be obtained after making a prediction without a delay, following the conventional online setting, or will not be available at all. Therefore, these methods are not directly applicable for addressing IVL.

However, the strategies employed by these methods to leverage unlabeled data can offer insightful cues for tackling IVL. Our MIPLOSC specifically draws inspiration from the micro-clustering method proposed by Ud Din et al. [15]. By capturing the geometric characteristics of the dynamic data distribution without storing all historical data, MIPLOSC enables immediate pseudo-labeling and oriented synthetic generation, as detailed in Sections III-B and III-C, respectively.

III. Proposed Method

MIPLOSC improves upon the wait-until-arrival strategy by incorporating two crucial components: the immediate pseudo-labeling mechanism and the oriented synthetic correction mechanism. These components leverage data distribution information, where online clustering technique is utilized to maintain micro-cluster systems for capturing dynamic geometric data features without storing all past data.

Algorithm 1 presents an overview of our MIPLOSC. In the initial stage (Line 1), the classifier $\mathcal{H}$ and two micro-cluster systems, $\mathcal{M}_T$ and $\mathcal{M}_U$, are pre-trained using available training data. Details about these two micro-cluster systems will be shortly presented in Section III-A. Once a new test data $\mathbf{x}_t$ is received, the latest classifier is employed to predict its label $\widehat{y}_t$ (Line 3). Afterwards, the immediate pseudo-labeling mechanism generates the pseudo-label y_t^* for $\mathbf{x}_t$ based on $\mathcal{M}_T$ (Line 4). If the pseudo-label is evaluated to be reliable, as discussed in Section III-B, immediate updates are performed on both the classifier $\mathcal{H}$ and the micro-cluster system $\mathcal{M}_U$ with $(\mathbf{x}_t, y_t^*)$ (Lines 5-7); otherwise, this pseudo-labeled data is excluded from the model update process. When delayed labels are received, they are directly used to update $\mathcal{H}$, $\mathcal{M}_T$, and $\mathcal{M}_U$ (Lines 10-12). During the update process of $\mathcal{M}_U$ with delayed labeled data, the algorithm identifies any label mismatches (Line 13). If a label mismatch is detected, the oriented synthetic correction mechanism, detailed in Section III-C, is activated to generate synthetic data (Line 14). These synthetic instances are subsequently used to update $\mathcal{H}$ and $\mathcal{M}_U$ (Lines 15-17).

A. Adapting Online Micro-clustering Techniques

Before presenting the two core components of our MIPLOSC, it is essential to introduce the integration of online micro-clustering algorithms within the IVL context. MIPLOSC incorporates two micro-cluster systems, namely $\mathcal{M}_T$ and $\mathcal{M}_U$. $\mathcal{M}_T$ is exclusively trained on ground truth data with the primary objective of circumventing label noise incurred in the pseudo-labeling step. Conversely, $\mathcal{M}_U$ is trained on all data used for updating the classifier, allowing it to identify any previous erroneous updates made to the classifier.

Both micro-cluster systems employ the same micro-clustering algorithm to continuously monitor the current data distribution in a class-wise manner. Micro-clusters of each class are maintained and updated in isolation. The choice of micro-clustering algorithms offers flexibility, with viable options such as Din et al.'s [15], CluStream [23], DenStream [24], and DBStream [25]. In this paper, we adopt the micro-clustering algorithm proposed by Din et al. [15].

In addition to the information provided by the original micro-cluster algorithms, our adaptation for the IVL context involves the inclusion of two extra variables: y for class label identification and a weight table WT to track time-decayed class weights. These weights serve as indicators of the recent frequency of data from a specific class entering the micro-cluster's range. The initial weights are set to 0, and upon data entry, the weight corresponding to the class in WT is incremented by 1. Decay of weights occurs at each time step, following the same configuration as the micro-cluster algorithm proposed by Din et al. [15]. Once the weight ratio of a class in a micro-cluster falls below 0.95, the micro-cluster is considered of low quality and subsequently removed.

B. Immediate Pseudo-labeling Mechanism

The immediate pseudo-labeling mechanism promptly assigns and evaluates pseudo-labels for test data immediately after predicting their labels. This mechanism operates under the assumption that concept drift is not excessively drastic and that an overlapping region exists in the data distribution before and after concept drift. Accordingly, the central regions of each class, located far from class boundaries, are considered less vulnerable to the influence of concept drift. To this end, we

Algorithm 1: MIPLOSC

Input: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$: Input features;
Input: $\mathbf{Y} = \{y_1, y_2, \ldots\}$: Target output labels, each obtained after a delay $\delta_t \in (0, \infty), t = 1, 2, \ldots$;
Input: $\mathcal{M}_U$, $\mathcal{M}_T$: Class-wise micro-cluster systems;
Input: $\mathcal{H}$: Classifier;
Input: k: Number of nearest neighbors;
Output: $\widehat{\mathbf{Y}} = \{\widehat{y}_1, \widehat{y}_2, \ldots\}$: Predicted labels.

Initialize $\mathcal{H}, \mathcal{M}_U$, $\mathcal{M}_T$ with initially available data;
for $t \leftarrow 1, 2, \ldots$ **do**
    $\widehat{y}_t \leftarrow \mathcal{H}(\mathbf{x}_t)$;
    $(y_t^*, reliable) \leftarrow$ *ImmediatePseudoLabeling*$(\mathcal{M}_T, \mathbf{x}_t, k)$;
    if $reliable$ **then**
        UpdateClassifier$(\mathcal{H}, \mathbf{x}_t, y_t^*)$;
        UpdateMicroClusterSystem$(\mathcal{M}_U, \mathbf{x}_t, y_t^*)$;
    // Delayed labeled data
    $\mathcal{LD} \leftarrow \{(\mathbf{x}_i, y_i) | \ i + \delta_i = t\}$;
    for $i \leftarrow 1$ **to** $len(\mathcal{LD})$ **do**
        UpdateClassifier$(\mathcal{H}, \mathbf{x}_i, y_i)$;
        UpdateMicroClusterSystem$(\mathcal{M}_T, \mathbf{x}_i, y_i)$;
        UpdateMicroClusterSystem$(\mathcal{M}_U, \mathbf{x}_i, y_i)$;
        if *DetectMismatch*$(\mathcal{M}_U, \mathbf{x}_i, y_i)$ **then**
            // Synthetic data
            $\mathcal{SD} \leftarrow$ *OrientedSyntheticCorrection*$(\mathcal{M}_U, \mathbf{x}_i, y_i, k)$;
            for j $\leftarrow$**to** $len(\mathcal{SD})$ **do**
                UpdateClassifier$(\mathcal{H}, \mathbf{x}_j, y_j)$;
                UpdateMicroClusterSystem$(\mathcal{M}_U, \mathbf{x}_j, y_j)$;

Algorithm 2: ImmediatePseudoLabeling

Input: $\mathcal{M}_T$: Micro-cluster system for ground truth data;
Input: $\mathbf{x}_t$: Feature data at t-th time step;
Input: k: Number of nearest neighbors;
Output: y_t^*: Pseudo-label for $\mathbf{x}_t$;
Output: $reliable$: Whether the assigned pseudo-label is reliable.

1 $\mathcal{N}_T \leftarrow$*kNNBasedOnCenters*$(\mathbf{x}_t, \mathcal{M}_T, k)$;
2 $y_t^* \leftarrow$*MajorityVoting*$(\mathcal{N}_T)$;
3 **if** *AllNeighborsSameClass*$(\mathcal{N}_T)$ **then**
4     $reliable \leftarrow True$;
5 **else**
6     $reliable \leftarrow False$;

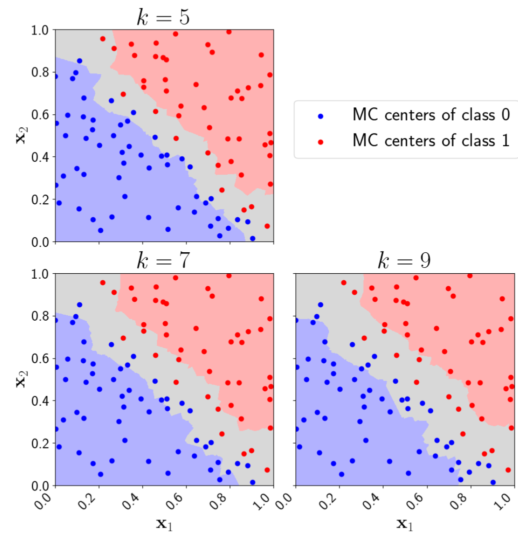


Fig. 1. Illustration of the effects of different choices of k that is used to assess the reliability of pseudo-labeled data. The class boundary is defined by $\mathbf{x}_1 + \mathbf{x}_2 = 1$, where the blue points represent the micro-cluster centers of class 0, and the red points represent the micro-clusters of class 1. The central regions of class 0 and class 1 are visually distinguished with blue and red backgrounds, respectively.

hypothesize that pseudo-labels assigned to data in such central regions of each class are more reliable and robust against concept drift. Such reliable data is anticipated to contribute to benefit model's adaptation to new concepts [26].

To leverage this characteristic, the first step is to determine whether a given data resides within the central region of a class. The determination relies on the summarized data distribution information provided by the micro-cluster system $\mathcal{M}_T$, as it is updated exclusively using the true labeled data, having less noise in the maintained data distribution and thus providing a closer approximation to the true distribution.

Algorithm 2 presents the proposed immediate pseudo-labeling mechanism. Within the MIPLOSC framework, this component first employs the k-NN algorithm on micro-clusters to determine whether a given data point is situated in the central region of a class. Specifically, we identify the k nearest micro-clusters based on their centers (Line 1), then the pseudo-label is assigned through the majority voting scheme (Line 2). If all of the nearest micro-clusters belong to the same class, the data is considered to be situated in the central region of its class, indicating the reliability of the pseudo-label. Only when the pseudo-labeled data can pass the reliability assessment, this data would be used for model update in $\mathcal{H}$ (Line 3-6).

The choice of k enables us to regulate the level to which the central region can be determined. Figure 1 illustrates the effects of difference choices of k. A larger k would lead to a reduced number of noisy pseudo-labeled data which might be wrongly considered as reliable. But this also reduces the number of pseudo-labeled data available for immediate utilization. Conversely, a smaller k would increase the number of pseudo-labeled data available for immediate updates but may also introduce a higher risk of pseudo-label noise. The tuning of this parameter will be discussed in Section IV.

C. Oriented Synthetic Correction Mechanism

Within the MIPLOSC framework, the oriented synthetic correction mechanism is activated upon the detection of label mismatch. When delayed labeled data is received, both classi-

Algorithm 3: OrientedSyntheticCorrection

Input: $\mathcal{M}_U$: Micro-cluster system for data used for updating classifier;
Input: $\mathbf{x}_i$: Feature data that caused label mismatch;
Input: y_i: Class label that caused label mismatch;
Input: k: Number of nearest neighbors;
Output: $\mathcal{SD}$: Synthetic data;

1 $mc \leftarrow$ *GetMisMatchedMicroCluster*$(\mathcal{M}_U, \mathbf{x}_i)$;
2 $(\mathbf{c}, r) \leftarrow$*GetCenterAndRadius*$(mc)$;
3 $\mathcal{N}_U \leftarrow$*kNNBasedOnCenters*$(\mathbf{c}, \mathcal{M}_U, k)$;
4 $\mathcal{C}_y \leftarrow$*GetMicroClustersOfClass*$(\mathcal{N}_U, y_i)$;
5 $n \leftarrow$*DetermineNumSyntheticData*$(\mathcal{N}_U, k)$; // Eq. (1)
6 $\mathcal{SD} \leftarrow$*GenSyntheticData*$(\mathcal{N}_U, \mathbf{c}, y_i, r, n)$; // Eq. (2)

fier $\mathcal{H}$ and micro-cluster system $\mathcal{M}_U$ are updated. During the update process of $\mathcal{M}_U$, a scenario may arise where the delayed labeled data falls within the range of an existing micro-cluster, but the class associated with this micro-cluster differs from the received delayed label. If this leads to the removal of the micro-cluster, it is referred to as a label mismatch in this paper.

The presence of a label mismatch indicates either an erroneous pseudo-label used in a previous classifier update or the occurrence of concept drift. In the former case, mitigating the impact of such mislabeled data during model updates is essential. In the latter case, while we cannot directly address concept drift, proper synthetic data generation could also be valuable in aiding the adaptation of classifier, as long as these synthetic data are consistent with the new concept to some extent. To rectify the classifier from the effects of previous erroneous updates and adapt to concept drift, we propose the oriented synthetic correction mechanism. This mechanism incorporates synthetic data that aligns with the current concept, thereby facilitating the correction of the classifier.

Algorithm 3 presents the oriented synthetic correction mechanism. We use $(\mathbf{x}_i, y_i)$ to denote a delayed labeled data that is employed for updating model $\mathcal{M}_U$ but leading to a label mismatch. Subsequently, we identify the micro-cluster associated with this label mismatch, represented by its center $\mathbf{c}$ and radius r (Lines 1-2). This micro-cluster can be either the one into which $(\mathbf{x}_i, y_i)$ was inserted or the one that was created based on $(\mathbf{x}_i, y_i)$. Then, we identify the k nearest micro-clusters to $\mathbf{c}$ in $\mathcal{M}_U$ (Line 3). Among these k nearest micro-clusters, we select those with the same class as y_i (Line 4), and denote their centers as $\mathcal{C}_y = \left\{\mathbf{c}_y^{(1)}, \mathbf{c}_y^{(2)}, \ldots, \mathbf{c}_y^{(k_c)}\right\}$. These selected micro-clusters are then utilized to determine the location and quantity of synthetic data.

The number of synthetic data is determined as:

$$n \sim \text{Poisson}\left(\frac{k - k_c}{2}\right) + 1. \tag{1}$$

where k_c denotes the count of micro-clusters among the k nearest micro-clusters that share the same class as y_i. The quantity of synthetic data is designed to be inversely correlated with k_c. If $\mathbf{c}$ is entirely surrounded by other micro-clusters that have the same class as y_i, it indicates that the classifier has effectively learned this region. Consequently, correcting this region requires less effort. Conversely, if there are no such surrounding micro-clusters sharing the same class as y_i, it indicates a poor fit of the classifier in this region. Therefore, a large number of synthetic data is required for correction. Each synthetic data is then generated as:

$$\widehat{\mathbf{x}} = \mathbf{c} + \frac{\mathbf{v}}{|\mathbf{v}|} \cdot \beta r \tag{2}$$

where the intermittent vector $\mathbf{v} = \sum_{i=1}^{k_c} \alpha_i \cdot (\mathbf{c}_y^{(i)} - \mathbf{c})$ is generated by a random linear combination of micro-cluster centers within $\mathcal{C}_y$. The coefficients $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_{k_c}\}$ and the scaling factor β are randomly chosen based on a uniform distribution for each synthetic data.

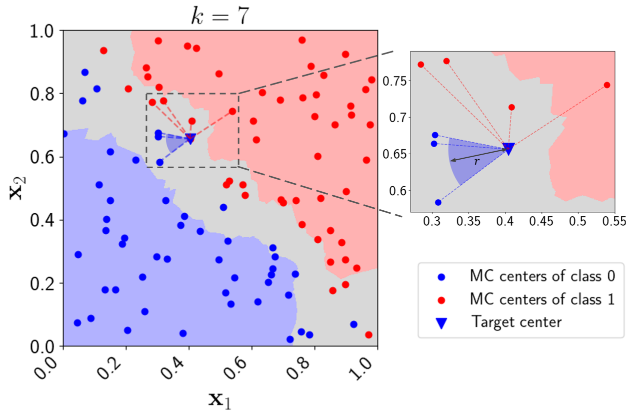


Fig. 2. Illustration of oriented synthetic correction. When a label mismatch is detected, synthetic data are generated within a hyper-sector defined by the centers of micro-clusters sharing the same class label as the delayed true label.

Figure 2 demonstrates the oriented synthetic generation approach, which ensures that all synthetic data are confined within a minimal hyper-sector that encompasses all centers in $\mathcal{C}_y$. The distance from $\mathbf{c}$ is constrained within a maximum radius r. The reliability of synthetic data surpasses that of uniformly generated data within the micro-cluster range. This is attributed to the fact that the side closer to micro-clusters with the same class label is considered more dependable.

IV. Experimental Setup

Unless stated otherwise, the default verification latency δ is set to $1,000$ in our experiments. We then conduct experimental studies based on 10 datasets, including 4 synthetic datasets and 6 real-world datasets, as summarized in Table I. These datasets are widely used in the field of online learning with IVL [27]. The real-world datasets used in our study can be accessed from the USP-DS repository, as mentioned in [27]. Our synthetic datasets are generated using a Python framework called *scikit-multiflow* [32]. Each synthetic streaming data consists of $20,000$ data points and is generated as follows:

- **Hyperplane** is generated by a rotating hyperplane with 10 continuous features ranging from 0 to 1. The decision

TABLE I
AN OVERVIEW OF THE DATASETS INVESTIGATED IN THIS PAPER.

Type	Dataset	# Data	# Attribute	# Class
Synthetic	Hyperplane [28]	20,000	10	2
	RBF	20,000	10	2
	SEA [29]	20,000	3	2
	Sine [30]	20,000	2	2
Real-world	Airlines [27]	20,000	7	2
	Electricity [31]	45,312	8	2
	INS-Grad [27]	24,150	33	6
	INS-Inc-Abt [27]	79,986	33	6
	Rialto [27]	20,000	27	10
	Weather [31]	18,159	8	2

boundary is $\sum_{i=1}^{10} w_i x_i = \theta$, where w_i is a randomly generated weight, and $\theta = \frac{1}{2}\sum_{i=1}^{10} w_i$ is the threshold. Incremental drift is simulated by adjusting the weights of the first 5 features with $w_i = w_i + d\sigma$, where $d = 0.02$ is the adjustment magnitude, $\sigma \in \{-1, 1\}$ controls the direction of the flipping change, and σ flips with a 0.1 probability.

- **RBF** is generated using a radial basis function. Random centroids are created with associated positions, standard deviations, weights, and class labels. Incremental drifts are simulated by continuously moving the centroids. Out of the 20 centroids, 5 are altered at a speed of 0.001.
- **SEA** contains three continuous features within the range of [0, 10]. The class labels are determined by the formula $x_1 + x_2 \leq \theta$. In our setup, abrupt and recurring drifts are simulated by changing the value of θ every 2000 data points in a cycle repeated twice. In each cycle, the sequential values of θ are $7 \to 9 \to 7 \to 9.5 \to 8$.
- **Sine** contains two numerical features ranging from 0 to 1. Data below a specified sine curve are considered positive, and others are labeled negative. An abrupt drift is introduced by modifying the sine function from $y = \sin(x)$ to $y = 0.5 + 0.3\sin(3\pi x)$ at the 10,000-th time step.

We compare MIPLOSC with the wait-until-arrival strategy to investigate the effectiveness of our immediate pseudo-labeling and oriented synthetic correction mechanisms. We also include a topline scenario to provide an upper-bound performance, where no verification latency presents ($\delta = 0$). For this group of competing methods, we utilize Online Bagging [33] of Hoeffding Trees [28] as the classifier, implemented using the *scikit-multiflow* Python package [32] to ensure reproducibility. Moreover, we compare MIPLOSC with state-of-the-art methods that address IVL using the pseudo-labeling with rollback strategy, namely SkipE-RNN [13] and SERMON [9]. Each of these methods employs its own base classifier, ensuring consistency with their original papers.

We perform pre-training using the first 500 data on each dataset. Parameter tuning is conducted with the first 5,000 data through grid search. The parameters for the micro-cluster systems are identical to those specified in [15]. For MIPLOSC, the number of nearest neighbors k is the only parameter to be tuned, selected from the set $\{5, 7, 9\}$. For SkipE-RNN and SERMON, the learning rate is chosen from $\{0.1, 0.01, 0.001\}$.

Accuracy serves as the performance metric for evaluating the overall performance of the methods. For online performance analysis, we employ prequential accuracy with a fading factor of 0.999. This approach, as outlined in Wang et al.'s [34], allows us to assess the real-time performance of the methods. To account for variability, we compute the average results across 10 runs for performance comparisons.

V. EXPERIMENTAL RESULTS

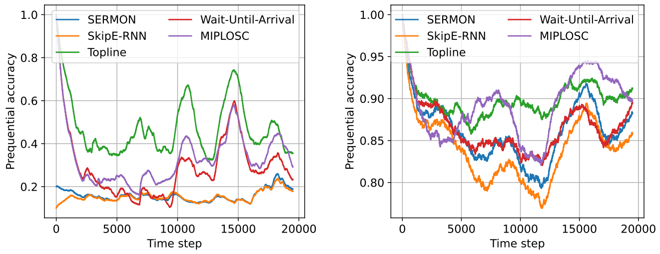
This section provides an empirical validation of our MIPLOSC from various perspectives. The space complexity of competing methods is analyzed and compared in Section V-A. Section V-B compares overall performance of MIPLOSC against other methods for addressing IVL. Section V-C further analyzes its performance at different time periods and compare it with its baseline wait-until-arrival strategy. To validate the effectiveness of the two components of MIPLOSC, an ablation study is conducted in Section V-D. Section V-E investigates the effects of verification latency values on our method.

A. Space Complexity Analysis and Comparison

Let d denote the input feature dimension and s denote the number of target classes. Except for the space cost for storing the classifier, the additional space demand of MIPLOSC arises from storing the micro-cluster systems, leading to a space complexity of $\mathcal{O}(2m(2d + s + 5))$, where m represents the maximum number of the micro-clusters determined by the micro-cluster algorithm. On the other side, the pseudo-labeling with rollback strategy, specifically SkipE-RNN [13] and SERMON [9], requires $\mathcal{O}(\delta h(d + s))$ space to store the previous classifier states and $\mathcal{O}(N(d + s))$ space to maintain historical labeled data for label matching. Consequently, the overall space complexity amounts to $\mathcal{O}(\delta h(d+s)+N(d+s))$, where h represents the number of hidden nodes in the network, and N is the number of historical labeled data.

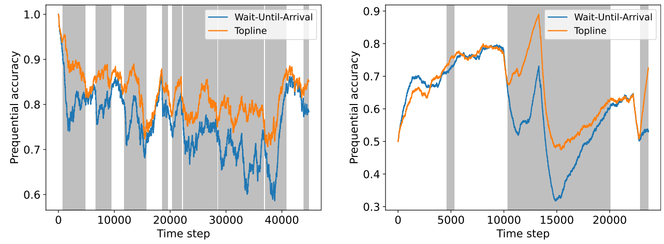
In the context of online learning with IVL, N is typically much larger than m. As a result, the space required solely for storing historical labeled data already exceeds that consumed by micro-cluster systems in MIPLOSC, not to mention the additional space needed for storing previous model states. Furthermore, as the label delay δ increases, the storage space required for model states becomes an increasingly crucial concern. Therefore, our proposed MIPLOSC generally requires less space compared to approaches that utilize the pseudo-labeling with rollback strategy.

Table II presents the empirical space costs for all methods. To provide a more comprehensive empirical analysis, we also include the corresponding computational costs in this table. We can see that the wait-until-arrival and idealistic topline approaches exhibit the best empirical space, as they require no additional memory. Besides that, MIPLOSC has a substantial space advantage compared to SkipE-RNN and SERMON. With an average rank of 3.0, the space cost of MIPLOSC is less than one-third of that of SkipE-RNN and SERMON, and even as low as one-fifth in several datasets. This advantage is



(a) Rialto (b) RBF

Fig. 3. Continuous performance comparison over time on representative datasets in terms of prequential accuracy. MIPLOSC outperforms competing methods for the majority of the time steps, except for the idealized topline scenario where no label delay is present.



(a) Electricity (b) INS-Grad

Fig. 4. Illustration of adaptation and stable periods on two representative datasets. Periods characterized by significant differences in prequential performance between the wait-until-arrival strategy and the topline scenario are recognized as adaptation periods, highlighted with a gray background. The remaining periods are considered as stable periods.

achieved by summarizing data distribution through the micro-cluster systems, eliminating the need to store all historical data and numerous previous model status, as required by SkipE-RNN and SERMON. In terms of computational cost, MIPLOSC exhibits relatively higher expenses, with an average rank of 4.0, outperforming SERMON and being inferior to SkipE-RNN. This can be attributed to the time-consuming k-NN process involved in pseudo-labeling and synthetic data generation at each time step.

Overall, MIPLOSC has the merits of having its economic efficiency in terms of space cost, making it a highly viable alternative method addressing IVL.

B. Overall Performance Comparison

Table III(a) presents the overall performance in terms of accuracy. We can see that MIPLOSC achieves the best (lowest) average rank of 2.7 across the datasets, except for the topline scenario. The topline scenario represents an upper-bound performance that cannot be practically implemented due to the assumption of zero verification latency. To validate the performance enhancement of MIPLOSC over its baseline wait-until-arrival strategy, we further conduct Wilcoxon signed-rank test between the two approaches across datasets. The null hypothesis states that MIPLOSC is not superior to the wait-until-arrival strategy. The statistical test yields a p-value of 0.0029, which is smaller than the significance level of 0.05. Thus, the null hypothesis is rejected, indicating that the proposed MIPLOSC significantly improves the predictive performance compared to its baseline method.

Figure 3 shows online performance on two representative datasets throughout test steps, demonstrating MIPLOSC's superiority over its competitors at most test steps. This observation aligns with our foundational assumption of the immediate pseudo-labeling mechanism, as discussed in Section III-B.

C. Performance Comparison at Different Time Periods

The learning process in online scenarios can be partitioned into two main periods: the adaptation periods, also known as plasticity periods, and the stable periods [35], [36]. Analyzing the online learning performance during these distinct periods is crucial for understanding the effectiveness of our method. The adaptation period of the model typically begins when a new concept occurs, often accompanied by a performance decline and recovery. If there is no subsequent concept drift, the adaptation period transitions into a stable period where the model becomes well-adapted to the concept, leading to online performance close to that of the topline scenario.

To empirically distinguish the adaptation and stable periods, we utilize the online performance difference between the topline scenario and the wait-until-arrival strategy. If the performance difference exceeds a specified threshold, we identify the current time step as belonging to the adaptation periods. This approach ensures that we naturally account for the impact of delays on the adaptation periods. It is noteworthy that the depicted partitioning approach is not feasible to implement in practice, as the upper-bound performance of the topline scenario is unknown. However, it is valuable for conducting further analysis and gaining insights into how our method impacts different periods within the online learning process.

Using a fixed partition threshold for all datasets is not suitable due to the variations in online performance ranges across different datasets. Instead, we employ a more flexible approach by employing the quantile as the threshold, so that we can adapt to the characteristics of each dataset individually. This involves applying different quantiles to different datasets based on the length of the time interval where the performance of the topline scenario and the wait-until-arrival strategy closely align. For datasets with longer matching time intervals, lower thresholds are set, while datasets with shorter intervals would have higher thresholds. Specifically, for Hyperplane, RBF, Electricity, and Rialto, we choose the quartile (the 25%-quantile) as the threshold; for the remaining datasets, we choose the median (the 50%-quantile) as the threshold. To illustrate the partition between the adaptation and stable periods, we present two representative results in Figure 4.

In this analysis, our primary focus is to compare MIPLOSC with the wait-until-arrival strategy, investigating the effectiveness of our proposed approaches during adaptation periods and stable periods respectively. Table IV presents the average accuracy achieved by these two approaches in these distinct periods across all datasets. To assess the effect of MIPLOSC

TABLE II

EMPIRICAL COMPARISON OF SPACE AND COMPUTATIONAL COSTS. THE AVERAGE RANK (AVGRANK) IS THE MEAN RANK OF EACH METHOD ACROSS DATASETS. TABLE II(A) REPORTS THE SPACE COST IN MEBIBYTE (MIB) THROUGHOUT THE ENTIRE RUNNING PROCESS. TABLE II(B) REPORTS THE AVERAGE COMPUTATIONAL COST IN MILLISECONDS (MS) AT EACH TIME STEP. THE VALUE IN BRACKETS IS THE RANK OF THE RESULTS IN EACH ROW.

(a) Space Cost (MiB)

	MIPLOSC	SkipE-RNN	SERMON	Topline	Wait-Until-Arrival
Hyperplane	24.0 (3)	98.3 (4)	101.5 (5)	9.3 (2)	9.2 (1)
RBF	26.9 (3)	97.3 (4)	99.7 (5)	8.9 (2)	8.6 (1)
SEA	23.2 (3)	90.0 (4)	94.5 (5)	7.0 (1)	7.0 (1)
Sine	17.7 (3)	89.8 (4)	93.6 (5)	6.9 (2)	6.5 (1)
Airlines	12.5 (3)	61.9 (4)	65.3 (5)	3.5 (1)	3.5 (1)
Electricity	36.7 (3)	153.3 (5)	153.2 (4)	15.7 (2)	15.5 (1)
INS-Grad	48.0 (3)	109.5 (5)	108.0 (4)	19.0 (2)	18.6 (1)
INS-Inc-Abt	65.5 (3)	191.4 (5)	188.7 (4)	34.9 (2)	32.6 (1)
Rialto	41.5 (3)	110.5 (4)	112.5 (5)	17.0 (2)	16.4 (1)
Weather	21.8 (3)	89.3 (4)	93.4 (5)	7.0 (2)	6.8 (1)
AvgRanks	3.0	4.3	4.7	1.8	1.0

(b) Computational Cost (ms)

	MIPLOSC	SkipE-RNN	SERMON	Topline	Wait-Until-Arrival
Hyperplane	13.381 (4)	5.853 (3)	38.193 (5)	1.440 (2)	1.412 (1)
RBF	13.293 (4)	5.586 (3)	20.390 (5)	1.426 (2)	1.395 (1)
SEA	11.569 (4)	5.308 (3)	15.898 (5)	0.855 (2)	0.838 (1)
Sine	10.702 (4)	5.225 (3)	10.978 (5)	0.791 (2)	0.774 (1)
Airlines	12.629 (4)	7.008 (3)	42.229 (5)	1.177 (2)	1.138 (1)
Electricity	12.276 (4)	6.918 (3)	28.894 (5)	1.199 (2)	1.159 (1)
INS-Grad	22.171 (4)	9.535 (3)	64.004 (5)	6.299 (2)	6.200 (1)
INS-Inc-Abt	20.103 (4)	9.655 (3)	71.373 (5)	6.444 (2)	6.334 (1)
Rialto	22.774 (4)	9.574 (3)	106.247 (5)	8.322 (2)	8.052 (1)
Weather	13.009 (4)	6.930 (3)	36.244 (5)	1.252 (2)	1.224 (1)
AvgRanks	4.0	3.0	5.0	2.0	1.0

TABLE III

OVERALL PERFORMANCE IN TERMS OF ACCURACY (%). THE AVERAGE RANK (AVGRANK) IS THE MEAN RANK OF EACH METHOD ACROSS DATASETS. TABLE III(B) REPORTS THE ABLATION RESULTS, WHICH SPECIFICALLY COMPARE TWO VARIANTS OF MIPLOSC AGAINST THE BASELINE WAIT-UNTIL-ARRIVAL STRATEGY, WHERE THE WAIT-UNTIL-ARRIVAL STRATEGY SERVES AS THE CONTROL METHOD. IN TABLE III(A), THE VALUE IN BRACKETS REPRESENTS THE RANK OF THE RESULTS WITHIN EACH ROW. IN TABLE III(B), THE BOLD TEXT HIGHLIGHTS THE RESULTS OF VARIANTS WHOSE PERFORMANCE SURPASSES THAT OF THE WAIT-UNTIL-ARRIVAL STRATEGY ON EACH DATASET.

	(a) Performance Comparison					(b) Ablation Analysis	
	MIPLOSC	SkipE-RNN	SERMON	Topline	Wait-Until-Arrival	W+IPL	W+OSC
Hyperplane	74.29±0.22 (2)	73.68±1.56 (5)	74.10±0.91 (3)	79.20±0.24 (1)	73.74±0.27 (4)	**74.30±0.11**	**74.00±0.18**
RBF	88.64±0.20 (2)	83.35±0.70 (5)	85.51±0.56 (4)	89.41±0.33 (1)	85.99±0.28 (3)	**88.59±0.27**	**86.29±0.12**
SEA	91.51±0.12 (5)	92.34±0.45 (2)	92.24±0.37 (3)	93.42±0.08 (1)	91.73±0.05 (4)	91.64±0.12	91.26±0.10
Sine	91.14±0.30 (2)	88.70±0.24 (5)	88.77±0.28 (4)	91.63±0.18 (1)	90.30±0.18 (3)	**90.47±0.36**	**90.87±0.24**
Airlines	62.94±0.67 (2)	58.19±0.32 (5)	59.48±0.45 (4)	63.86±0.28 (1)	61.48±0.41 (3)	**62.53±0.45**	**61.63±0.87**
Electricity	75.89±0.25 (2)	75.21±0.68 (4)	74.38±0.40 (5)	81.99±0.30 (1)	75.56±0.28 (3)	**76.11±0.33**	75.07±0.47
INS-Grad	63.17±0.55 (4)	63.86±0.49 (3)	63.86±0.37 (2)	67.07±0.78 (1)	61.38±0.97 (5)	**62.56±0.39**	**61.88±0.47**
INS-Inc-Abt	60.64±0.27 (4)	62.31±0.40 (2)	62.25±0.17 (3)	66.41±0.57 (1)	60.56±0.59 (5)	60.36±0.50	**61.36±0.42**
Rialto	29.67±0.83 (2)	15.73±0.42 (5)	15.82±0.42 (4)	44.24±0.89 (1)	25.14±1.16 (3)	**26.71±0.48**	**31.74±1.05**
Weather	73.78±0.32 (2)	72.18±1.59 (5)	72.79±0.86 (4)	74.97±0.36 (1)	73.32±0.34 (3)	**73.81±0.27**	**74.00±0.17**
AvgRanks	2.7	4.1	3.6	1.0	3.6	**1.6 vs 2.6**	**1.8 vs 2.6**

in these periods, we conduct Wilcoxon signed-rank tests with a significance level of 0.05. The null hypothesis supposes that MIPLOSC is not superior to the wait-until-arrival strategy. The resulting p-values are $0.0098 < 0.05$ for the adaptation periods and $0.5 > 0.05$ for the stable periods. These results indicate that MIPLOSC significantly improves performance during the adaptation periods, while not causing a significant degradation during stable periods. Consequently, the proposed MIPLOSC contributes to an overall accuracy improvement by particularly enhancing adaptation process.

D. Ablation Analysis of Our MIPLOSC

This section aims to investigate the effectiveness of MIPLOSC's two core components: the immediate pseudo-labeling mechanism (Section III-B) and the oriented synthetic correction mechanism (Section III-C). To assess the effectiveness of each component, we integrate them individually into the wait-until-arrival strategy, resulting in two MIPLOSC degraded variants: W+IPL (wait-until-arrival strategy with only the immediate pseudo-labeling mechanism) and W+OSC (wait-until-arrival strategy with only the oriented synthetic correction mechanism that focuses solely on classifier correction

TABLE IV

PERFORMANCE IN ADAPTATION AND STABLE PERIODS IN TERMS OF AVERAGE ACCURACY (%). THE p-VALUES OF WILCOXON SIGNED-RANK TESTS COMPARING OUR MIPLOSC AGAINST THE WAIT-UNTIL-ARRIVAL STRATEGY ARE PRESENTED IN THE LAST ROW, WITH THE SUPERIOR PERFORMANCE HIGHLIGHTED IN BOLD.

	Adaptation periods		Stable periods	
	MIPLOSC	Wait-Until-Arrival	MIPLOSC	Wait-Until-Arrival
Hyperplane	**72.48**	71.75	**79.11**	79.05
RBF	**89.69**	85.64	85.55	**87.00**
SEA	90.77	**91.34**	**92.24**	92.12
Sine	**86.31**	85.13	**95.01**	94.43
Airlines	**61.59**	59.30	**63.77**	62.83
Electricity	**74.89**	73.67	78.58	**80.70**
INS-Grad	**58.13**	52.18	67.71	**69.66**
INS-Inc-Abt	58.06	**59.09**	**63.16**	61.99
Rialto	**29.98**	24.28	**28.73**	27.69
Weather	**72.69**	71.99	**74.77**	74.55
p-value	**$0.0098 < 0.05$**		$0.5 > 0.05$	

from concept drift). By comparing the performance of these variants with the original wait-until-arrival strategy, we can demonstrate the effectiveness of each component.

Using the wait-until-arrival strategy as the control method,

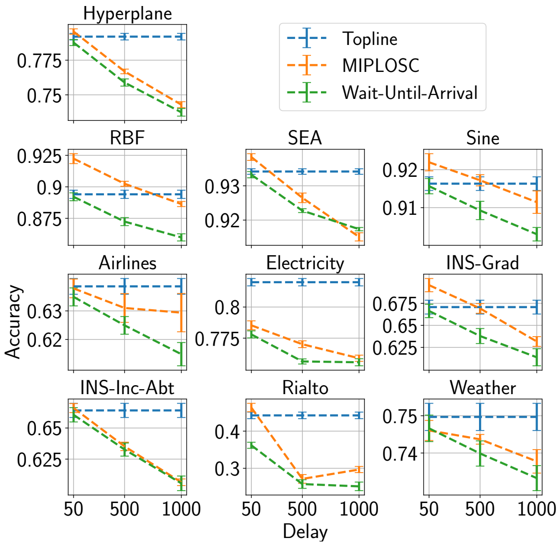


Fig. 5. Accuracy trends in relation to latency values. The error bars indicate standard deviation. MIPLOSC consistently outperforms the wait-until-arrival strategy across multiple latency levels for almost all datasets.

we conduct a comparative analysis of W+IPL and W+OSC. Table III(b) reports the average accuracy of these two variants. We can see that both degraded variants outperform the baseline wait-until-arrival strategy. The average ranks against the control method are 1.6 vs. 2.6 for W+IPL and 1.8 vs. 2.6 for W+OSC. Wilcoxon signed-rank tests at the significant level of 0.05 are further conducted to compare the performance of each variant against the control method. The p-values are 0.007 for W+IPL and 0.032 for W+OSC, respectively. The results provide strong evidence of the significant improvement brought about by both components, affirming their effectiveness in enhancing the overall performance of our MIPLOSC.

E. Effects of Latency Values on Our MIPLOSC

Ensuring robust performance across different levels of latency is important for an effective approach for addressing IVL. This section investigates the efficacy of MIPLOSC across varying delays (δ) and assess how performance evolves with increasing label delay. We set δ to three different values: 50, 500, and 1000, and then compare MIPLOSC against the topline scenario and the wait-until-arrival strategy.

Figure 5 illustrates accuracy trends across all datasets with respect to different δ values. We can see that MIPLOSC consistently outperforms the wait-until-arrival strategy at all δ values in almost all datasets, highlighting its effectiveness in handling diverse levels of verification latency. Notably, in some datasets, the performance of MIPLOSC at $\delta = 50$ can even surpass that of the topline scenario. This observation can be attributed to the facilitation provided by MIPLOSC during the adaptation process. For example, in the initial stage, with a limited number of data points (500), the classifier might struggle to adapt to the concept, thereby making the support of MIPLOSC beneficial. As the δ value increases, the impact of delay becomes more pronounced, resulting in a widening performance gap between MIPLOSC and the topline scenario.

In summary, our MIPLOSC demonstrate robust performance under different degrees of verification delay, ensuring its effectiveness across various latency conditions.

VI. Conclusion

This paper proposed a novel approach, called MIPLOSC, to address the IVL issue in online learning. MIPLOSC integrates two core components: an immediate pseudo-labeling mechanism and an oriented synthetic correction mechanism, both leveraging micro-cluster systems. Compared to existing IVL methods, MIPLOSC effectively utilizes temporarily unlabeled data for immediate classifier updates and reduce the loss of valuable information caused by previous erroneous updates. To evaluate the performance of MIPLOSC, comprehensive experimental evaluations were conducted using synthetic and real-world datasets. The results demonstrated the superiority of MIPLOSC over state-of-the-art IVL methods. The experimental findings highlighted a significant improvement over the baseline wait-until-arrival strategy upon which MIPLOSC was built. The ablation study confirmed the importance and contribution of each proposed component to the overall performance of MIPLOSC. Experimental results also exhibited the robustness of MIPLOSC to different levels of latency. Further study that analyzed the performance of MIPLOSC at different time periods found that MIPLOSC primarily enhances the adaptation process in online learning under IVL.

Moreover, MIPLOSC exhibited significant advantages in terms of space cost. The utilization of micro-cluster systems not only provides data distribution information for both components but also reduces the space cost by eliminating the need to store all historical data. Theoretical and empirical analyses emphasize the spatial-saving benefits of MIPLOSC compared to state-of-the-art methods.

While MIPLOSC offers notable benefits, it is essential to acknowledge its limitations and considerations. Firstly, as discussed in Section III-B, the immediate pseudo-labeling mechanism of MIPLOSC assumes concept drift is not excessively drastic. Consequently, in scenarios with significant concept drift, MIPLOSC's performance may be constrained. Secondly, given both core components of MIPLOSC rely on micro-cluster systems, its performance is closely linked to the parameter tuning of the micro-clustering method. Inappropriate parameter settings in the micro-clustering method can impact the overall performance of MIPLOSC. Therefore, it is crucial to prioritize micro-clustering methods that are less sensitive to parameter variations.

Future research efforts could be directed towards addressing various challenges in the context of IVL, including tackling drastic concept drift, adopting parameter-robust online clustering methods, effectively handling of class imbalance, and accounting for the presence of label noise.

ACKNOWLEDGMENT

This work was supported the by State Key Laboratory of Robotics (Grant No. 2023-O11), National Natural Science Foundation of China (Grant No. 62002148), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. ZDSYS201703031748284), and Research Institute of Trustworthy Autonomous Systems (RITAS).

REFERENCES

[1] Y. Song, J. Lu, H. Lu, and G. Zhang, "Fuzzy clustering-based adaptive regression for drifting data streams," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 3, pp. 544–557, 2020.

[2] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining Knowledge Discovery*, vol. 30, no. 4, p. 964–994, 2016.

[3] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019.

[4] J. a. Gama, I. Žliobaitundefined, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, 2014.

[5] V. M. Souza, D. F. Silva, G. E. Batista, and J. Gama, "Classification of evolving data streams with infinitely delayed labels," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 2015, pp. 214–219.

[6] V. Souza, T. Pinho, and G. Batista, "Evaluating stream classifiers with delayed labels information," in *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, 2018, pp. 408–413.

[7] H. M. Gomes, M. Grzenda, R. Mello, J. Read, M. H. Le Nguyen, and A. Bifet, "A survey on semi-supervised learning for delayed partially labelled data streams," *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–42, 2022.

[8] C. Madhusudana, H. Kumar, and S. Narendranath, "Condition monitoring of face milling tool using k-star algorithm and histogram features of vibration signal," *Engineering Science and Technology, an International Journal*, vol. 19, no. 3, pp. 1543–1551, 2016.

[9] M. Das, M. Pratama, and T. Tjahjowidodo, "A self-evolving mutually-operative recurrent network-based model for online tool condition monitoring in delay scenario," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, p. 2775–2783.

[10] M. Masud, J. Gao, L. Khan, J. Han, and B. M. Thuraisingham, "Classification and novel class detection in concept-drifting data streams under time constraints," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 6, pp. 859–874, 2011.

[11] A. L. Cristiani, T. P. da Silva, and H. de Arruda, Camargo, "A fuzzy approach for classification and novelty detection in data streams under intermediate latency," in *Intelligent Systems: 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20–23, 2020, Proceedings, Part II*, 2020, p. 171–186.

[12] A. L. Cristiani and H. de Arruda Camargo, "A fuzzy multi-class novelty detector for data streams under intermediate latency," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021, pp. 1–6.

[13] M. Das, M. Pratama, J. Zhang, and Y. Ong, "A skip-connected evolving recurrent neural network for data stream classification under label latency scenario," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 3717–3724, 2020.

[14] J. Shao, Y. Tan, L. Gao, Q. Yang, C. Plant, and I. Assent, "Synchronization-based clustering on evolving data stream," *Information Sciences*, vol. 501, pp. 573–587, 2019.

[15] S. Ud Din, J. Shao, J. Kumar, W. Ali, J. Liu, and Y. Ye, "Online reliable semi-supervised learning on evolving data streams," *Information Sciences*, vol. 525, pp. 153–171, 2020.

[16] S. Mohamad, M. Sayed-Mouchaweh, and A. Bouchachia, "Active learning for classifying data streams with unknown number of classes," *Neural Networks*, vol. 98, pp. 1–15, 2018.

[17] E. Lughofer, E. Weigl, W. Heidl, C. Eitzinger, and T. Radauer, "Recognizing input space and target concept drifts in data streams with scarcely labeled and unlabelled instances," *Information Sciences*, vol. 355-356, pp. 127–151, 2016.

[18] P. Li, X. Wu, and X. Hu, "Mining recurring concept drifts with limited labeled streaming data," *ACM Transaction on Intelligent Systems and Technology*, vol. 3, no. 2, pp. 1–32, 2012.

[19] X. Wu, P. Li, and X. Hu, "Learning from concept drifting data streams with unlabeled data," *Neurocomputing*, vol. 92, pp. 145–155, 2012.

[20] Y. Jia, S. Yan, and C. Zhang, "Semi-supervised classification on evolutionary data," in *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 2009, p. 1083–1088.

[21] L. Zhu, S. Pang, A. Sarrafzadeh, T. Ban, and D. Inoue, "Incremental and decremental max-flow for online semi-supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2115–2127, 2016.

[22] T. Wagner, S. Guha, S. Kasiviswanathan, and N. Mishra, "Semi-supervised learning on data streams via temporal label propagation," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, 2018, pp. 5095–5104.

[23] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proceedings of the 29th International Conference on Very Large Data Bases*, 2003, p. 81–92.

[24] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proceedings of the 2006 SIAM international conference on data mining*, 2006, pp. 328–339.

[25] M. Hahsler and M. Bolaños, "Clustering data streams based on shared density between micro-clusters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1449–1461, 2016.

[26] A. Tsymbal, M. Pechenizkiy, P. Cunningham, and S. Puuronen, "Dynamic integration of classifiers for handling concept drift," *Information Fusion*, vol. 9, no. 1, pp. 56–68, 2008.

[27] V. M. A. Souza, D. M. dos 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*, vol. 34, no. 6, pp. 1805–1858, 2020.

[28] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 97–106.

[29] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 377–382.

[30] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence*, A. L. C. Bazzan and S. Labidi, Eds., 2004, pp. 286–295.

[31] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2283–2301, 2013.

[32] J. Montiel, J. Read, A. Bifet, and T. Abdessalem, "Scikit-multiflow: A multi-output streaming framework," *Journal of Machine Learning Research*, vol. 19, no. 72, pp. 1–5, 2018. [Online]. Available: https://scikit-multiflow.github.io/

[33] N. Oza, "Online bagging and boosting," in *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, 2005, pp. 2340–2345.

[34] S. Wang, L. L. Minku, and X. Yao, "A systematic study of online class imbalance learning with concept drift," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4802–4821, 2018.

[35] S. Grossberg, "Nonlinear neural networks: Principles, mechanisms, and architectures," *Neural Networks*, vol. 1, no. 1, pp. 17–61, 1988.

[36] J. L. Lobo, I. Laña, J. Del Ser, M. N. Bilbao, and N. Kasabov, "Evolving spiking neural networks for online learning over drifting data streams," *Neural Networks*, vol. 108, pp. 1–19, 2018.