UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Toward Utilitarian Online Learning with Class Evolution

by Zhi Cao

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of
Professor Chin-Teng Lin,
Professor Xin Yao

University of Technology Sydney
Faculty of Engineering and Information Technology

Jan 2025

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Zhi Cao, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Computer Science, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: Jan 2025

ACKNOWLEDGMENTS

I would like to sincerely thank my two principle supervisors: Professor Chin-Teng Lin and Professor Xin Yao. Thank you so much for granting me the opportunity and financial support to pursue this research. Your guidance and encouragement have been instrumental to this journey, inspiring me to grow both academically and personally. I am deeply grateful for your dedication, insight, and patience on me. Thank you for believing in my potential and for pushing me to grow beyond what I thought possible.

I would like to thank all the members in CIBCI (Computational Intelligence and Brain Computer Interface Lab) and the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation. In particular, I am grateful to Dr. Yu-Cheng Chang for discussing research problems with me and for providing immensely valuable assistance and collaboration. My sincere thanks also extend to Prof. Ye Shi, Prof. Li-Yan Song, and Prof. Jia-Lin Liu, who dedicated significant time and offered me practical suggestions. I would also like to thank Dr. Shu-Yi Zhang for the countless discussions, brainstorming sessions, and selfless support through many challenges along the way. I am thankful to my ex-girlfriend, Dr. Ya-Zhi Cui, who have experienced both happiness and difficulties with me.

Finally, I am forever thankful to my family and friends for their love, patience and faith in me. Your encouragement has been my foundation, providing the strength I needed through every obstacle.

To each of you, my heartfelt gratitude. This work would not have been possible without you, and I am forever thankful.

Dedication

To my families

Qi-Yue Zhou,

Rui-Ming Zhou,

Jing-Yi Cao,

Hui Fang,

and

Chun-Xi Cao

with love and gratitude.

Abstract

In real-world data stream mining, the composition of classes undergoes unpredictable changes, giving rise to the challenge of class evolution, which encompasses class emergence, class disappearance, and class reoccurrence. While most existing approaches necessitate the storage of past data to adapt their model to class evolution, several studies have focused on developing online learning algorithms that process instances in the data stream one-by-one and do not require the storage of historical data for model adaptation. However, existing online learning algorithms on handling data streams with class evolution are developed under potentially restrictive assumptions regarding 1) the prior distribution of classes, 2) the availability of labels for instances in the data stream for model adaptation, or 3) the assumption that the feature space remains fixed rather than varying. The non-stationary and unpredictable nature of the real-world data stream leads to unsatisfactory performance of these algorithms. Therefore, this research addresses these limitations by proposing online learning algorithms designed to manage data streams with class evolution for utilitarian, real-world applications.

First, the Ensemble of Ensemble One-versus-all (OVA) Framework (EEOF) is proposed to handle data streams with class evolution across a broader range of prior distributions on classes. Previous studies implicitly assume that the number of instances in any single class is consistently less than that of other classes. This assumption becomes invalid when a class emerges with a dominant amount, harming the performance of existing methods on multi-class classification tasks. EEOF is constructed in an ensemble

of ensemble approach to enhance diversity within each OVA classifier. In particular, it incorporates a novel adaptive model adaptation method that does not rely on any underlying assumptions about the data stream, with the aim of balancing error feedback between the emerging classes and the other classes. Additionally, a confidence-triggered fallback mode is designed to prevent performance degradation resulting from incorrect decisions regarding class disappearance. Experimental studies are conducted on a broad range of synthetic data streams, and the real-world data streams, demonstrating that EEOF achieves higher accuracy in diverse class evolution scenarios, particularly in the unsolved scenarios in which classes emerge with a dominant amount.

Moreover, this thesis also proposes a Semi-supervised Online Ensemble Learning Framework (SOELF) to handle data streams with class evolution under incomplete supervision. A significant challenge in utilizing existing online learning algorithms for real-world applications is their reliance on fully labeled data streams to adapt model to novel classes. SOELF incorporates a data generation module based on an ensemble online clustering model to generate diverse instances for model adaptation. The pseudo-labels for these generated instances are determined using the decay G-mean metric. This approach enables the model to leverage unlabeled instances and effectively adapt to novel classes with only a limited number of labeled instances. An inverse-prior-ratio model adaptation strategy is designed to introduce more instances of existing classes for model adaptation when novel classes emerge, balancing the exploitation between classes to tackling the issue of classifiers in existing online learning framework excessively focusing on labeled instances of novel classes. Empirical analyses and ablation studies demonstrate the superiority of SOELF and the necessity of its components in handling synthetic and real-world data streams with class evolution under incomplete supervision.

Finally, an Ensemble of Passive-Aggressive Model (EPAM) is developed to address the problem of online learning in open data space, where both feature and label spaces

evolve and model adapts to individual instances without the need for storing past data. Previous work in online learning assumes a fixed space, either in terms of features or labels, which is overly idealistic when applied to real-world applications. We delve into this problem by first constructing several baseline methods grounded in state-of-the-art approaches from both online learning with class evolution and online learning with feature evolution. We then analyze their limitations in handling real-world data streams with open data space and proceed to introduce EPAM. In EPAM, the feature contributed bias classifier (FCBC) is integrated as the OVA classifier with an adaptive bias term capable of adapting to diverse feature spaces created by different combinations of features. Furthermore, a model adaptation strategy is designed to balance the error feedback among the negative class of each FCBC, thereby enhancing the multi-class classification performance. Simulation results on diverse synthetic and real-world data streams, along with a comprehensive ablation study, demonstrate the superiority of EPAM in managing data streams with open data space.

In summary, the main contributions of this research are as follows: 1) a novel ensemble of ensemble one-versus-all framework is proposed to address class evolution in the online learning procedure. This framework integrates a novel model adaptation strategy and a confidence-triggered fallback mode to tackle diverse class evolution scenarios, particularly when classes emerge with a dominant amount. 2) a semi-supervised online ensemble learning framework is proposed to delve into online learning with class evolution under incomplete supervision. A novel data generation module and an inverse-prior-ratio model adaptation strategy are developed to exploit the unlabeled instances for more efficient model adaptation. 3) Finally, an innovative ensemble of passive-aggressive model is proposed to resolve the evolution on both label and feature spaces. A novel feature contributed bias classifier and a novel model adaptation strategy are developed to enhance classification accuracy in data streams with open data space.

List of Publications

Related to the Thesis :

1. **Z. Cao**, S. Zhang and C. T. Lin, "Online Ensemble of Ensemble OVA Framework for Class Evolution with Dominant Emerging Classes," *2023 IEEE International Conference on Data Mining (ICDM)*, Shanghai, China, 2023, pp. 968-973.

2. **Z. Cao**, S. Zhang, C. T. Lin and X. Yao, "Online Ensemble Learning for Class Evolution under Incomplete Supervision", submitted to *2025 Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.

3. **Z. Cao**, P. Qin, C. T. Lin and X. Yao, "Online Learning in Open Data Space", submitted to *2025 IEEE International Conference on Data Engineering (ICDE)*.

Others :

4. **Z. Cao**, Y. Shi, Y. C. Chang, X. Yao and C. T. Lin, "Twin Fuzzy Networks With Interpolation Consistency Regularization for Weakly Supervised Anomaly Detection," in *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 9, pp. 5086-5097, 2024.

5. X. Wu, Z. Hu, K. Pei, L. Song, **Z. Cao** and S. Zhang, "Methods for deep learning model failure detection and model adaption: A survey," *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, Wuhan, China, 2021, pp. 218-223.

TABLE OF CONTENTS

List of Figures

Figure **Page**

List of Tables

Table **Page**

CHAPTER 1

INTRODUCTION

1.1 Online Learning with Class Evolution

Over the past few decades, the rapid advancement of technologies such as communication media [27], autonomous systems [14], smartphones [83], and smart city infrastructure [61] has led to the continuous generation of vast data streams, characterized by large volumes and high velocities [4, 62]. Properly analysis of these streaming data can yield essential insights for building intelligent entities [70]. Classification is one of the most widely studied research topics in data stream mining [20, 30]. The construction of a classifier in a classification task aims to approximate the mapping function from the input features of an instance to discrete classes, i.e. $\mathscr{F} : \mathscr{X} \rightarrow \mathscr{Y}$, where $\mathscr{F}$ is the mapping function, $\mathscr{X}$ is the feature space and $\mathscr{Y}$ is the label space. A well-known pre-requisite for most machine learning algorithms is the independently and identically distributed (i.i.d.) assumption, referring that the instances of the train and test data shall follow the same distribution and be independent to each other [71].

However, the identically-distributed assumption tends to be invalid in real-world

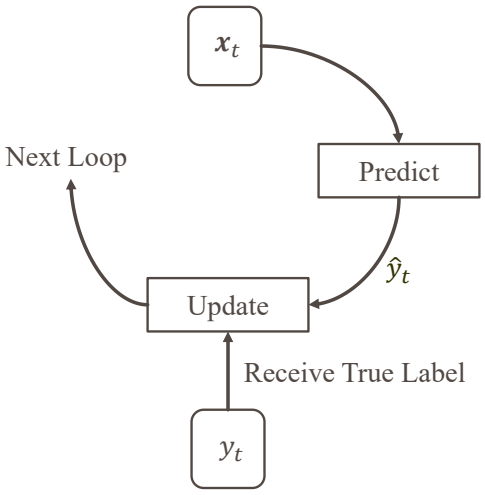


Figure 1.1: Illustration of the online learning process in data stream.

data streams. One of the most fundamental principles of real-world data streams is their non-stationary nature, which implies that the data distribution in both the feature space and the label space, even the spaces themselves, evolve over time. The term *class evolution* describes the phenomenon in which the compositions of classes in a data stream for a real-world classification task is not a fixed set, but instead undergoes unpredictable changes [75]. This means that the label space $\mathcal{Y}_t$, evolves with time t. As the label space evolves, novel classes may emerge, existing classes may disappear, and disappeared classes may reoccur again. An ideal classifier is supposed to adapt to class evolution in real-world data streams. In other words, such a classifier approximates the mapping function $\mathcal{F}: \mathcal{X} \rightarrow \mathcal{Y}_t$. Online learning, a distinct mode of learning characterized by processing instances one-by-one and requiring no data storage of past data (as illustrated in Figure 1.1), has received significant attention in recent years [43, 47]. This single-pass data processing strategy is well-suited to handle the challenges posed by large volumes and high velocities in real-world data streams [81], particularly useful for applications such as communication media analysis [75], spam filtering [53], and sensor networks [79]. Furthermore, the in-time model adaptation in online learning effectively tracks the changes in data streams, while also presents the challenge of adapting to data streams

based on a single available data instance.

Research on data stream classification with class evolution has been ongoing for years [20, 48], primarily focusing on the emergence of novel classes. However, most existing studies develop approaches that require massive data storage of past instances to adapt the model to class evolution. These algorithms either process incoming instances from the data stream in a chunk-by-chunk manner [19, 87], or necessitate an additional buffer module to store a certain number of past instances [17, 36, 37, 94]. Several studies have explored online learning with class evolution [38, 75]. These studies construct their online learning algorithms within an ensemble framework of one-versus-all (OVA) classifiers. Each OVA classifier is dedicated to a distinct class, predicting whether an incoming instance belongs to this class or not. A new OVA classifier is initialized and incorporated into the ensemble for the emergence of a novel class. While current online learning algorithms have demonstrated notable performance improvements on handling class evolution, these algorithms rely on numerous constraints and underlying assumptions about the data stream. In real-world scenarios, the behavior of data streams is often non-ideal and unpredictable over time. To be specific,

- **Unexpected prior probability of the novel class.** Previous OVA frameworks [38, 75] adapt their models based on the underlying assumption that the number of instances from any single given class is always smaller than that from other classes. Following this assumption, to handle the class-imbalanced issue between the positive class and the negative class in each OVA classifier, these methods only under-sample instances from other classes when updating each OVA classifier. Such an assumption is often valid in stationary real-world scenarios [15]. However, in dynamic real-world data streams, instances from different classes do not arrive in an uniform and predictable manner. For instance, in the context of a *Twitter* data stream [50], an explosive new topic may suddenly gain prominence, or a

profuse network attack may follow the discovery of a system flaw [76]. In such cases, the novel class emerges with a dominant number of instances, making the aforementioned assumption invalid. The one-side under-sampling strategy in previous work results in the ensemble of OVA classifiers receiving more error feedback from the dominant emerging class than the existing classes, leading to an overemphasis on the novel class and, consequently, inaccurate predictions in the multi-class classification task.

- **Limited availability of true labels.** In real-world data stream mining, as mentioned above, instances are continuously generated, and novel classes emerge constantly. All existing online learning methods that handle class evolution are supervised algorithms [38, 75]. However, it is impractical for experts to label every instance of emerging novel classes. Labeling a small, cost-affordable number of instances for these novel classes provides a more feasible solution for model adaptation, presenting the problem of learning with class evolution under incomplete supervision [95].

- **Data distributed in open data space.** One common strong assumption about data streams is that the feature space remains unchanged throughout. A more practical scenario is that both the feature space and label space evolve over time. For example, the vocabulary people used to write tweets changes from year to year, and even from day to day. The topics also change. We refer to this phenomenon as open data space, with the property of feature evolution [23, 32, 59] and class evolution simultaneously. However, existing algorithms are only capable of handling two out of the three challenges, namely 1) free of data storage, 2) feature evolution, and 3) class evolution. To the best of our knowledge, no existing work is capable of handling open data space in the online learning manner.

In this thesis, we aims to develop utilitarian online learning algorithms to address the

aforementioned challenges associated with class evolution in real-world data streams.

1.2 Literature Review

1.2.1 Concept Drift

Class evolution is associated with the changes in the prior probabilities of classes, denoted as $p(y)$. It is also considered as a specific kind of concept drift, as it impacts the joint probability distribution $p(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathbb{R}^d$ is a feature vector of length d. However, existing research in the field of concept drift concentrates on the drift of the distribution in feature space, $p(\boldsymbol{x})$, and the decision boundary, $p(y|\boldsymbol{x})$ [55]. Algorithms designed to handle concept drift adaptation typically assume that the set of classes remains fixed and no novel class will emerge within the data stream [34]. As a result, these algorithms are not directly applicable in scenarios involving class evolution [75].

1.2.2 Concept Evolution

Research addressing class evolution primarily focuses on the concept evolution domain, which only deals with the emergence of novel classes [20], encompassing tasks such as novel class detection and novel class adaptation. In this thesis, our main focus is on the task of effectively adapting the model to class evolution in the data stream instead of detection. Based on the number of maintained classifiers, methods in the concept evolution literature are categorized into the single-model and the ensemble-model approaches.

Single-model approach. In this approach, a single multi-class classifier is adapted to recognize novel classes in the data stream. Neural network-based models, as proposed in [82] and [94], are designed to adapt to high-dimensional data streams with fully labeled instances in their buffer module. In the SACCOS [26] framework, the classifier

is retrained with recent partially labeled instances when novel classes emerge. A matrix sketching method is proposed in [91], employing an adapted global matrix to capture the information of stored instances from novel classes, alongside an extended local matrix for these classes. Clustering methods, such as EMC [19] and MINAS [17], construct novel micro-clusters on buffered fully or partially labeled buffered instances and incorporate these clusters into the learning model. DISSFCM [12] employs a semi-supervised fuzzy clustering algorithm on chunk data along with the past prototypes to update prototypes for classification. More recently, a prototype-based semi-supervised learning framework [21] is proposed, maintaining a dynamic set of prototypes at multiple levels of granularity. Prototypes for novel classes are extracted from the latest data chunk and incorporated into the classifier. Although the aforementioned approaches have achieved significant improvements in various application scenarios, these methods all necessitate storing large volumes of past instances to adapt the model to novel patterns in data streams.

Ensemble-model approach. In this approach, a common strategy is to replace the outdated base classifier with a newly trained one on the latest data. In LNCS [22], a set of base classifier is constructed for each data chunk to incrementally learn novel classes. Algorithm C&NCBM [51] maintains a fixed ensemble size of classifiers, replacing the least accurate base classifier with a newly trained one on the latest data chunk to capture the latest concept. Further investigations explore the strategy of building base classifiers on dynamic rather than fixed chunk sizes, as seen in SAND [36], ECHO [37], NovelDetector$^{\mathrm{DAE}}$ [65], and ESCR [93]. These methods use labeled instances and unlabeled instances with high-confidence pseudo-labels in the dynamic data chunk for the training of a new base classifier. SENCForest [64] introduces two levels of mechanisms for novel classes adaptation. The stored data instances in the buffer module are either utilized to grow subtrees in existing SENCForest trees, or train and incorporate a new SENCForest when the current SENCForest reaches the handling

threshold for the number of classes. Class-based ensemble is an alternative approach to construct ensemble models, such as CLAM [2] and SCANR [57]. These approaches update ensemble models with new per-class classifiers trained on incoming data chunks. However, the ensemble models depend on a batch of stored instances to construct new base classifier. In online learning scenarios, it is impractical to build an entirely new base classifier for each individual incoming instance to update the ensemble model.

As a summary, these methods, including both single-model and ensemble-model approaches, are inadequate for managing scenarios of online learning with class evolution.

1.2.3 Feature Evolution

Contrary to the common strong assumption of a fixed feature set, real-world data streams exhibit non-stationary dimensionality, with the data instances distributed in an open data space, where both the components of features and labels evolve with time. This problem is first investigated by M. Masud et al. in their proposed models DXMiner [58] and MCM [59]. These models behaves in the ensemble-based approach to address the class evolution issue. The feature space conversion method is designed to construct a homogeneous feature space for each data chunk to perform model prediction and training. However, as analyzed in the previous subsection, these models process data chunk-by-chunk and are unable to handle online learning scenarios.

Subsequent researches are mainly concentrated on online learning with either feature evolution or class evolution, but not both. In this section, we review the work of online learning with feature evolution, while a discussion of online learning with class evolution will follow in the subsequent section. In online learning with feature evolution, various scenarios have been investigated. In STSD [89] and OL$_{SF}$ [90], the passive-aggressive (PA) online updating rule [16] is utilized to handle the trapezoidal feature evolution, which is characterized by the emergence of novel features without the disappearance

of existing ones. An optimization objective is formulated to force the classifier to satisfy the hinge loss constraint while preserving inherent information on existing features and regularizing new parameters on novel features. An adaptive neural network with shortcut connections is designed in SLFN-S [7] to manage the nonlinear separable cases in data streams exhibiting trapezoidal feature evolution. The active learning [54] and the bandit feedback [31] scenarios are further investigated. The algorithm FESL [44] considers a different form of feature evolution, characterized by a brief overlapping period for disappearing features and novel features. The linear correlation between the disappearing features and novel features is used in FESL for mapping the old model onto the novel feature space. The concept drift issue is further investigated in algorithms EDM [92] and FEMC [68]. Random lengths of overlapping periods for different disappearing features are managed by the following work PUFE [45]. Lian et al. [52] integrate deep models in their algorithm OLD^{3}S to manage high-dimensional data streams. Capricious feature evolution has recently gained much attentions. Unlike previous forms of feature evolution, it imposes no specific assumptions on the dynamics of feature evolution, and the set of features changes randomly. In OLVF [6], the projection confidence is introduced to estimate whether the classifier is capable of making an accurate prediction in current feature space. OCDS [40] and GLSC [41] construct a homogeneous universe feature space by generative graphical models for model prediction and adaptation. AGDES [42] extends the solution to managing the semi-supervised scenarios. More recently, the algorithm OLI^{2}DS [85] which is constructed based on the PA update rule achieves state-of-the-art classification performance across diverse scenarios of feature evolution. The cumulative class ratio is incorporated in the loss term to balance error feedback between imbalanced classes. The model is also designed to prioritize features based on their informativeness, focusing more on those that provide more information in model adaptation. In their subsequent work OLIFL [86], these idea are applied to handle

the semi-supervised scenarios. Although a spectrum of different scenarios on feature evolution has been investigated, the set of classes is assumed to be a fixed set in these work. Consequently, methods in online learning with feature evolution are inadequate for addressing the problem of class evolution, a challenge that arises in data streams with open data space.

1.2.4 Online Learning with Class Evolution

Only a few researches focus on developing online learning approaches to handle data streams with class evolution. The adapted OVA decision trees are proposed in [38], with each tree distinguishing whether an incoming instance belongs to a specific class or not. A new tree will be initialized when a novel class emerges. Each labeled instance is used to update their corresponding tree, and one other tree that is most likely to misclassify this instance. Sun et al. [75] introduce the CBCE algorithm, which considers more forms of class evolution, rather than focusing solely on concept evolution. In their framework, an innovative activation/deactivation strategy is employed to adapt the ensemble of OVA classifiers to class disappearance and class reoccurrence. In addition, an one-side under-sampling method on instances from negative class is incorporated to address the dynamic class imbalance issue arising from the gradual evolution of classes. However, in these algorithms, it is assumed that the positive class (belonging to a specific class) is always the minority and the negative class (belonging to the other classes) is the majority for each OVA model. This assumption becomes invalid when a novel class emerges with a dominant amount. In addition, these algorithms are all supervised ones and cannot accommodate the semi-supervised data streams with class evolution. Furthermore, these methods assume a fixed feature space throughout, which is an overly idealistic assumption in real-world scenarios.

The issues identified in prior work motivate the development of more utilitarian

online learning algorithms for handling class evolution, as outlined below:

- **Issue of unexpected prior probability of the novel class.** As outlied in Section 1.2.2, most existing approaches rely on storing past data to adapt their model, making them inadequate for handling strict online learning scenarios. Although several studies have focused on online learning algorithms, as summarized in Section 1.2.4, these model are built based on an underlying assumtion that, at any time, the number of instances in any single class remains consistently lower than the sum of instances in all other classes. Consequently, they fail to resolve the challenge posed by unexpected prior probability of the novel class in online learning scenarios.

- **Issue of limited availability of true labels.** As outlined in Section 1.2.2, current semi-supervised approaches for addressing class evolution necessitate massive storage of historical data. Moreover, all existing online learning algorithms for handling class evolution are fully supervised, as reviewed in Section 1.2.4. To the best of our knowledge, no prior work is capable of handling online learning with class eovlution under semi-supervised scenarios.

- **Issue of data distributed in open data space.** As discussed in Section 1.2.3 and Section 1.2.4, approaches that jointly tackle class evolution and feature evolution operate data stream in a chunk-by-chunk manner, and therefore require storing massive historical data for model adaptation. Existing online learning algorithms are concentrated on either class evolution or feature evolution, not both. To the best of our knowledge, no prior work is capable of handling both feature evolution and class evolution simultaneously in an online learning context.

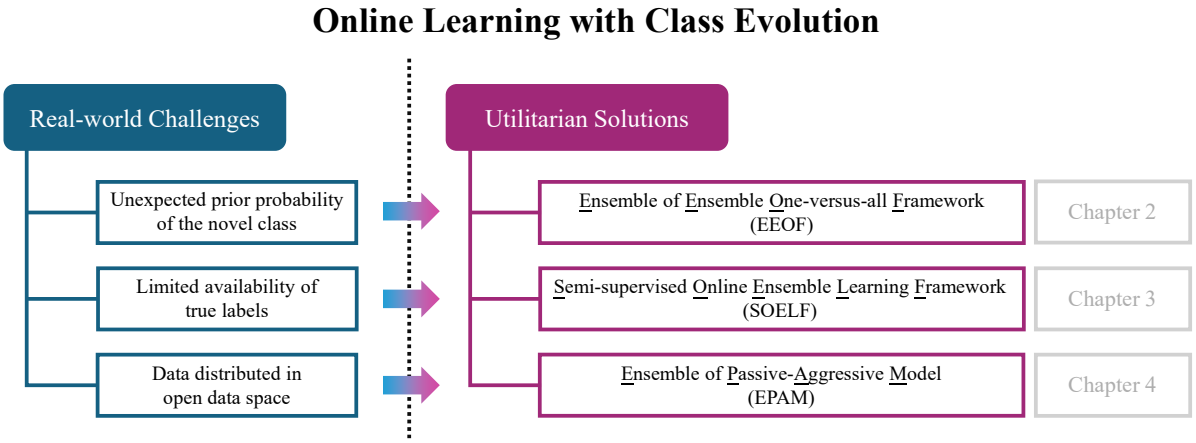


Figure 1.2: Research map for proposing utilitarian online learning algorithms to address real-world challenges.

1.3 Research Overview

This research investigates the online learning algorithms for problems and challenges in real-world data streams with class evolution. The research map can be found in Figure 1.2. The main research work and contributions of this thesis are concentrated on research questions as follows.

Research question 1: *How can we design an online learning algorithm that overcomes the constraints of the underlying assumptions made in previous approaches, in order to handle a broader spectrum of class evolution scenarios?*

In Chapter 2, we construct a novel Ensemble of Ensemble OVA Framework (EEOF) to address class evolution in the online learning procedure. The ensemble of ensemble architecture introduces diversity to enhance the binary classification of OVA classifiers. The one-side under-sampling strategies on instances from the negative class of an OVA classifier, as proposed in previous work, are based on the underlying assumption that the size of any given class is always smaller than the sum of other classes, which is no longer valid in real-world dynamic data streams, for example, a data stream with dominant emerging classes. Accordingly, a novel adaptive adaptation method is integrated in EEOF

to generate dynamic weights for the adaptation of each ensemble OVA classifier, ensuring a balanced error feedback between instances of the dominant emerging classes and other existing classes. Furthermore, in CBCE [75], the identification of disappeared classes relies solely on a stationary probability threshold. In scenarios where a class emerges with a a dominant amount, existing classes that generate instances with minimal probabilities are prone to being mistakenly regarded as disappeared classes. Accordingly, we design a confidence-triggered fallback mode to temporarily incorporates the inactivated OVA classifiers when the activated ones fail to provide reliable predictions. This work thoroughly investigates the challenging scenario of class evolution with dominant emerging classes. Comprehensive experimental studies on synthetic and real-world data streams demonstrate the superiority of EEOF in handling a broader spectrum of class evolution scenarios.

Research question 2: *How can we design an online learning algorithm to leverage the unlabeled instances in data streams, thereby addressing semi-supervised class evolution scenarios?*

The challenge of model adaption with class evolution under incomplete supervision is introduced in Chapter 3. Existing algorithms in online learning with class evolution are all supervised approaches. Although these algorithms are capable of adapting with limited labeled instances, they tend to excessively focus on labeled instances from emerging novel classes due to neglecting on the unlabeled data from existing classes. To overcome these limitations, we design a novel Semi-supervised Online Ensemble Learning Framework (SOELF). This framework adapts to data streams in an online learning manner. In this framework, a data generation module is constructed based on an ensemble online clustering model, enabling the generation of diverse instances from the current unlabeled instance to facilitate the model adaptation. The decaying G-mean

metric is utilized to assign more reliable pseudo-labels to these generated instances. Furthermore, the inverse-prior-ratio model adaptation strategy is designed to introduce more instances of existing classes for model adaptation when labeled instances of novel classes emerge, aiming to balancing the exploitation on unlabeled instances between different classes and preventing the model from excessively focusing on labeled novel classes. Above techniques endow SOELF with better performance in multi-class classification tasks across diverse data streams with class evolution under incomplete supervision.

Research question 3: *How can we design an online learning algorithm to manage class evolution scenarios with no fixed feature space, i.e. scenarios where data is distributed in open data space?*

Rather than assuming a fixed feature space throughout the data stream, Chapter 4 explores the challenge of feature evolution in online learning with class evolution. To the best of our knowledge, no existing approach addresses both feature evolution and class evolution simultaneously in an online learning context. Online learning in open data space is still an open problem. To initiate the investigation and shed some insights on this problem, we first construct several baseline models constructed from the state-of-the-art online learning approaches in both fields of class evolution and feature evolution. We then analyze their limitations and propose a novel Ensemble of Passive-Aggressive Model (EPAM) to manage online learning in open data space. In EPAM, a novel feature contributed bias classifier (FCBC) is incorporated as the base OVA classifier to handle feature evolution in data streams. The adaptive bias vector in FCBC dynamically adjusts to diverse feature combinations in open data space. Furthermore, a novel model adaptation strategy is utilized to balance error feedback among classes designated as the negative class in a base OVA classifier, enhancing the performance of EPAM on multi-class classification tasks in data streams with class evolution. Both

synthetic and real-world data streams demonstrate the superiority of EPAM on handling online learning with class evolution and feature evolution.

1.4 Structure of this Dissertation

The remainder of this thesis is structured around different real-world scenarios in online learning with class evolution that have been overlooked in previous researches but present significant challenges, as illustrated in Figure 1.2.

In real-world data streams, the prior probability of a novel class is generally unpredictable, thereby invalidating the underlying assumptions adopted in earlier studies. In response to this research question, Chapter 2 provides an in-depth investigation of the challenging scenarios with unexpected prior probability for the emerging novel class, and introduces an innovative ensemble of ensemble OVA framework, EEOF, to handle a broader spectrum of class evolution scenarios.

Based upon the robust ensemble of ensemble structure introduced in Chapter 2, Chapter 3 and Chapter 4 extend the study of online learning with class evolution by focusing on challenges related to limited label availability and evolving feature spaces, respectively. In real-world applications, the scarcity of labeled instances is a fundamental obstacle for online learning algorithms. Chapter 3 thoroughly investigates diverse scenarios with class evolution under incomplete supervision, and proposes a novel semi-supervised online ensemble learning framework, SOELF, to tackle this problem. Moreover, assuming a fixed feature space throughout the data stream is often unrealistic. Chapter 4 challenges a common and strong assumption about the real-world data streams that the feature space remains static over time, and provides a solution, ensemble of passive-aggressive model (EPAM), on handling the online learning with class evolution and feature evolution.

Finally, Chapter 5 provides the conclusion of this research, discusses its limitation,

and outlines potential directions for future work.

1.5 Summary

In this chapter, we introduce the research problems associated with online learning with class evolution in real-world data streams. A comprehensive literature reivew is conducted, revealing several critical yet overlooked challenges in exsiting studies: 1) the unexpected prior probability of the novel class, 2) the limited availability of true labels, and 3) the dynamic nature of data distributed in open data space. These challenges motivate the design of novel solutions, which are presented in detail in the subsequent chapters.

CHAPTER 2

ENSEMBLE OF ENSEMBLE ONE-VERSUS-ALL FRAMEWORK

2.1 Background

In data streams with the evolving class compositions, most existing approaches require the storage of past data for model adaptation [20]. These methods typically either process data in a chunk-by-chunk manner [19, 87] or integrate a buffer module [17, 36, 94]. Although several studies have developed online learning approaches [38, 75], these methods are built based on the assumption that the number of instances in any single class is consistently less than the sum number of other classes. This assumption becomes invalid when a class emerges with a dominant amount, e.g., news about a pandemic outbreak, significantly impairing the performance of existing methods. With such an unexpected prior probability of the novel class, the one-sided under-sampling strategies in these methods cause the classifiers to receive more error feedback from the dominant emerging class than the existing classes, leading to inaccurate predictions. Moreover, in scenarios where a class emerges with a dominant number, the state-of-the-art method,

CBCE [75], tends to mistakenly regard existing classes with minimal probabilities as disappeared ones, leading to unreliable predictions.

To address the above limitations, this chapter thoroughly investigates this challenging scenarios involving dominant emerging classes, which are not considered in previous work but prevalent in real-world applications. A novel ensemble of ensemble one-versus-all framework (EEOF) is proposed to address class evolution in the online learning procedure. The ensemble of ensemble architecture introduces diversity to enhance the performance of OVA classifiers. We attribute the failure of the previous OVA frameworks primarily to the improper assumption, and design a novel adaptive model adaptation method to generate dynamic weights for the learning of each ensemble OVA classifier, ensuring a balanced error feedback between the dominant emerging classes and other existing classes. This novel adaptive model adaptation method is built with no assumption about the distribution of classes in the data stream, catering the unsolved scenarios where a class emerges with a dominant number. To prevent existing classes being wrongly regarded as disappeared classes, a confidence-triggered fallback mode is designed. This mode temporarily incorporates the inactivated OVA classifiers in classification when the activated OVA classifiers fail to provide reliable predictions. Experiments on synthetic and real-world data streams demonstrate the superiority of the proposed framework EEOF.

2.2 Problem Formulation

In this section, a formal formulation for online learning with class evolution is introduced. Given a continuous data stream $\{(\boldsymbol{x}_t, y_t)\}_1^{+\infty}$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the feature vector of length d arriving at time t and $y_t \in \mathscr{Y}_t$ is the associated class label. Each class c_i in $\mathscr{Y}_t$ has positive prior probability at time t, i.e. $p_t^{(i)} > 0$. Each instance $(\boldsymbol{x}_t, y_t)$ arrived at time t follows the distribution $p_t(\boldsymbol{x}, y)$ at time t, i.e. $(\boldsymbol{x}_t, y_t) \sim p_t(\boldsymbol{x}, y)$. In online learning, only

one example $\boldsymbol{x}_t$ is available at time t. The true label y_t of $\boldsymbol{x}_t$ arrives after the prediction $\hat{y}_t$ is made before time $t+1$.

In the data stream, the set of classes $\mathscr{Y}_t$ may change over time, referred to as class evolution. Based on how classes evolve, class evolution is categorized into 3 principal forms: 1) class emergence, 2) class disappearance, and 3) class reoccurrence. A class c_{novel} emerges at time t if $c_{novel} \in \mathscr{Y}_t$ and $c_{novel} \notin \bigcup_{i=1}^{t-1} \mathscr{Y}_i$. A class $c_{disappear}$ disappears at time t if $c_{disappear} \notin \mathscr{Y}_t$ and $c_{disappear} \in \mathscr{Y}_{t-1}$. A class $c_{reoccur}$ disappears at time d and reoccurs at time t if $c_{reoccur} \in \mathscr{Y}_t$ and $c_{reoccur} \notin \bigcup_{i=d}^{t-1} \mathscr{Y}_i$ and $c_{reoccur} \in \mathscr{Y}_{d-1}$.

The task of online learning with class evolution is to construct an online classifier $\mathscr{F}$ that is capable of 1) making accurate prediction $\hat{y}_t$ for arriving example $\boldsymbol{x}_t$ at time t, and 2) updating after receiving the true label y_t.

2.3 Proposed Algorithm

In light of the limitations of existing approaches, this chapter delivers the ensemble of ensemble OVA framework (EEOF) to tackle the problem of online learning with class evolution, including the unsolved scenario of class emergence with a dominant amount. In this section, we will delve into the details of our proposed framework.

2.3.1 Ensemble of ensemble OVA framework

In existing OVA frameworks, k OVA classifiers are maintained for the k-class classification problem. Each OVA classifier serves as a binary classifier, predicting whether an instance $\boldsymbol{x}_t$ belongs to a specific class c_i (designated as the positive class, $+1$), or to the remaining $k-1$ classes (designated as the negative class, -1). Given $\mathscr{F}_i$ providing the prediction score for class c_i, the instance $\boldsymbol{x}_t$ is classified into the class estimated with the

highest prediction score:

$$\hat{y}_t = \arg\max_i \mathscr{F}_i(\boldsymbol{x}_t). \tag{2.1}$$

In this research, instead of maintaining one ensemble model, we maintain an ensemble of OVA classifiers, denoted as $\mathscr{F}_i$, for each class c_i to introduce diversity among the inner OVA classifiers and achieve a more accurate estimation of the prediction score. We name this approach to be an ensemble of ensemble OVA framework, denoted as $\mathscr{F} = \{\mathscr{F}_1, \mathscr{F}_2, ..., \mathscr{F}_n\}$, where each element, ensemble OVA model, is denoted as $\mathscr{F}_i = \{\mathscr{F}_{i1}, \mathscr{F}_{i2}, ..., \mathscr{F}_{im}\}$. Here, n is the total number of classes that appear in the data stream, and m is the ensemble size. The prediction score for class c_i is estimated by the average of prediction scores across all base classifiers in $\mathscr{F}_i$. By constructing an ensemble of OVA classifiers for each class c_i, the algorithm is capable of reducing the risk of choosing the wrong base OVA classifier and therefore providing more accurate prediction scores [18]. The final classification result $\hat{y}_t$ is determined as:

$$\hat{y}_t = \arg\max_i \frac{1}{m} \sum_{j=1}^{m} \mathscr{F}_{ij}(\boldsymbol{x}_t). \tag{2.2}$$

2.3.2 Model Adaptation

An inherent challenge in the ensemble of ensemble OVA arises in adapting the inner base classifiers to the evolving data stream. Considering the nature of the OVA classifier in a k-class classification problem, where $k > 2$, at least $k - 1$ OVA classifiers will encounter issues related to class imbalance during the model adaptation process. The imbalance in error feedback between the positive and negative classes can result in unsatisfactory prediction scores being produced by the OVA classifier. In previous work such as [38] and [75], the assumption is made that the positive class represents the minority class, while the negative class represents the majority class. This assumption is often valid in stationary real-world scenarios, where the number of instances from one class is smaller

compared to the number of instances from other classes [15]. However, in dynamic real-world data streams, instances from different classes do not arrive in a uniform and predictable manner. In the specific scenario where class emergence with a dominant amount occurs, the one-sided under-sampling strategies employed in previous work on the negative class will make the new OVA classifier for the emerging class to be trained predominantly on this class only, leading to a tendency of overestimating the prediction score for that class.

To address this issue, we propose and incorporate an integrated adaptive cost-sensitive model adaptation method in EEOF to balance the error feedback between the dominant emerging class and the other classes. In the ensemble OVA model for the emerging class, the loss on instances from the negative class is overweighted. In the ensemble OVA models for other classes, this dominant emerging class is treated as part of the negative class, and the loss is underweighted along with other negative classes. This entire adjustment procedure relies on the dynamic prior probability $p_t^{(i)}$ of each class c_i at time t. We utilize the time decayed method [78, 79] to estimate $p_t^{(i)}$:

$$\hat{p}_t^{(i)} = \beta \hat{p}_{t-1}^{(i)} + (1-\beta)\mathbb{1}[y_t == c_i], \tag{2.3}$$

where $\hat{p}_t^{(i)}$ is the estimated prior probability, β is the time decay factor. For each ensemble OVA model $\mathscr{F}_i$ for class c_i, assuming the loss weight for the positive class is set to 1 ($r_+ = 1$), the loss weight for the negative class is adjusted as follows:

$$r_- = \frac{r_+ \hat{p}_t^{(i)}}{1-\hat{p}_t^{(i)}} = \frac{\hat{p}_t^{(i)}}{1-\hat{p}_t^{(i)}}. \tag{2.4}$$

This adaptive method is not limited to the considered scenario of dominant emerging class, and can be applied to various other dynamic data streams exhibiting similar imbalance issues, proving a versatile solution.

In the online learning scenario, model adaptation is performed with only a single instance. To introduce the diversity within the ensemble OVA model $\mathscr{F}_i$, we generate the

Algorithm 2.1 Adaptation

Input:

- Instance $(\boldsymbol{x}_t, y_t)$
- EEOF Model $\mathscr{F} = \{\mathscr{F}_i\}$, where $\mathscr{F}_i = \{\mathscr{F}_{ij}\}$
- Score matrix $\hat{Y}_t = [\hat{y}_{t,ij}]$, where $\hat{y}_{t,ij} = \mathscr{F}_{ij}(\boldsymbol{x}_t)$
- Estimated prior probability vector $\hat{\boldsymbol{p}}_t = [\hat{p}_t^{(i)}]$

Output:

- Updated EEOF model $\mathscr{F}$

for $i \leftarrow 1 : n$ **do**
  for $j \leftarrow 1 : m$ **do**
    if $y_t == c_i$ **then**
      $r_+ \sim Poisson(1)$
      if $r_+ > 0$ **then**
        Update $\mathscr{F}_{ij}$ with $r_+ * Loss(+1, \hat{y}_{t,ij})$
      end if
    else
      $r_- \sim Poisson(\hat{p}_t^{(i)}/(1-\hat{p}_t^{(i)}))$
      if $r_- > 0$ **then**
        Update $\mathscr{F}_{ij}$ with $r_- * Loss(-1, \hat{y}_{t,ij})$
      end if
    end if
  end for
end for

loss weight for each base OVA classifier $\mathscr{F}_{ij}$ using the Poisson distribution, a commonly employed technique in various online learning algorithms [67]. The detailed pseudo-code is displayed in Algorithm 2.1.

2.3.3 Confidence-triggered Fallback Mode

In data streams with class evolution, the set of classes being present may undergo continuous changes. When a novel class emerges, a new ensemble OVA model is initialized and integrated into the whole framework to handle this class. Conversely, when an existing class disappears, the ensemble OVA model corresponding to that class is deactivated

and excluded from the classification process in Eq. (2.2). If a disappeared class reoccurs, the previously deactivated ensemble OVA model for that class is reactivated and used for classification again. This activate/deactivate strategy is natural for handling class evolution, eliminating the inference of confidence scores from disappeared classes and achieving higher classification accuracy.

In the previous work [75], a fixed disappearance threshold for the prior probability is used. Classes with an estimated probability $\hat{p}_t^{(i)}$ below the threshold are considered as disappeared classes. However, using a stationary disappearance threshold is not suitable for a dynamic data stream. The choice of an appropriate threshold should ensure that it accurately identifies the truly disappeared classes, while avoiding the misclassification of existing classes with minimal prior probability as disappeared classes. In the specific scenario where class emergence with a dominant amount occurs, we have observed that existing classes often appear with a minimal prior probability due to the overwhelming presence of the emerging classes. Consequently, these existing classes are mistakenly excluded from the classification process, resulting in unreliable classification results.

An intuitive solution to address this issue could involve selecting a more appropriate disappearance threshold. However, in the absence of prior knowledge about the data stream, selecting an optimal disappearance threshold becomes impractical. Moreover, in an unpredictable dynamic data stream, using a fixed threshold becomes unreliable as it may not cater all possible cases that could arise in the data stream.

Given all the aforementioned issues, our focus is primarily on addressing the problem of unreliable classification. To tackle this, we introduce a confidence-triggered fallback mode into the classification process of EEOF, aiming to enhance the reliability of predictions. The implementation of this approach is illustrated in Algorithm 2.2. In EEOF, each activated ensemble OVA model $\mathscr{F}_i$ is responsible for predicting a score s_i to determine if an instance belong to class c_i or not. A unreliable classification is made when all

Algorithm 2.2 Classification

Input:

- Instance $\boldsymbol{x}_t$
- EEOF Model $\mathscr{F} = \{\mathscr{F}_i\}$, where $\mathscr{F}_i = \{\mathscr{F}_{ij}\}$

Output:

- Prediction $\hat{y}_t$
- Score matrix $\hat{Y}_t = [\hat{y}_{t,ij}]$

$\hat{Y}_t \leftarrow [\hat{y}_{t,ij}]$, $\hat{y}_{t,ij} \leftarrow 0$
$S = [s_i]$, $s_i \leftarrow -1$
for each $\mathscr{F}_i \in \mathscr{F}$ **do**
  for each base OVA classifier $\mathscr{F}_{ij} \in \mathscr{F}_i$ **do**
    $\hat{y}_{t,ij} \leftarrow \mathscr{F}_{ij}(\boldsymbol{x}_t)$
  end for
  if $\mathscr{F}_i$ is activated **then**
    $s_i \leftarrow \frac{1}{m}\sum_{j=1}^{m} \hat{y}_{t,ij}$
  end if
end for
// Confidence-triggered fallback mode
$confidence\_score = \max(s_1, ..., s_n)$
if $confidence\_score \leq confidence\_threshold$ **then**
  for each $\mathscr{F}_i \in \mathscr{F}$ **do**
    if $\mathscr{F}_i$ is inactivated **then**
      $s_i \leftarrow \frac{1}{m}\sum_{j=1}^{m} \hat{y}_{t,ij}$
    end if
  end for
end if
$\hat{y}_t \leftarrow \arg\max_i s_i$

activated ensemble OVA models predict that the instance $\boldsymbol{x}_t$ does not belong to any of their corresponding classes. Classifying $\boldsymbol{x}_t$ to class with the highest prediction score is unreliable because it may actually belong to one of the classes represented by inactivated ensemble OVA models. Therefore, in our confidence-triggered fallback mode, we define an unreliable classification as one where all prediction scores are below a confidence threshold. In the adaptation process of $\mathscr{F}_i$, the instance $\boldsymbol{x}_t$ is assigned a positive label $+1$ if it belongs to class c_i, and assigned a negative label -1 otherwise. During prediction, a

negative prediction score implies that $\boldsymbol{x}_t$ does not belong to class c_i. Consequently, we set the value of confidence threshold as 0. When an unreliable classification occurs, the inactivated ensemble OVA models are temporally activated to classify the instance $\boldsymbol{x}_t$.

2.3.4 Overview

After introducing each novel component in the classification process, namely the ensemble of ensemble OVA framework, model adaptation, and the confidence-triggered fallback mode, we can now integrate these techniques to effectively address the problem of online learning with class evolution. An overview of the proposed EEOF is displayed in Algorithm 2.3.

2.4 Experimental Study

In this section, we present a comprehensive experimental study to demonstrate the effectiveness of EEOF in addressing the problem of online learning with class evolution. We compare EEOF with the state-of-the-art method on a range of synthetic and real-world data streams. Additionally, we conduct an ablation study to highlight the significance of the different components in EEOF.

2.4.1 Experimental Settings

1) Data Stream

We utilize three tabular datasets, namely Letter Recognition (16 attributes) [72], Statlog (Landsat Satellite) (36 attributes) [74], and Covertype (54 attributes) [8], as well as an image dataset, MNIST (784 attributes) [49] for generating synthetic data streams. To thoroughly explore the previously neglected scenario of class emergence with a dominant amount, we consider data stream with $k-1$ existing classes and 1

Algorithm 2.3 An overview of EEOF in data stream

$\mathscr{F} \leftarrow \emptyset$ // *Initialize EEOF model*
$\hat{\boldsymbol{p}}_t \leftarrow \emptyset$ // *Initialize estimated prior probability vector*
while Instance $\boldsymbol{x}_t$ arrives **do**
  if $t == 1$ **then**
    // *The first instance in data stream*
    $\hat{y}_t \leftarrow 0$
    $\hat{Y}_t \leftarrow 0$
  else
    $\hat{y}_t, \hat{Y}_t \leftarrow Classification(\mathscr{F}, \boldsymbol{x}_t)$
  end if
  Receive true label y_t
  Update $\hat{\boldsymbol{p}}_t$ with the occurrence of class y_t
  if no $\mathscr{F}_i$ in $\mathscr{F}$ is available for y_t **then**
    // *Class emergence*
    Initialize $\mathscr{F}_{novel}$ with m base classifier for class y_t
    Initialize $\hat{p}_{novel}$ for class y_t
    $\mathscr{F} \leftarrow \mathscr{F} \cup \mathscr{F}_{novel}$
    $\hat{\boldsymbol{p}}_t \leftarrow \hat{\boldsymbol{p}}_t \cup \hat{p}_{novel}$
  else if $\mathscr{F}_i$ in $\mathscr{F}$ is available for y_t and $\hat{p}_t^{(i)} == 0$ **then**
    // *Class reoccurrence*
    Activate $\mathscr{F}_i$ for classification
  end if
  for $\hat{p}_t^{(i)} \in \hat{\boldsymbol{p}}_t$ **do**
    if $\hat{p}_t^{(i)} < disappearance\_threshold$ **then**
      // *Class disappearance*
      Inactivate $\mathscr{F}_i$ for classification
      $\hat{p}_t^{(i)} \leftarrow 0$
    end if
  end for
  $\mathscr{F} = Adaptation(\mathscr{F}, \hat{\boldsymbol{p}}_t, \hat{Y}_t, \boldsymbol{x}_t, y_t)$
end while

emerging class. We use the curve of Gaussian distribution to model the change of $p(y)$ of the emerging class with time t, as depicted in Figure 2.1. The rate of increase is defined by the standard variation σ of the Gaussian distribution. The duration of the dominance is represented by the parameter T_{max}. The extend of dominance is described by the maximum prior probability p_{max} in the data stream, which is the peak value of the curve of Gaussian distribution. The remaining classes in the data streams are considered as

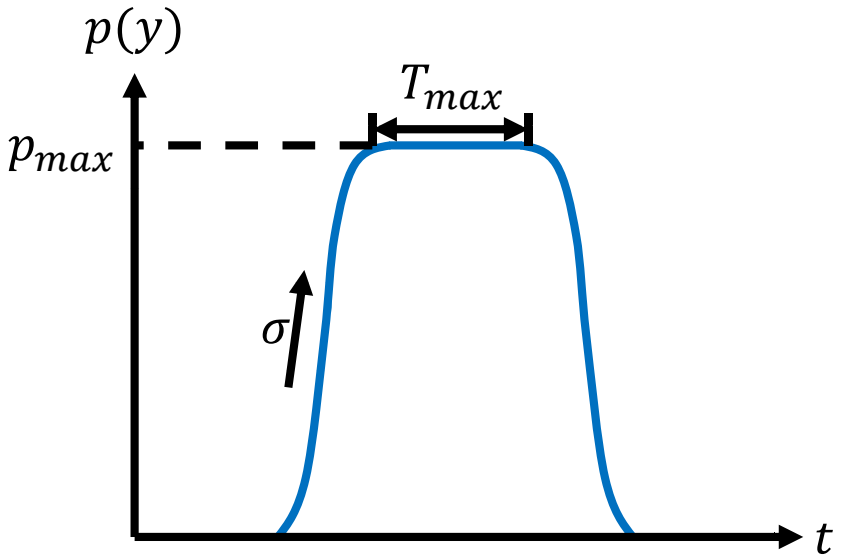


Figure 2.1: The curve of the prior probability of the emerging class with a dominant amount described by p_{max}, T_{max}, and σ.

existing classes with the same positive prior probability. In these synthetic data streams, we choose 4 classes as existing classes and 1 class as emerging class. The default values for these three parameters are set as follows: $p_{max} = 0.95$, $\sigma = 1000 * \frac{1}{3}$, $T_{max} = 4000$. We vary the values of these parameters to explore different scenarios. Specifically, we vary p_{max} with values ranging in $[0.20, 0.35, 0.5, 0.65, 0.80, 0.95]$ (from subordinate to dominant), σ with values $1000 * [\frac{1}{3}, \frac{2}{3}, 1, \frac{4}{3}]$, and T_{max} with values $[0, 1000, 2000, 3000, 4000]$. The prior probability curves of the emerging class are shown in Figure 2.2(a)(b)(c). In addition to the previously mentioned scenarios, we also simulate the scenario of class disappearance and reoccurrence to observe the performance of EEOF. We choose 4 classes as existing classes, and designate 1 class as the disappeared and reoccurred class. The prior probability curves of classes are illustrated in Figure 2.2(d).

We also evaluate our model on seven real-world data streams that exhibit the problem of class evolution. These data streams include four preprocessed data streams of TwitterCrawl data stream [50], Poker-hand data stream [13], KDDCUP99 data stream [76],

and UNSW-NB15 data stream [63]. The TwitterCrawl data stream comprises 50 million tweets posted between 2008 and 2011. The label for each tweet is determined by the hashtag describing its topic. We utilize the Tweet Stream - 20 classes, and Tweet Stream - A/B/C, as preprocessed in [75]. In the Poker-hand data stream, each record represents a poker hand consisting of five playing cards, and the labels correspond to different possible poker hands. The KDDCUP99 data stream contains network connection records associated with normal connection and various types of intrusion connections. The full version of this data stream contains five million records. For our experiments, we use the commonly considered 10% sample size version. The UNSW-NB15 data stream comprises realistic network traffic generated by the IXIA PerfectStorm tool and is labeled with 9 families of attacks plus normal flows. The key characteristics of all above real-world data streams are displayed in Table 2.1.

2) Parameter Settings

To the best of our knowledge, only two existing approaches are designed for online learning with class evolution. In our experimental evaluation, we compare our algorithm EEOF to the state-of-the-art algorithm CBCE [75]. To ensure a fair comparison, we use the same base classifier model, truncated Kernelized Logistic Regression (KLR) [47], and these same values for hyperparameters. In KLR, we select the Gaussian Kernel as the kernel function, and the kernel width σ_{kernel} is set to the 5-th percentile of the pairwise distances between all pairs of instances [56]. We conduct a variant of 5-folder cross-validation, where four out of every five instances are used for training, and the remaining one instance is used for validation. This procedure is used to tune the hyper-parameters, specifically the regularization parameter λ and the learning rate η. The tuned hyperparameters for these data streams are listed in Table 2.2. Other hyperparameters are set to commonly recommended or empirically determined values:

Table 2.1: Characteristics of real-world data streams.

Data Stream	**Instances**	**Features**	**Classes**	**Class Evolution Types**
Tweet Stream-A	39600	242	3	Class Emergence
Tweet Stream-B	15004	247	3	Class Disappearance and Reoccurrence
Tweet Stream-C	68750	242	4	Class Emergence
Tweet Stream-20 classes	143381	524	20	Class Emergence, Disappearance and Reoccurrence
KDDCUP99	494021	117	23	Class Emergence, Disappearance and Reoccurrence
Poker-hand	829201	25	10	Class Emergence, Disappearance and Reoccurrence
UNSW-NB15	1087136	199	10	Class Emergence, Disappearance and Reoccurrence

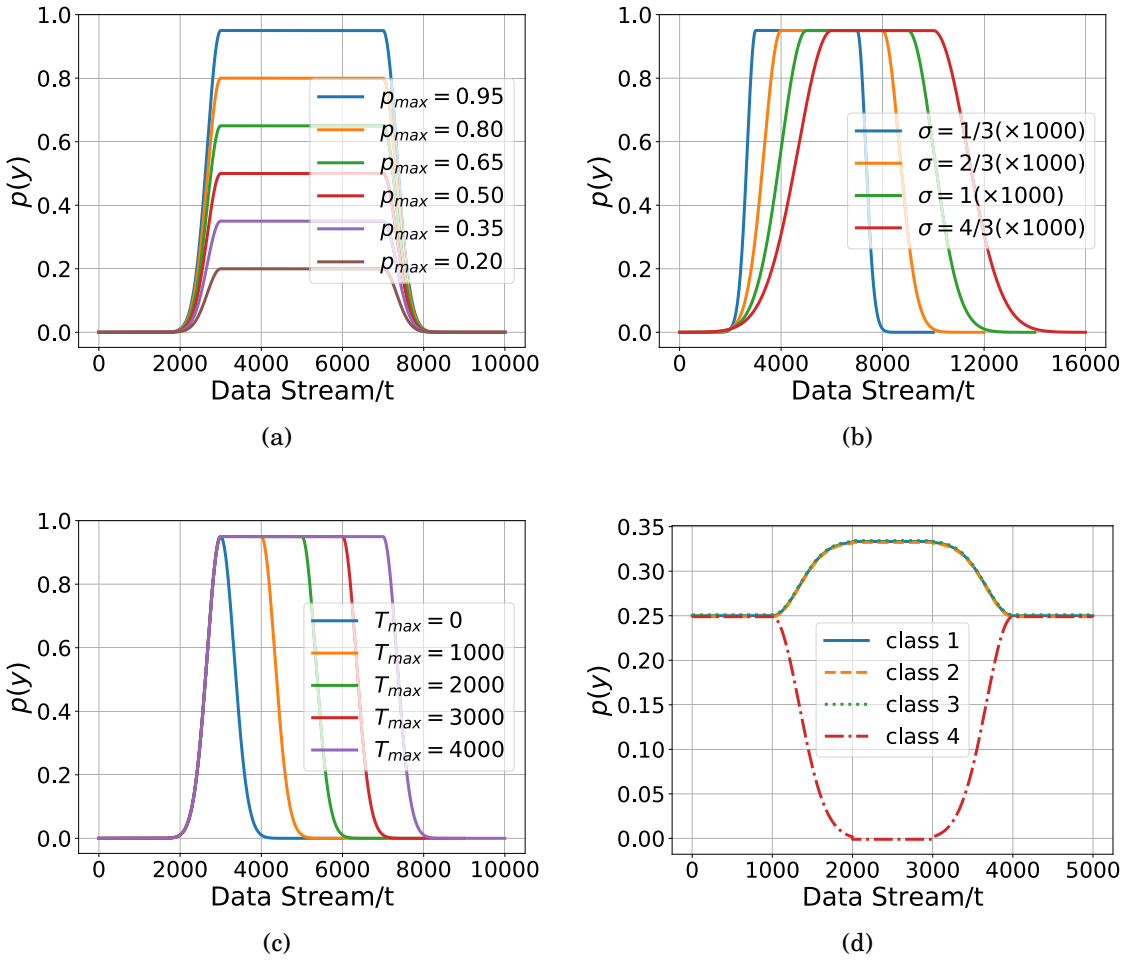


Figure 2.2: (a)(b)(c): Different scenarios with varying values for parameters p_{max}, T_{max}, and σ. (d): The prior probability curves of classes in the scenario of class disappearance and class reoccurence.

maximum stored kernel vectors in KLR is set to 5000, the decayed ratio $\beta = 0.9$, the disappearance threshold is set to 10^{-5}, and ensemble size m in each ensemble OVA model is 10.

3) Evaluation Metrics

In the online learning scenario, the model employed to classify each instance continuously adapts, resulting a different model for each successive instance. Metrics that incorporate a forgetting mechanism are more suitable for evaluating the performance in this context [25]. Additionally, in the data stream with class evolution, the distribution of

Table 2.2: Tuned Hyperparameters of KLR for Data Streams

Data Stream	η	λ	σ_{kernel}
Letter (Synthetic)	0.01	1	1
Statlog (Synthetic)	0.01	1	14
Covertype (Synthetic)	2	0.0001	5.4
MNIST (Synthetic)	6	0.0001	7
TwitterCrawl (Real-world)	0.3	0.0005	0.13
KDDCUP99 (Real-world)	0.0001	0.0001	0.3
Poker-hand (Real-world)	8	0.001	3.4
UNSW-NB15 (Real-world)	0.0001	0.0001	0.34

instances across different classes is dynamically imbalanced. The error estimators using sliding windows as defined in [25] are insufficient to measure the overall performance of all classes. While the average of the time-decayed G-mean is commonly used in the field of online learning [73, 80, 81], this metrics does not account for changes in the set of classes. The time-decayed recalls for disappeared classes remain constant, and are still included after these classes have disappeared. In [75], the average of G-mean in every data chunk is calculated. However, this metrics is primarily used to compare their method with chunk-based algorithms, and is improper to monitor algorithms in the online learning scenario.

To assess the overall performance of a model across the data stream, we apply the G-mean using sliding windows to monitor the performance of multi-class classification, and then average these results. The G-mean using a sliding window is defined as:

$$G_W(t) = G\text{-}mean(\{(y_i, \hat{y}_i)\}_{t-W+1}^{t}), \tag{2.5}$$

where W is the sliding window size set to 200. The average of G-mean using sliding windows on the data stream with a total length of N is defined as:

$$\text{Average of G-mean} = \frac{1}{N-W+1}\sum_{i=W}^{N} G_W(i). \tag{2.6}$$

Similar to the previous work [75], we conduct 10 independent runs for each algorithm on each data stream to reduce the impact of random variations, and the significance of the

difference between algorithms is examined using the Wilcoxon rank sum test [29]. Our source code is available at `https://github.com/lflfdxfn/EEOF`.

2.4.2 Experimental Analysis

1) Performance in Synthetic Data Streams

The performance of our method EEOF and the compared method CBCE on synthetic data streams with class emergence in a dominant amount is displayed in Figure 2.3. As shown in Figure 2.3(a)(d)(g)(j), prior probability curves of emerging classes with different dominant levels are explored. The average G-mean of CBCE on data streams does not change obviously when p_{max} is smaller than 0.8, but drops rapidly when $p_{max} = 0.95$. In cases where $p = 0.95$, the novel class emerges with a considerably high prior probability, causing the model adaptation in CBCE to focus more on the dominant novel class while weakening the classification performance on existing classes. Moreover, when the novel class emerges with a prior probability of 0.95, the existing classes in the data stream are sampled with a prior probability of 1.25%, which may be mistakenly regarded as disappeared classes and excluded from the classification process. Our algorithm EEOF outperforms CBCE in all cases and the performance degradation is slower almost in all cases as p_{max} reaches 0.95. Figures 2.3(b)(e)(h)(k) illustrate the investigations on different rates of the increasing prior probability for the novel class. The performance of CBCE deteriorates in all cases as σ decreases. In contrast, our algorithm EEOF outperforms CBCE across all choices of σ, and the performance remains stable in data stream cases of Letter, Statlog and MNIST. In Figure 2.3(c)(f)(i)(l), EEOF and CBCE are examined on data streams with different lengths of duration for the dominant emerging class. In CBCE, due to the model adaptation process is constructed based on an incomplete assumption, the performance drops quickly as the emerging class keeps dominant in the prior probability. Our method EEOF maintains an unaffected high

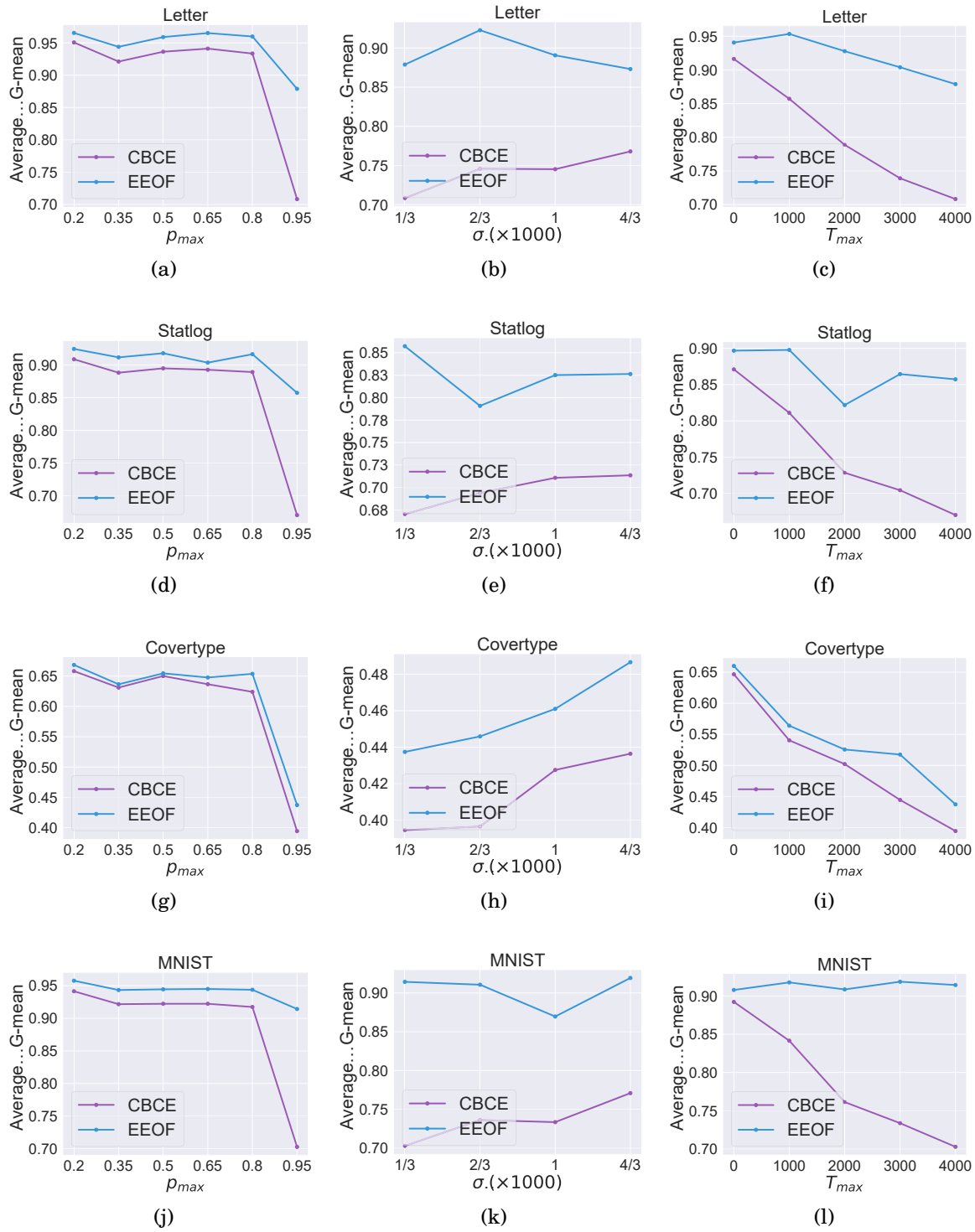


Figure 2.3: The average G-mean values of CBCE and EEOF in diverse synthetic data streams for class evolution with a dominant emerging class.

performance in the MNIST case and less affected higher performance in other cases. Such a thoroughly experimental study on the unsolved scenarios of the class emergence with dominant amount demonstrate the superiority of our proposed algorithm.

In addition to the aforementioned scenarios, we conduct an experimental study on synthetic data streams with class disappearance and class reoccurrence. In Table 2.3, the mean and standard deviation results for EEOF and CBCE in 10 runs on these synthetic data streams is presented. It is evident that EEOF performs significantly better than CBCE in all synthetic data streams, further proving the superiority of EEOF on handling diverse class evolution scenarios.

Table 2.3: The average of G-mean (mean/std) in 10 runs on synthetic data streams with class disappearance and class reoccurrence.

Algorithm	**Letter**	**Statlog**	**Covertype**	**MNIST**
EEOF	**0.9572/0.0006**†	**0.8864/0.0004**†	**0.7257/0.0024**†	**0.9547/0.0137**†
CBCE	0.9374/0.0004	0.8578/0.0015	0.7008/0.0021	0.9022/0.0035

* The best mean result for each data stream is highlighted in boldface. If this result is significantly better than the result obtained from the other algorithm (Wilcoxon rank sum test at 95% confidence level), it is marked with a † symbol.

2) Performance in Real-world Data Streams

The performance of EEOF and CBCE on real-world data streams is presented in Table 2.4. EEOF consistently achieves significantly higher G-mean results in all real-world data streams, except for Tweet Stream - B. As shown in Table 2.1, Tweet Stream - B exhibits class disappearance and reoccurrence, which falls outside the focused scenarios of our method EEOF. The underlying assumption in CBCE still holds in this case. However, the performance difference between EEOF and CBCE in Tweet Stream - B is negligible compared to the improvements of EEOF in the other real-world data streams, particularly in the case of the KDDCUP99 data stream. These results on real-world data streams effectively demonstrate the capability of our proposed method in handling diverse class evolution scenarios.

Table 2.4: The average of G-mean (mean/std) in 10 runs on real-world data streams.

Data Stream	EEOF	CBCE
Tweet Stream - A	**0.6444/0.0064**†	0.6226/0.0017
Tweet Stream - B	0.7245/0.0014	**0.7284/0.0020**†
Tweet Stream - C	**0.6205/0.0025**†	0.5957/0.0020
Tweet Stream - 20 classes	**0.0586/0.0009**†	0.0571/0.0014
KDDCUP99	**0.8410/0.0005**†	0.2593/0.0002
Poker-hand	**0.3956/0.0089**†	0.3839/0.0026
UNSW-NB15	**0.8518/0.0001**†	0.8513/0.0001

* The best mean result for each data stream is highlighted in boldface. If this result is significantly better than the result obtained from the other algorithm (Wilcoxon rank sum test at 95% confidence level), it is marked with a † symbol.

3) Ablation Study

To assess the importance of each component within EEOF in handling the unsolved scenarios and real-world data streams, we conduct an ablation study by investigating following three variants of EEOF:

- EOF: In the variant Ensemble of OVA Framework (EOF), a single OVA model replaces the ensemble OVA model in EEOF. It can also be regarded as EEOF with the hyperparameter $m = 1$.
- EEOF-Adapt: In the variant EEOF-Adapt algorithm, the model adaptation method designed in EEOF is replaced by the method in CBCE.
- EEOF-Pred: In the variant EEOF-Pred algorithm, the confidence-triggered fallback mode is removed from the framework.

We compare the performance of these variants with that of EEOF on our focused scenarios of class emergence with a dominant amount, as well as on the real-world data streams. The results are presented in Table 2.5 and Table 2.6, respectively. For simplicity, in Table 2.5, we set the same default values for p_{max}, σ, and T_{max}, and

vary them by choosing the the minimum and maximum values of the selected ranges. The best results are highlighted in boldface. If EEOF significantly outperforms at least two variants, or a variant performs significantly better than EEOF (Wilcoxon rank sum test at 95% confidence level), the best result of the better-performing algorithm is marked with a † symbol. As observed in the columns of EEOF and EOF in both tables, the ensemble of ensemble OVA structure improves the performance in all cases. The replacement of the designed model adaptation method causes the EOF algorithm to perform weaker in almost all class emergence scenarios, except for the MNIST case with $p_{max} = 0.95$, $T_{max} = 0$, $\sigma = 1/3(\times 1000)$ and the MNIST case with $p_{max} = 0.20$, $T_{max} = 4000$, $\sigma = 1/3(\times 1000)$. This is due to the short duration of the dominant emerging class in the data stream and the non-dominant probability of the emerging class. It can also be concluded that the confidence-triggered fallback mode helps to achieve equivalent or better performance by comparing EEOF with the variant EEOF-Pred in both tables, except for the real-world data stream Tweet Stream - B, where the degradation is still considerably slight. Hence, the default framework of EEOF is generally recommended to addressing scenarios involving class emergence with a dominant amount, and diverse real-world data streams with class evolution.

Table 2.5: The average of G-mean (mean/std) in 10 runs of the ablation study on synthetic data streams with class emergence.

Data Stream	p_{max}	T_{max}	$\sigma(\times 1000)$	**EEOF**	**EOF**	**EEOF-Adapt**	**EEOF-Pred**
Letter	0.20	4000	1/3	**0.9654/0.0010**†	0.9562/0.0061	0.9638/0.0002	0.9524/0.0010
Letter	0.95	4000	1/3	**0.8789/0.0161**†	0.8463/0.0092	0.8448/0.0102	0.7105/0.0010
Letter	0.95	0	1/3	**0.9406/0.0015**†	0.9338/0.0034	0.9387/0.0009	0.9176/0.0010
Letter	0.95	4000	4/3	**0.8732/0.0022**†	0.8685/0.0067	0.8601/0.0006	0.7695/0.0004
Statlog	0.20	4000	1/3	**0.9245/0.0005**†	0.9159/0.0031	0.9223/0.0005	0.9120/0.0006
Statlog	0.95	4000	1/3	**0.8573/0.0023**†	0.8411/0.0164	0.8046/0.0054	0.6699/0.0004
Statlog	0.95	0	1/3	**0.8968/0.0007**†	0.8898/0.0038	0.8956/0.0005	0.8728/0.0007
Statlog	0.95	4000	4/3	**0.8263/0.0020**†	0.8075/0.0126	0.7951/0.0010	0.7188/0.0016
Covertype	0.20	4000	1/3	**0**.6683/0.0050	0.6510/0.0100	**0.6716/0.0030**	0.6683/0.0050
Covertype	0.95	4000	1/3	**0.4374/0.0139**†	0.4239/0.0153	0.4108/0.0071	0.4221/0.0081
Covertype	0.95	0	1/3	**0.6600/0.0059**†	0.6387/0.0135	0.6591/0.0038	**0.6600/0.0059**
Covertype	0.95	4000	4/3	**0.4866/0.0102**†	0.4699/0.0164	0.4374/0.0073	0.4683/0.0108
MNIST	0.20	4000	1/3	0.9577/0.0009	0.9443/0.0028	**0.9587/0.0006**†	0.9452/0.0009
MNIST	0.95	4000	1/3	**0.9143/0.0109**†	0.8874/0.0180	0.8941/0.0117	0.7046/0.0011
MNIST	0.95	0	1/3	0.9078/0.0052	0.8922/0.0080	**0.9153/0.0023**†	0.8911/0.0055
MNIST	0.95	4000	4/3	**0.9193/0.0082**†	0.8904/0.0162	0.9042/0.0066	0.7725/0.0065

* The best mean result is highlighted in boldface. If EEOF significantly outperforms at least two variants, or a variant performs significantly better than EEOF (Wilcoxon rank sum test at 95% confidence level), it is marked with a † symbol.

Table 2.6: The average of G-mean (mean/std) in 10 runs of the ablation study on real-world data streams.

Data Stream	**EEOF**	**EOF**	**EEOF-Adapt**	**EEOF-Pred**
Tweet Stream - A	**0.6444/0.0064**†	0.6239/0.0071	0.6475/0.0022	0.6200/0.0042
Tweet Stream - B	0.7245/0.0014	0.7005/0.0031	0.7258/0.0007	**0.7288/0.0016**†
Tweet Stream - C	**0.6205/0.0025**†	0.6035/0.0038	0.6151/0.0024	0.5980/0.0035
Tweet Stream - 20 classes	**0.0586/0.0009**†	0.0353/0.0024	0.0581/0.0005	0.0568/0.0008
KDDCUP99	**0.8410/0.0005**†	0.8207/0.0045	0.8396/0.0006	0.2592/0.0003
Poker-hand	0.3956/0.0089	0.3518/0.0076	0.3420/0.0016	**0.4021/0.0087**
UNSW-NB15	**0.8518/0.0001**†	0.8516/0.0001	0.8517/0.0000	0.8515/0.0001

* The best mean result is highlighted in boldface. If EEOF significantly outperforms at least two variants, or a variant performs significantly better than EEOF (Wilcoxon rank sum test at 95% confidence level), it is marked with a † symbol.

2.5 Summary

In this chapter, we propose a novel framework EEOF to address the class evolution problem and handle scenarios involving dominant emerging classes, which are not considered in the previous work but prevalent in real-world situations. Our proposed algorithm operates in an online learning manner with no need for storing past data. This framework is constructed in an ensemble of ensemble manner to enhance diversity within each OVA ensemble model and incorporates a novel adaptive model adaptation method that does not rely on any underlying assumptions about the data stream. Additionally, a confidence-triggered fallback mode is integrated to prevent performance degradation resulting from incorrect decisions regarding class disappearance. Comparative evaluations against the state-of-the-art method demonstrate that our algorithm achieves significantly better performance on both synthetic and real-world data streams, especially in scenarios featuring dominant emerging classes.

CHAPTER 3

SEMI-SUPERVISED ONLINE ENSEMBLE LEARNING FRAMEWORK

3.1 Background

Real-world data streams are continuously generated in large volumes and at high velocities [62]. It is unaffordable and impractical for experts to label every instance in data streams. To adapt classifiers with class evolution in dynamic data streams, a feasible solution involves that only a few instances of novel classes is labeled by experts, offering incomplete supervision information for model adaptation [95]. For example, labeling a few instances for a new trending topic on *Twitter* or a novel object that a robot has encountered in a new environment [24] would significantly aid in maintaining the classifiers. This scenario poses a challenge for learning with class evolution under incomplete supervision.

Existing semi-supervised algorithms on handling class evolution all necessitate the storage of past data. In ensemble-model approaches, such as ActMiner [60], SCDMiner

[35], ECHO [37] and ESCR [93], a new base classifier is always built with a recent bunch of incoming data and then incorporated into the ensemble model with existing base classifiers. In online learning scenarios, it is impractical to train a new base classifier for each incoming instance. In single-model approaches, many clustering methods, such as MINAS [17] and DISSFCM [12], are proposed to handle class evolution by incorporating micro-clusters for novel classes. Recently, a prototype-based semi-supervised learning framework [21] is proposed. This model maintains a dynamic set of prototypes which are extracted at multiple levels of granularity, and the prototypes for novel classes are extracted using the recent data chunk, achieving the state-of-the-art performance in handling evolving data streams with limited labels. However, these methods require the storage of past data to extract patterns of novel classes, which is impractical in the online learning scenarios.

Existing online learning algorithms are all supervised approaches [11, 38, 75]. In scenarios with incomplete supervision, these methods tend to overlook the abundance of the unlabeled instances and excessively focus on the labeled instances only. When novel classes emerge, labeled instances for these classes arrive. These methods adapt their classifier totally based on the exploitation on these novel classes, while neglect the unlabeled instances from other classes. This results in their suboptimal performance in multi-class classification tasks in real-world data streams.

To effectively address the semi-supervised scenarios, an ideal model should 1) leverage the abundant unlabeled instances in data streams and 2) balance the exploitations of instances between the emerging novel class and other existing classes. Correspondingly, in this chapter, we introduce a novel semi-supervised online ensemble learning framework (SOELF) to tackle class evolution under incomplete supervision in data streams. Unlike existing semi-supervised algorithms, SOELF processes instances in an online manner, eliminating the requirement for past data storage. First, we integrate a data

generation module and an inverse-prior-ratio model adaptation strategy to leverage the unlabeled instances in data streams for model adaptation. The data generation module, built on an ensemble online clustering model, generates diverse instances by exploiting current unlabeled instance, enhancing the model adaptation of the ensemble of ensemble framework in SOELF. This module also employs the decaying G-mean metrics to assign more reliable pseudo-labels to these generated instances. Second, the inverse-prior-ratio model adaptation strategy ensures balanced model adaptation by introducing more instances from existing classes when labeled instances of novel classes emerge, balancing the exploitation between classes. This technique mitigates the issue of overemphasis on novel classes. We conduct comprehensive experimental studies on both synthetic and real-world data streams under diverse incomplete supervision scenarios. The experimental results demonstrate that our method achieves higher accuracy compared to both existing online learning algorithms and the state-of-the-art semi-supervised algorithm that requires data storage.

3.2 Problem Formulation

In this section, a formal formulation for online learning with class evolution under incomplete supervision is introduced. Given a continuous data stream $\{\boldsymbol{x}_t\}_1^{+\infty}$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the feature vector of length d arriving at time t. The class label of $\boldsymbol{x}_t$ is y_t, which belongs to the class set $\mathscr{Y}_t$ at time t. Here, $\mathscr{Y}_t$ is the set of classes with positive underlying prior probabilities at time t, i.e. $\mathscr{Y}_t = \{c_i | p_t^{(i)} > 0\}$. Each instance $(\boldsymbol{x}_t, y_t)$ arrived at time t follows the distribution $p_t(\boldsymbol{x}, y)$ at time t, i.e. $(\boldsymbol{x}_t, y_t) \sim p_t(\boldsymbol{x}, y)$. In data streams with class evolution, the composition of $\mathscr{Y}_t$ evolves in three principle forms: 1) class emergence, 2) class disappearance, and 3) class reoccurrence. A class c_{novel} emerges at time t if $c_{novel} \in \mathscr{Y}_t$ and $c_{novel} \notin \bigcup_{i=1}^{t-1} \mathscr{Y}_i$. A class $c_{disapper}$ disappears at time t if $c_{disappear} \notin \mathscr{Y}_t$ and $c_{disappear} \in \mathscr{Y}_{t-1}$. A class $c_{reoccur}$ disappears at time d and reoccurs at time t if

$c_{reoccur} \in \mathscr{Y}_t$ and $c_{reoccur} \notin \bigcup_{i=d}^{t-1} \mathscr{Y}_i$ and $c_{reoccur} \in \mathscr{Y}_{d-1}$.

We introduce two hyperparameters to describe the incomplete supervision in data streams with class evolution. For a novel class, the expert labels the first q instances in the data stream for model adaptation. Mathematically, for $\boldsymbol{x}_t$, its true label y_t is available if $\sum_{i=1}^{t} \mathbb{1}[y_i == y_t] \leq q$. Additionally, we consider a more general situation where the expert continues labeling the data stream with a probability of θ for maintenance. Mathematically, if $\sum_{i=1}^{t} \mathbb{1}[y_i == y_t] > q$, the true label y_t is available with a probability of θ, where $0 \leq \theta < 1$. The available true label arrives after the prediction $\hat{y}_t$ is made before time $t+1$.

The task of online learning with class evolution under incomplete supervision is to construct an online classifier $\mathscr{F}$ that is capable of: 1) making an accurate prediction $\hat{y}_t$ for the arriving instance $\boldsymbol{x}_t$, and 2) updating according to $\boldsymbol{x}_t$ or $(\boldsymbol{x}_t, y_t)$, depending on the availability of y_t.

3.3 Proposed Algorithm

In this section, we first analyze issues of current online learning frameworks and then present components of SOELF on addressing these issues. An overview of SOELF is also provided.

3.3.1 Limitation in Existing Online Learning Algorithms

In the online learning framework EEOF [11], an ensemble of ensemble OVA framework is employed as the online classifier $\mathscr{F}$. An ensemble of OVA classifier $\mathscr{F}_i$ is constructed for each class c_i, denoted as $\mathscr{F} = \{\mathscr{F}_1, \mathscr{F}_2, ..., \mathscr{F}_n\}$, where n is the total number of classes that have appeared in the data stream. Each ensemble of OVA classifier $\mathscr{F}_i$ is composed of m base OVA classifiers $\mathscr{F}_{ij}$, where m is the ensemble size. Each base OVA classifier $\mathscr{F}_{ij}$ predicts whether an instance $\boldsymbol{x}_t$ belongs to a specific class c_i (designated as the positive

class, +1) or the remaining $n-1$ classes (designated as the negative class, -1). The prediction score for $\boldsymbol{x}_t$ belonging to class c_i is averaged across all base OVA classifiers in $\mathscr{F}_i$, and the final classification result $\hat{y}_t$ is assigned to the class with the highest prediction score:

$$\hat{y}_t = \arg\max_i \frac{1}{m} \sum_{j=1}^{m} \mathscr{F}_{ij}(\boldsymbol{x}_t). \tag{3.1}$$

Given an incoming instance $(\boldsymbol{x}_t, y_t)$, EEOF estimates the prior probability for each class and employs an adaptive cost-sensitive model adaptation method to balance the error feedback between classes. The time decayed method [78] is utilized to estimate the prior probability $p_t^{(i)}$ for class c_i at time t:

$$\hat{p}_t^{(i)} = \beta \hat{p}_{t-1}^{(i)} + (1-\beta)\mathbb{1}[y_t == c_i], \tag{3.2}$$

where $\hat{p}_t^{(i)}$ is the estimated prior probability, β is the time decay factor. For $\mathscr{F}_i(c_i = y_t)$, $\boldsymbol{x}_t$ belongs to the positive class and the loss weight r_+ is set to 1. For $\mathscr{F}_i(c_i \neq y_t)$, $\boldsymbol{x}_t$ belongs to the negative class and the loss weight r_- is set to $\hat{p}_t^{(i)}/(1-\hat{p}_t^{(i)})$. The Poisson distribution is employed to generate diverse loss weights for $\mathscr{F}_{ij}$ in $\mathscr{F}_i$.

While this model adaptation method appears promising, it is designed solely for labeled instances. In class evolution with incomplete supervision, when a novel class emerges, EEOF persists in learning from the labeled instances of this class while disregarding unlabeled instances and consequently accessing few or even no instances from other existing classes. Although the cost-sensitive model adaptation is intended to balance the error feedback between classes, the absence of instances from other classes makes this method inefficient. The neglect of unlabeled instances and the improper model adaptation strategy cause the classifier to overly focus on the novel class, resulting in a deterioration of multi-class classification performance. These issues are not unique for EEOF. Other online learning frameworks such as adapted OVA decision trees [38] and CBCE [75], also encounter these challenges.

3.3.2 Data Generation Module

In scenarios of class evolution under incomplete supervision, data streams consist of a limited number of labeled instances and a substantial amount of unlabeled instances. To exploit the abundant unlabeled instances, we maintain an additional ensemble online clustering model $\mathscr{C}$ alongside the ensemble of ensemble OVA classifiers $\mathscr{F}$ in SOELF for data generation. The ensemble online clustering model $\mathscr{C}$ consists of n base online clustering models for n appeared classes, i.e. $\mathscr{C} = \{\mathscr{C}_1, \mathscr{C}_2, ..., \mathscr{C}_n\}$, where $\mathscr{C}_i$ is the base online clustering model for class c_i with a set of cluster centers, i.e. $\mathscr{C}_i = \{\boldsymbol{center}_{ik}\}_k$. The number of cluster centers k varies across different base online clustering models. Many online clustering algorithms, such as CluStream [1], DenStream [10], and DBSTREAM [33], are capable of mainaining such ensemble online clustering model. Here, we select DenStream as the base online clustering model.

Given a labeled instances $(\boldsymbol{x}_t, y_t)$, the corresponding base online clustering model for class y_t is updated, and the online classifier $\mathscr{F}$ is updated in the aforementioned manner. Note that the ensemble online clustering model $\mathscr{C}$ is also capable of classification by calculating the distances between $\boldsymbol{x}_t$ and the cluster centers. The final prediction is the class with the closest cluster center to $\boldsymbol{x}_t$:

$$\hat{y}_{t,\mathscr{C}} = \arg\max_i \frac{1}{\min_k(Dist(\boldsymbol{x}_t, \boldsymbol{center}_{ik}))}. \tag{3.3}$$

The classification performance of $\mathscr{F}$ and $\mathscr{C}$ on the data stream can be partially estimated using the labeled instances. We employ the decaying G-mean $G_{t,\mathscr{M}}$ to monitor the performance of both models:

$$G_{t,\mathscr{M}} = \sqrt[n]{\prod_{i=1}^{n} R_{i,t,\mathscr{M}}}, \tag{3.4}$$

$$R_{i,t,\mathscr{M}} = \begin{cases} R_{i,t-1,\mathscr{M}}, & \text{if } y_t \neq c_i \\ \beta R_{i,t-1,\mathscr{M}} + (1-\beta)\mathbb{1}[y_t == \hat{y}_{t,\mathscr{M}}], & \text{if } y_t = c_i, \end{cases} \tag{3.5}$$

where $R_{i,t,\mathscr{M}}$ is the decaying recall [78] of the model $\mathscr{M} \in \{\mathscr{F}, \mathscr{C}\}$ for class c_i at time t and $\hat{y}_{t,\mathscr{F}}$ is the prediction of the online classifier $\mathscr{F}$, i.e. $\hat{y}_t = \hat{y}_{t,\mathscr{F}}$.

Given an unlabeled instance $\boldsymbol{x}_t$, a set of diverse samples with pseudo-labels $\{(\boldsymbol{s}_{t,ij}, \hat{y}_{t,gen})\}$ is generated for the adaptation of the online classifier $\mathscr{F} = \{\mathscr{F}_{ij}\}$. The selection of the pseudo-label $\hat{y}_{t,gen}$ is based on the estimated classification performance of $\mathscr{F}$ and $\mathscr{C}$. The prediction from the model with the higher decaying G-mean is more likely to be chosen as the pseudo-label:

$$\hat{y}_{t,gen} = \begin{cases} \hat{y}_{t,\mathscr{F}} & \text{with a probability of } prob_{\mathscr{F}} \\ \hat{y}_{t,\mathscr{C}} & \text{with a probability of } prob_{\mathscr{C}}, \end{cases} \tag{3.6}$$

where $prob_{\mathscr{M}} = G_{t,\mathscr{M}}/(G_{t,\mathscr{F}} + G_{t,\mathscr{C}}), \mathscr{M} \in \{\mathscr{F}, \mathscr{C}\}$.

For each base OVA classifier $\mathscr{F}_{ij}$, the sample $\boldsymbol{s}_{t,ij}$ is generated by the weighted summation of random interpolations between $\boldsymbol{x}_t$ and cluster centers in the base online clustering model $\mathscr{C}_h$ for the class $\hat{y}_{t,gen}$, i.e. $c_h = \hat{y}_{t,gen}$:

$$\boldsymbol{s}_{t,ij} = \sum_{k=1}^{|\mathscr{C}_h|} \rho_k \boldsymbol{s}_{t,ij}^{(k)}, \tag{3.7}$$

where $\boldsymbol{s}_{t,ij}^{(k)}$ is the random interpolation between $\boldsymbol{x}_t$ and the k-th cluster center in $\mathscr{C}_h$, and ρ_k is the normalized weight calculated based on the distance between $\boldsymbol{x}_t$ and the k-th cluster center:

$$\boldsymbol{s}_{t,ij}^{(k)} = (\boldsymbol{x}_t + \alpha_{ij}^{(k)} * (\boldsymbol{center}_{hk} - \boldsymbol{x}_t)), \tag{3.8}$$

$$\rho_k = \frac{1/Dist(\boldsymbol{x}_t, \boldsymbol{center}_{hk})}{\sum_{a=1}^{|\mathscr{C}_h|} 1/Dist(\boldsymbol{x}_t, \boldsymbol{center}_{ha})}. \tag{3.9}$$

Here, the random variable $\alpha_{ij}^{(k)}$ follows a uniform distribution on the interval $[0,1]$. Interpolations from closer cluster centers will have higher weights. Detailed pseudo-code is displayed in Algorithm 3.1. Such a set of generated samples with the pseudo-label selected based on decaying G-mean, introduces diversity and enhances the performance of the online classifier $\mathscr{F}$ in an ensemble of ensemble OVA framework.

Algorithm 3.1 Generation

Input:

- Instance $\boldsymbol{x}_t$
- Ensemble of ensemble OVA classifier model $\mathscr{F} = \{\mathscr{F}_i\}$
- Ensemble online clustering model $\mathscr{C} = \{\mathscr{C}_i\}$
- $G_{t,\mathscr{F}}$, decaying G-mean of $\mathscr{F}$
- $G_{t,\mathscr{C}}$, decaying G-mean of $\mathscr{C}$
- $\hat{y}_{t,\mathscr{F}}$, predictions of $\mathscr{F}$
- $\hat{y}_{t,\mathscr{C}}$, predictions of $\mathscr{C}$

Output:

- The set of generated samples $\mathbb{S}_t = \{(\boldsymbol{s}_{t,ij}, \hat{y}_{t,gen})\}$
- Class index h with $c_h = \hat{y}_{t,gen}$

$\mathbb{S}_t \leftarrow \varnothing$
// Select pseudo-label based on the decaying G-mean
$prob_{\mathscr{F}} \leftarrow G_{t,\mathscr{F}}/(G_{t,\mathscr{F}} + G_{t,\mathscr{C}})$
$prob_{\mathscr{C}} \leftarrow G_{t,\mathscr{C}}/(G_{t,\mathscr{F}} + G_{t,\mathscr{C}})$
Select a value as $\hat{y}_{t,gen}$ from $\{\hat{y}_{t,\mathscr{F}}, \hat{y}_{t,\mathscr{C}}\}$ with probabilities of $prob_{\mathscr{F}}$ and $prob_{\mathscr{C}}$ respectively.
// Generate features of these samples
$h \leftarrow$Find$(i|c_i = \hat{y}_{t,gen})$
for $i \leftarrow 1 : n$ **do**
  for $j \leftarrow 1 : m$ **do**
    Calculate ρ_k for $k \in [1, |\mathscr{C}_h|]$
    Calculate $\boldsymbol{s}_{t,ij}^{(k)}$ for $k \in [1, |\mathscr{C}_h|]$
    $\boldsymbol{s}_{t,ij} \leftarrow \sum_{k=1}^{|\mathscr{C}_h|} \rho_k * \boldsymbol{s}_{t,ij}^{(k)}$
    $\mathbb{S}_t \leftarrow \mathbb{S}_t \cup \{(\boldsymbol{s}_{t,ij}, \hat{y}_{t,gen})\}$
  end for
end for

3.3.3 Inverse-Prior-Ratio Model Adaptation Strategy

As we have analyzed above, the model adaptation methods in previous online learning frameworks [11, 38, 75] are inadequate to tackle with scenarios of class evolution under incomplete supervision. When novel classes emerge with labeled instances, these algorithms tend to excessively focus on the novel classes, resulting in a deterioration of the multi-class classification performance. Remarkably, the performance of these methods may even deteriorate further as the number of labeled instances for novel classes increases, as evidenced by subsequent experimental results.

To address this issue, we design the inverse-prior-ratio model adaptation strategy to introduce more instances from existing classes into the model adaptation process. As the labeled instances of a novel class c_i arrive, the estimated prior probability $\hat{p}_t^{(i)}$ for this class increases and the estimated prior probabilities $\hat{p}_t^{(j)}(j \neq i)$ for other classes decrease. To prevent the online classifier from excessively focusing on the class c_i, the set of generated samples of class c_i should be used to adapt the online classifier $\mathscr{F}$ with lower probability. Conversely, the sets of generated samples of other classes $c_j(j \neq i)$ should be used with higher probabilities. For an unlabeled instance arrives at time t, suppose the pseudo-label of the generated samples is class c_h, i.e. $c_h = \hat{y}_{t,gen}$, the adaptation probability is describe as:

$$prob_{adapt} = e^{-K*|\mathscr{F}|*\hat{p}_{t-1}^{(h)}}, \tag{3.10}$$

where $|\mathscr{F}|$ represents the number of classes that have appeared in the data stream, K is the hyperparameter defining the adaptation probability in the ideal scenario where all appeared classes have an equal prior probability, and $\hat{p}_{t-1}^{(h)}$ is the estimated prior probability of the class c_h at time $t-1$. The negative sign in the exponent of e ensures an inverse relationship between $prob_{adapt}$ and the estimated prior probability $\hat{p}_{t-1}^{(h)}$, and the incorporation of $|\mathscr{F}|$ is used to counteract the decrease in prior probability resulting from the increasing number of classes. If these generated samples with pseudo-label

$\hat{y}_{t,gen}$ is used for model adaptation, the estimated prior probability for class c_i is then updated as follows:

$$\hat{p}_t^{(i)} = \beta \hat{p}_{t-1}^{(i)} + (1-\beta)\mathbb{1}[\hat{y}_{t,gen} == c_i]. \tag{3.11}$$

3.3.4 Overview

An overview of SOELF in data streams is provided in Algorithm 3.2. Given an unlabeled instance, the data generation module generates a set of diverse samples (line 21), and the inverse-prior-ratio model adaptation strategy is employed (line 22~23). The model adaptation process using the set of generated samples is similar to the model adaptation process using a single labeled instance, as displayed in Algorithm 3.3.

3.4 Experimental Study

In this section, we present a comprehensive experimental study demonstrating the effectiveness of SOELF compared with state-of-the-art methods on a variety of synthetic and real-world data streams under diverse incomplete supervision scenarios. Additionally, we conduct an ablation study to highlight the significance of different components in SOELF.

3.4.1 Experimental Settings

1) Data Stream

Class evolution encompasses class emergence, class disappearance, and class reoccurrence. We utilize three tabular datasets, namely Letter Recognition[1] (16 attributes), Statlog (Landsat Satellite)[2] (36 attributes), and Covertype[3] (54 attributes), as well as an

[1] https://archive.ics.uci.edu/dataset/59/letter+recognition
[2] https://archive.ics.uci.edu/dataset/146/statlog+landsat+satellite
[3] https://archive.ics.uci.edu/dataset/31/covertype

Algorithm 3.2 An Overview of SOELF in data stream

Initialize $\mathscr{F}, \mathscr{C}, \hat{\boldsymbol{p}}_t, G_{t,\mathscr{F}}, G_{t,\mathscr{C}}$ from an offline training dataset.
while Instance $\boldsymbol{x}_t$ arrives **do**
  // Model Prediction
  $\hat{y}_t \leftarrow \mathscr{F}(\boldsymbol{x}_t)$
  // Model Adaptation
  $\hat{y}_{t,\mathscr{F}} \leftarrow \hat{y}_t$
  $\hat{y}_{t,\mathscr{C}} \leftarrow \mathscr{C}(\boldsymbol{x}_t)$
  if true label y_t is available **then**
    Update $\hat{\boldsymbol{p}}_t$ with the occurrence of class y_t
    // If a novel class emerges
    if no $\mathscr{F}_i$ in $\mathscr{F}$ is available for y_t **then**
      Initialize $\mathscr{F}_{novel}$ with m base classifier for class y_t
      Initialize $\mathscr{C}_{novel}$ for class y_t
      $\mathscr{F} \leftarrow \mathscr{F} \cup \mathscr{F}_{novel}$
      $\mathscr{C} \leftarrow \mathscr{C} \cup \mathscr{C}_{novel}$
    end if
    Update $G_{t,\mathscr{F}}$ and $G_{t,\mathscr{C}}$ with y_t, $\hat{y}_{t,\mathscr{F}}$, and $\hat{y}_{t,\mathscr{C}}$
    $\mathscr{F} \leftarrow$ Update$(\boldsymbol{x}_t, y_t, \mathscr{F}, \hat{\boldsymbol{p}}_t)$ // *Update algorithm in EEOF*
    Update $\mathscr{C}$ with $(\boldsymbol{x}_t, y_t)$
  else
    $(\mathbb{S}_t, h) \leftarrow Generation(\boldsymbol{x}_t, \mathscr{F}, \mathscr{C}, G_{t,\mathscr{F}}, G_{t,\mathscr{C}}, \hat{y}_{t,\mathscr{F}}, \hat{y}_{t,\mathscr{C}})$
    $prob_{adapt} \leftarrow e^{-K*|\mathscr{F}|*\hat{p}_t^{(h)}}$
    if $Uniform(0,1) \leq prob_{adapt}$ **then**
      Update $\hat{\boldsymbol{p}}_t$ with the occurrence of class c_h
      $\mathscr{F} \leftarrow Update-Set(\mathbb{S}_t, \mathscr{F}, \hat{\boldsymbol{p}}_t)$
    end if
  end if
end while

image dataset, MNIST[4] (784 attributes), for generating synthetic data streams on class evolution scenarios of 1) class emergence and 2) class disappearance and recoccurrence under diverse cases of incomplete supervision. This aims to demonstrate that SOELF provides more accurate predictions in handling issues encountered by previous methods during the class emergence, while performing comparably in other class evolution scenarios. In the class emergence scenarios, we select two classes as existing classes and one class as the emerging class. The prior probability curves of these classes are

[4]http://yann.lecun.com/exdb/mnist/

Algorithm 3.3 Update-Set

Input:

- Generated Set $\mathbb{S}_t = \{(\boldsymbol{s}_{t,ij}, \hat{y}_{t,gen})\}$
- Ensemble of ensemble OVA classifier model $\mathscr{F} = \{\mathscr{F}_i\}$
- Estimated prior probability vector $\hat{\boldsymbol{p}}_t = [\hat{p}_t^{(i)}]$

Output: Updated $\mathscr{F}$.

for $i \leftarrow 1 : n$ **do**
  for $j \leftarrow 1 : m$ **do**
    $\hat{y}_{t,ij} \leftarrow \mathscr{F}_{ij}(\boldsymbol{s}_{t,ij})$
    if $\hat{y}_{t,gen} == c_i$ **then**
      $r_+ \sim Poisson(1)$
      if $r_+ > 0$ **then**
        Update $\mathscr{F}_{ij}$ with $r_+ * Loss(+1, \hat{y}_{t,ij})$
      end if
    else
      $r_- \sim Poisson(\hat{p}_t^i/(1-\hat{p}_t^i))$
      if $r_- > 0$ **then**
        Update $\mathscr{F}_{ij}$ with $r_- * Loss(-1, \hat{y}_{t,ij})$
      end if
    end if
  end for
end for

illustrated in Figure 3.1(a). In the scenarios of class disappearance and reoccurrence, we select three classes as existing class, with one class disappearing for a while and then reoccurring latter, as illustrated in Figure 3.1(b).

We further investigate SOELF on six real-world data streams that exhibit different types of class evolution. These data streams include four preprocessed data streams of the TwitterCrawl data stream [50], namely Tweet Stream - A/B/C/20 classes in [75], Poker-hand [13], and KDDCUP99 [76]. The key characteristics of these real-world data streams are displayed in Table 2.1.

Besides selecting various data streams involving class evolution, we thoroughly examine the performance of SOELF under diverse scenarios of incomplete supervision. The number of labeled instances for a novel class, i.e. q, is chosen from $\{10, 100\}$. The

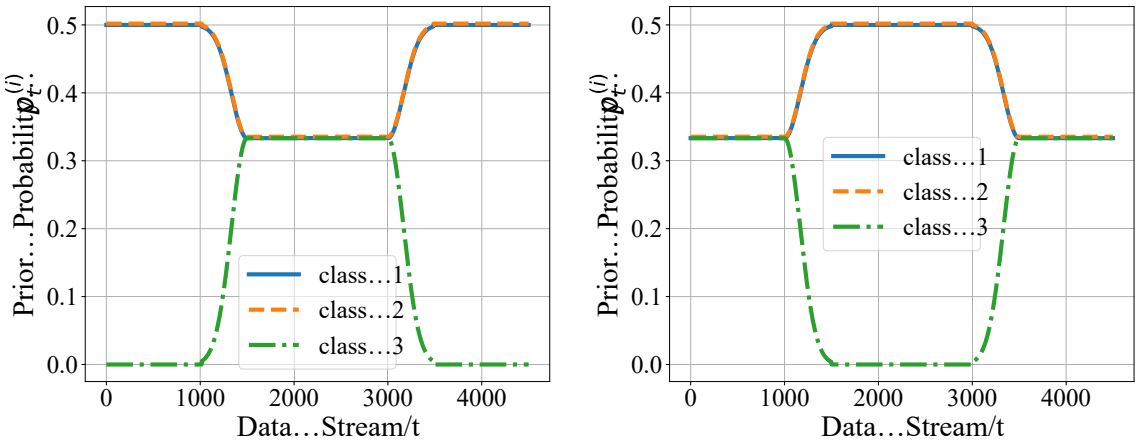


Figure 3.1: The prior probability curves of classes in synthetic data streams with (a) class emergence and (b) class disappearance and reoccurrence.

probability of labeling instances in the data stream, i.e. θ, is chosen from $\{0, 0.05, 0.10\}$, where 0 means no instances in the data stream are labeled except for the first q instances for a novel class. These settings provide six incomplete supervision scenarios for each data stream mentioned above.

2) Comparison Algorithms

Five algorithms are chosen for comparison to examine SOELF:

- CBCE [75] - A popular supervised online learning framework on handling class evolution in data streams.
- EEOF [11] - The state-of-the-art supervised online learning framework on handling class evolution in data streams.
- CBCE-v/EEOF-v - These two algorithms are variants of CBCE and EEOF. In CBCE/EEOF, a class is excluded from the classification process if its estimated prior probability falls below a threshold. This strategy is misleading in incomplete supervision scenarios, where a class may be considered disappeared simply because

the model has not received labeled data for this class in a long time. For a fair comparison, we compare our method with the variants of CBCE and EEOF, i.e. CBCE-v and EEOF-v, where no classes are considered disappeared.

- PT [21] - The state-of-the-art semi-supervised algorithm on handling class evolution with limited labels. This algorithm processes data chunk-by-chunk and extracts prototypes at multiple levels of granularity. In [21], this algorithm does not have a short name. For convenience, we refer to this algorithm as PT (prototype) in the following sections.

These supervised online learning algorithms, CBCE, EEOF, CBCE-v, and EEOF-v, are adapt to the incomplete supervision scenarios by utilizing only labeled instances for model adaptation. The semi-supervised algorithm PT operates by first making predictions on a data chunk and then updating with the chunk data. This framework includes novel class detection to request true labels and model adaptation. In this study, we focus solely on the model adaptation part. To adapt this method to our scenarios, the novel class detection module is disregarded, and available labeled instances are provided for model adaptation.

3) Parameter Settings

The same size of the initial labeled training dataset is set to 1000 for all algorithms before processing on data streams. In CBCE/EEOF/CBCE-v/EEOF-v/SOELF, the same base OVA classifier, Kernelized Logistic Regression (KLR) [47], is employed with the same hyperparmeters for a fair comparison. The hyperparameters of KLR are tuned on the initial training dataset using methods suggested in previous works [11, 75]. Specifically, a 5-folder cross-validation is employed to determine the the learning rate η and the regularization parameter λ. The kernel width σ is set to the 5-th percentile of the pairwise distances between all instances. The tuned hyperparameters for data streams

Table 3.1: Tuned Hyperparameters of KLR for Data Streams

Data Stream	η	λ	σ
Letter (Synthetic)	9	10^{-4}	5.29
Statlog (Synthetic)	3	10^{-4}	32.7
Covertype (Synthetic)	1	10^{-4}	4.00
MNIST (Synthetic)	3	10^{-4}	4.61
Tweet Stream - A	5	3×10^{-4}	1.24
Tweet Stream - B	3	10^{-4}	1.25
Tweet Stream - C	8	10^{-4}	1.24
Tweet Stream - 20 classes	2	10^{-4}	1.32
Poker-hand	8	10^{-3}	3.37
KDDCUP99	10^{-4}	10^{-4}	0.30

are listed in Table 3.1. The time decaying factor β is set to 0.9. The ensemble size is set to 10 for EEOF/EEOF-v/SOELF that maintain ensemble of ensemble frameworks. The hyperparameters for DenStream are configured as follows: the decaying factor $\beta_{DenStream}$ is set to 0.01, the stream speed v is set to 1 to process instances one-by-one, the number of instances for initialization is set to 4, ϵ that defines the ϵ neighborhood is set to the averaged value of the μ-th distances of instances during initialization, and other hyperparameters, including μ, are set to default values. The hyperparameter K in the inverse-prior-ratio model adaptation strategy in SOELF is set to 3.3. For PT, the chunk size is set to 1000, the same as the size of initial training dataset. Other hyperparameters not mentioned in this section are set to their default values as suggested in their corresponding papers.

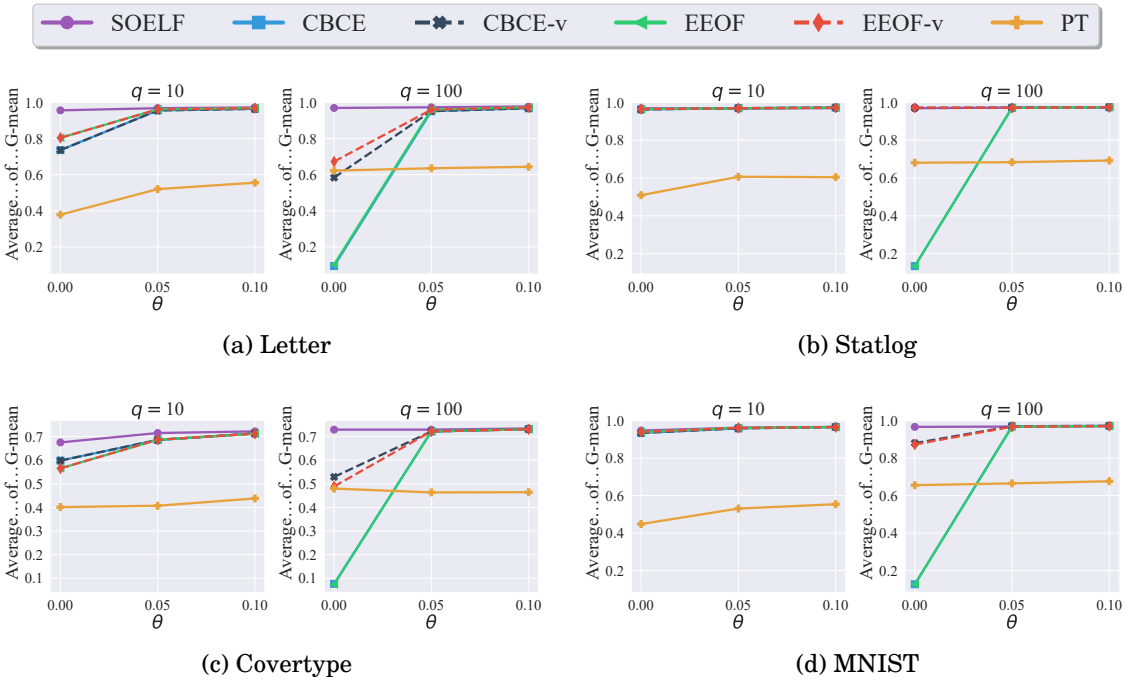
4) Evaluation Metrics

To evaluate algorithms on handling class evolution with incomplete supervison, we utilze the same evaluation metrics to monitor the performance of multi-class classification along the data stream, as described in Eq. (2.6). We perform 10 independent runs for each algorithm on each data stream to mitigate random influences. The significance of the difference between algorithms is examined using the Wilcoxon rank sum test [29].

3.4.2 Experimental Analysis

1) Performance on Synthetic Data Streams

The performance of our method SOELF and the other compared methods on synthetic data streams with class evolution under various scenarios of incomplete supervision is displayed in Figure 3.2 and Figure 3.2. The performance of the online learning frameworks, CBCE and EEOF, drops as the number of labeled instance for novel classes q increases from 10 to 100 for scenarios with $\theta = 0$. We attribute this primarily to two reasons: 1) the misjudgement of disappeared classes, and 2) the model adaptation of the classifier excessively focusing on the novel classes. CBCE-v and EEOF-v are variants where no existing classes are considered disappeared, and they perform better than the original versions in scenarios with $q = 100$ and $\theta = 0$. However, a decline in performance is still observed as q increases, as shown in Figure 3.2(a) and Figure 3.2(c)(d), primarily due to the second reason. The semi-supervised method PT processes the data stream chunk-by-chunk, requiring a sufficient collection of data for adapting the classifier to novel classes. As a result, PT cannot update and be ready to classify new class data immediately upon the emergence of a new class, as online learning models do. Consequently, its performance falls behind the online learning models almost in all cases. Our proposed method, SOELF, adheres to an online learning framework and addresses the issues encountered in existing online learning frameworks. In the scenarios with $\theta = 0$, contrary to previous online learning models, the performance of SOELF remains stable and even improves with increasing q, which is attributed to the inverse-prior-ratio model adaptation strategy. In addition, the performance of SOELF is considerably stable as θ, the probability of labeling instances in the data stream, changes. The superiority of SOELF is particularly pronounced when $\theta = 0$. We also attribute this to the exploitation on unlabeled instances of the data generation module in SOELF. In summary, SOELF is capable of effectively managing various scenarios of incomplete supervision regarding



(a) Letter

(b) Statlog

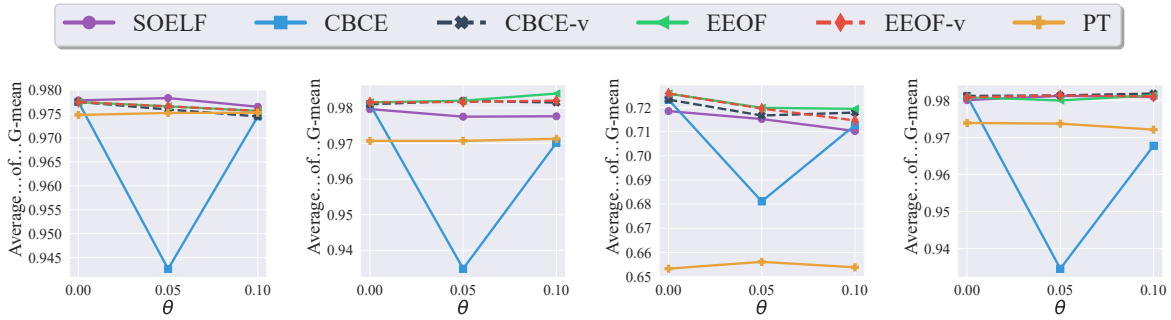
(c) Covertype

(d) MNIST

Figure 3.2: Plots of the average of G-mean results for the algorithms on synthetic data streams of (a) Letter, (b) Statlog, (c) Covertype and (d) MNIST with class emergence under various scenarios of incomplete supervision. For the comparison purpose, the y-axis limits of the plots for each data stream are set to the same scale across different choices of q.

class emergence, and outperforms other methods.

We further conduct an experimental study on synthetic data streams with class disappearance and reoccurrence, as shown in Figure 3.3. Since there are no novel classes in this form of class evolution, the parameter q is not considered. Only the parameter θ varies to generate diverse scenarios of incomplete supervision. Our method, SOELF, is specifically designed to manage the model adaptation of an online classifier on class emergence, rather than class disappearance and reoccurrence, which lie outside the targeted scenarios of SOELF. Consequently, SOELF performs comparably in cases such as Letter and MNIST, exhibiting a slight decrease in performance (less than 0.01 in the average of G-mean) compared to the best performance achieved by other algorithms in cases such as Statlog and Covertype.

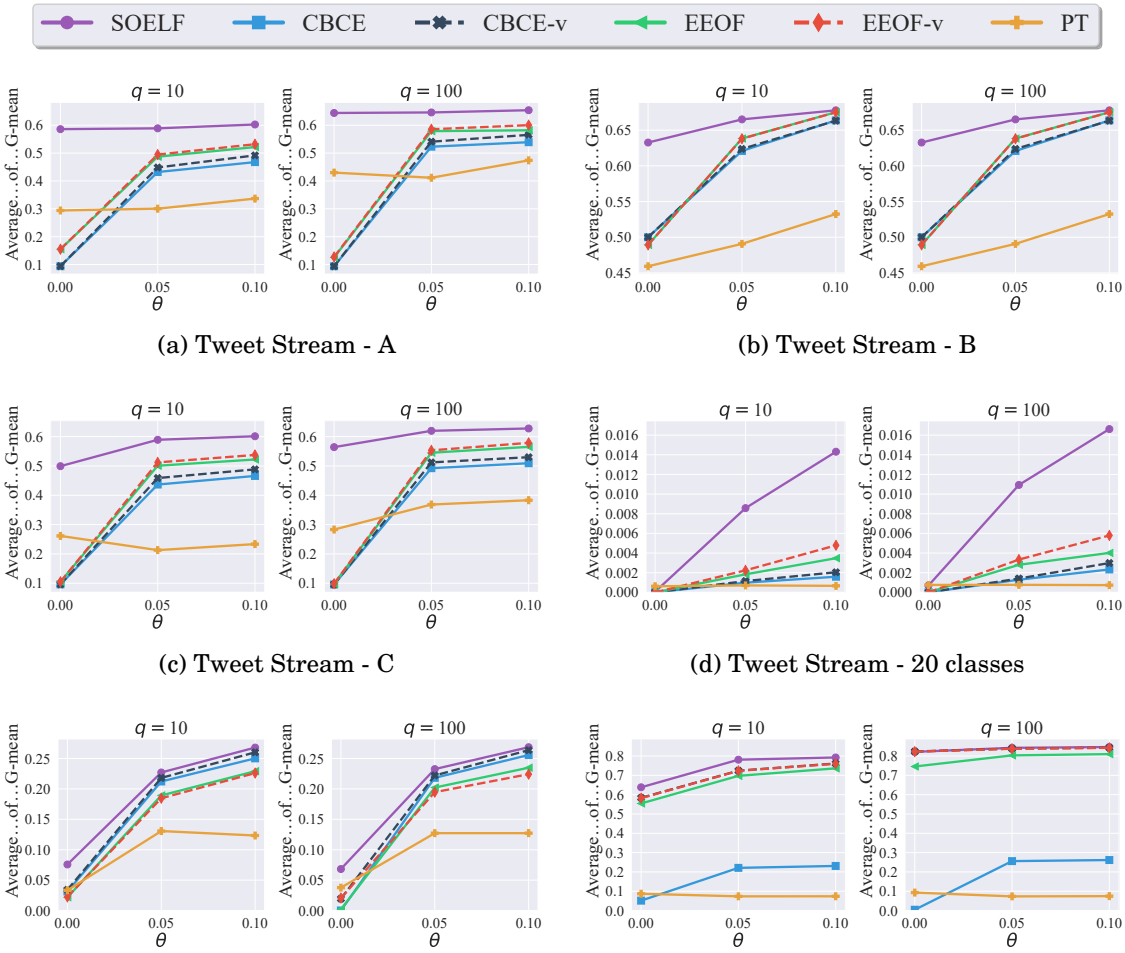


(a) Letter (b) Statlog (c) Covertype (d) MNIST

Figure 3.3: Plots of the average of G-mean results for the algorithms on synthetic data streams with class disappearance and reoccurrence under various scenarios of incomplete supervision.

2) Performance on Real-world Data Streams

The performance of SOELF and other methods on real-world data streams is depicted in Figure 3.4. SOELF demonstrates superior performance across nearly all scenarios of class evolution under incomplete supervision, except for the Tweet Stream - 20 classes with $\theta = 0$, where all algorithms fail to yield satisfactory predictions. Real-world data streams involve concept drift and the emergence of novel classes. Although SOELF shows comparable or slightly worse performance on synthetic data streams with class disappearance and reoccurrence, in Tweet Stream - B, where the same form of class evolution exists and concept drift is involved, SOELF still exhibits significant improvements due to its continuous exploitation on unlabeled instances to adapt classifier to concept drift, as shown in Figure 3.4(b). Previous online learning frameworks and their variants, namely CBCE, EEOF, CBCE-v, and EEOF-v, exhibit unstable performance with varying θ, and their performance decreases with an increase in q in several cases, such as Tweet Stream - A and Poker-hand with $\theta = 0$, as shown in Figure 3.4(a)(e). SOELF produces more stable and more accurate predictions in these data streams. For KDDCUP99, SOELF shows its privilege particularly when q is lower, i.e. $q = 10$. While PT performs stably in all cases, this method still suffers from the delay of model adaptation on novel classes. All these experimental results on real-world data streams featuring ample novel classes, substan-



(a) Tweet Stream - A

(b) Tweet Stream - B

(c) Tweet Stream - C

(d) Tweet Stream - 20 classes

(e) Poker-hand

(f) KDDCUP99

Figure 3.4: Plots of the average of G-mean results for the algorithms on six real-world data streams with class evolution under various scenarios of incomplete supervision. For the comparison purpose, the y-axis limits of the plots for each data stream are set to the same scale across different choices of q.

tial data volume, and high-dimensional data, effectively showcase the efficacy of our proposed framework SOELF in addressing class evolution under incomplete supervision.

3) Ablation Study

Three variants of SOELF is designed to assess the significance of components in SOELF:

- SOELF-GenF: In this variant, the original unlabeled instance $\boldsymbol{x}_t$ is assigned with

a pseudo-label $\hat{y}_{t,\mathscr{F}}$ predicted by the online classifier $\mathscr{F}$ to examine the sufficiency of the data generation module.

- SOELF-GenO: In this variant, the original unlabeled instance $\boldsymbol{x}_t$ is assigned with a pseudo-label $\hat{y}_{t,\mathscr{C}}$ predicted by the ensemble online clustering model $\mathscr{C}$ to examine the sufficiency of the data generation module.
- SOELF-Inverse: A common approach would be to adapt the online classifier $\mathscr{F}$ using generated samples on all unlabeled instances. We set $prob_{adapt} = 1$ in this variant to examine the sufficiency of the inverse-prior-ratio model adaptation strategy.

Synthetic data streams with class emergence and real-world data streams under various scenarios of incomplete supervision are selected for examination, as displayed in Table 3.2. The win/equal/loss counts the number of cases where these variants significantly outperform, show no significantly difference, or perform significantly worse, compared with SOELF respectively. Upon comparing SOELF-GenF, SOELF-GenO with SOELF, it is noteworthy that although SOELF-GenF and SOELF-GenO outperform SOELF slightly in 3 and 1 cases respectively, SOELF consistenly provides significantly more accurate predictions in 23 and 31 cases, particularly evident in data streams such as Tweet Stream - A/B/C/20 classes. These improvements are attribute to the diverse generated samples with reliable pseudo-labels facilitated by the integration of the decaying G-mean metric. Upon comparing SOELF-Inverse with SOELF, SOELF exhibits slightly worse performance in four cases, but significantly outperforms in 25 cases, particularly in data streams such as Letter, Statlog, Poker-hand, and KDDCUP99. Unlike adapting online classifier on every unlabeled instance in SOELF-Inverse, the inverse-prior-ratio model adaptation strategy in SOELF ensures a balanced exploitation of unlabeled instances across classes on model adaptation, thereby preventing the online classifier from excessively focusing on any particular class. Therefore, the default framework of

SOELF is generally recommended for addressing diverse scenarios of class evolution under incomplete supervision.

Table 3.2: The average of G-mean results (mean/std) averaged over 10 runs of the ablation study on synthetic data streams with class evolution and real-world data streams under various scenarios of incomplete supervision.

Data Stream	q	θ	**SOELF**	**SOELF-GenF**	**SOELF-GenO**	**SOELF-Inverse**
Letter	10	0.00	**0.9561/0.0057**	0.9187/0.0388 -	0.9414/0.0187 -	0.9450/0.0068 -
Letter	10	0.10	**0.9720/0.0020**	0.9660/0.0050 -	0.9693/0.0035 -	0.9650/0.0049 -
Letter	100	0.00	**0.9693/0.0095**	0.9664/0.0060 =	0.9396/0.0117 -	0.9439/0.0060 -
Letter	100	0.10	**0.9776/0.0018**	0.9743/0.0035 =	0.9709/0.0053 -	0.9674/0.0048 -
Statlog	10	0.00	**0.9692/0.0024**	0.9671/0.0063 =	0.9503/0.0130 -	0.9291/0.0066 -
Statlog	10	0.10	0.9696/0.0022	**0.9706/0.0020 =**	0.9596/0.0046 -	0.9534/0.0035 -
Statlog	100	0.00	0.9698/0.0018	**0.9708/0.0016 =**	0.9426/0.0163 -	0.9184/0.0040 -
Statlog	100	0.10	0.9731/0.0013	**0.9742/0.0014 =**	0.9619/0.0027 -	0.9546/0.0049 -
Covertype	10	0.00	**0.6751/0.0140**	0.5796/0.0507 -	0.6590/0.0181 =	0.6677/0.0052 =
Covertype	10	0.10	**0.7218/0.0041**	0.7099/0.0079 -	0.7139/0.0052 -	0.7086/0.0045 -
Covertype	100	0.00	**0.7292/0.0043**	0.7039/0.0127 -	0.7153/0.0094 -	0.6824/0.0076 -
Covertype	100	0.10	**0.7341/0.0038**	0.7263/0.0065 -	0.7280/0.0070 =	0.7140/0.0045 -
MNIST	10	0.00	0.9473/0.0072	0.9447/0.0064 =	0.9475/0.0060 =	**0.9631/0.0019 +**
MNIST	10	0.10	0.9666/0.0027	**0.9679/0.0023 =**	0.9668/0.0028 =	0.9658/0.0015 =
MNIST	100	0.00	0.9669/0.0023	**0.9691/0.0017 +**	0.9640/0.0033 -	0.9621/0.0016 -

MNIST	100	0.10	0.9708/0.0021	**0.9723/0.0028 =**	0.9695/0.0020 =	0.9648/0.0014 -
Tweet Stream - A	10	0.00	**0.5859/0.0101**	0.3963/0.0618 -	0.5467/0.0116 -	0.5620/0.0127 -
Tweet Stream - A	10	0.10	**0.6026/0.0073**	0.5570/0.0207 -	0.5918/0.0075 -	0.5880/0.0075 -
Tweet Stream - A	100	0.00	**0.6440/0.0132**	0.5057/0.0336 -	0.6259/0.0055 -	0.6439/0.0078 =
Tweet Stream - A	100	0.10	0.6539/0.0044	0.6026/0.0145 -	0.6453/0.0062 -	**0.6557/0.0034 =**
Tweet Stream - B	-	0.00	0.6326/0.0081	0.5688/0.0241 -	0.6163/0.0056 -	**0.6378/0.0053 =**
Tweet Stream - B	-	0.10	0.6777/0.0082	0.6730/0.0082 =	0.6762/0.0052 =	**0.6919/0.0060 +**
Tweet Stream - C	10	0.00	0.4992/0.0120	0.3390/0.0209 -	0.4764/0.0133 -	**0.5077/0.0039 =**
Tweet Stream - C	10	0.10	**0.6012/0.0039**	0.5378/0.0156 -	0.5742/0.0107 -	0.5891/0.0058 -
Tweet Stream - C	100	0.00	0.5641/0.0036	0.4109/0.0139 -	0.5459/0.0071 -	**0.5705/0.0049 +**
Tweet Stream - C	100	0.10	**0.6278/0.0053**	0.5795/0.0075 -	0.6154/0.0042 -	0.6256/0.0045 =
Tweet Stream - 20 classes	10	0.00	**0.0001/0.0000**	0.0000/0.0000 -	0.0000/0.0000 -	0.0000/0.0000 -
Tweet Stream - 20 classes	10	0.10	0.0143/0.0014	0.0087/0.0012 -	0.0119/0.0010 -	**0.0155/0.0007 +**
Tweet Stream - 20 classes	100	0.00	**0.0007/0.0003**	0.0000/0.0000 -	0.0001/0.0001 -	0.0004/0.0001 -
Tweet Stream - 20 classes	100	0.10	0.0166/0.0009	0.0114/0.0014 -	0.0141/0.0013 -	**0.0167/0.0008 =**
Poker-hand	10	0.00	0.0758/0.0082	**0.0905/0.0217 =**	0.0397/0.0027 -	0.0374/0.0041 -
Poker-hand	10	0.10	**0.2680/0.0021**	0.1444/0.0086 -	0.2084/0.0066 -	0.2023/0.0011 -
Poker-hand	100	0.00	0.0683/0.0082	**0.0720/0.0454 =**	0.0138/0.0012 -	0.0127/0.0017 -

Poker-hand	100	0.10	**0.2686/0.0028**	0.1497/0.0102 -	0.2055/0.0041 -	0.2033/0.0023 -
KDDCUP99	10	0.00	0.6381/0.0109	0.5966/0.0066 -	**0.6510/0.0098 +**	0.5863/0.0125 -
KDDCUP99	10	0.10	**0.7925/0.0027**	0.7829/0.0027 -	0.7776/0.0027 -	0.7905/0.0035 =
KDDCUP99	100	0.00	0.8215/0.0014	**0.8237/0.0021 +**	0.8070/0.0029 -	0.6770/0.0078 -
KDDCUP99	100	0.10	0.8464/0.0020	**0.8542/0.0008 +**	0.8125/0.0018 -	0.8350/0.0011 -
Win/Tie/Loss			-	3/12/23	1/6/31	4/9/25

* The highest mean results are highlighted in bold. The symbols +/=/- indicate that the variant performs significantly better, or shows no significantly difference, or performs significantly worse, compared with SOELF respectively (Wilcoxon rank sum test at 95% confidence level).

3.5 Summary

In this chapter, we propose a novel semi-supervised online ensemble learning framework, SOELF, to address class evolution under incomplete supervision. To the best of our knowledge, this is the first study to delve into online learning with class evolution under incomplete supervision. Unlike existing semi-supervised approaches, our algorithm operates in an online manner without the need to store past data. SOELF integrates a data generation module based on an ensemble online clustering model and the decaying G-mean metric, to exploit the unlabeled instances and generate diverse samples with reliable pseudo-labels for model adaptation. Additionally, an inverse-prior-ratio model adaptation strategy is introduced to prevent the online classifier from excessively focusing on novel classes with few labeled instances, which is encountered in existing online learning frameworks. Comparative evaluations against the state-of-the-art methods demonstrate that our algorithm SOELF significantly outperforms existing approaches on both synthetic and real-world data streams under various scenarios of incomplete supervision.

CHAPTER 4

ENSEMBLE OF PASSIVE-AGGRESSIVE MODEL

4.1 Background

In real-world data streams, both sets of features and classes are evolving with time, posing the problem of feature evolution [23, 32, 59] and class evolution [11, 75] simultaneously. For example, in a social network like *Twitter*, novel topics emerge while outdated topics are forgotten. Some forgotten topics such as the Olympic Games, may regain popularity. Similarly, the vocabulary in each tweet evolves over time, with new words being coined and older terms gradually falling out of use. In this scenario, data instances are distributed in open data space. An example of data distributed in open data space is illustrated in Figure 4.1. Learning in open data space has been investigated in several early studies [58, 59]. However, these algorithms necessitate substantial data storage for model adaptation, processing data streams in a chunk-by-chunk manner. Following researches focus on either developing online learning algorithms with class evolution [11, 75], or constructing online learning algorithms with feature evolution [42, 89]. To the best of our knowledge, no existing work is capable of addressing both feature evolution

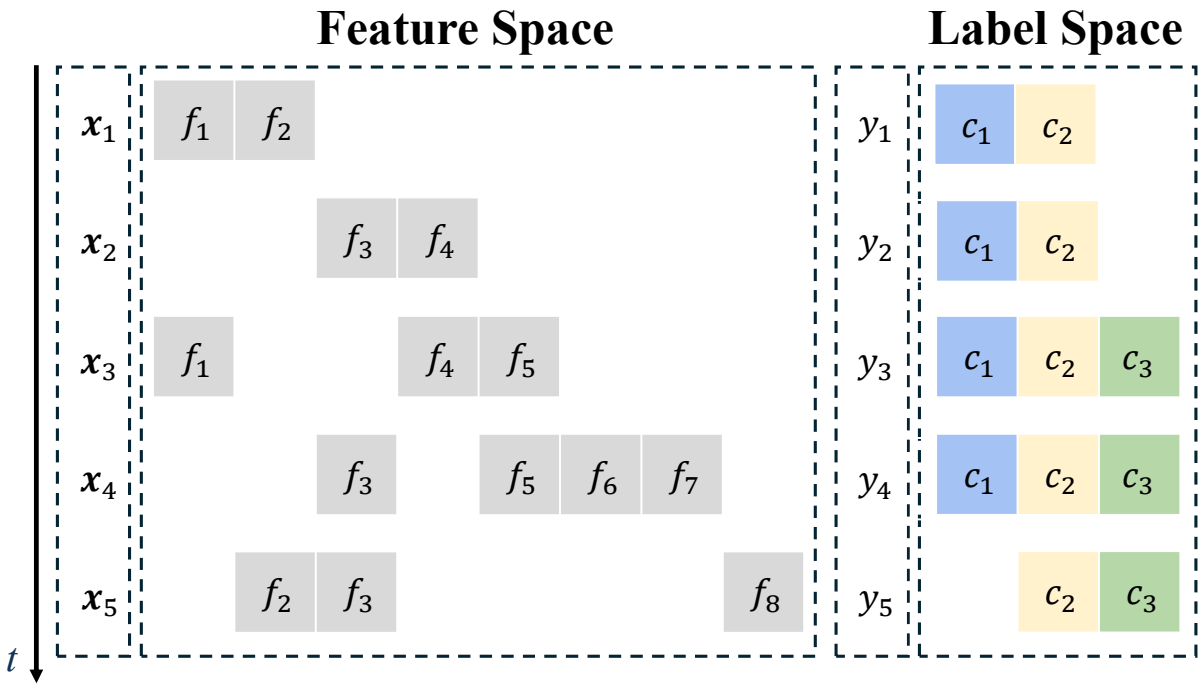


Figure 4.1: An illustrative example of a data stream distributed in open data space. At time $t = 3$, within the feature space of $\boldsymbol{x}_3$, feature f_5 emerges while feature f_3 disappears. In the label space of y_3, the binary classification between classes c_1 and c_2 transitions to a three-class classification with the emergence of class c_3. By time $t = 5$, feature f_8 emerges, features f_5, f_6 and f_7 disappear, and previously disappeared feature f_2 reoccurs. Class c_1 disappears from the label space.

and class evolution simultaneously in an online learning manner, which means free of data storage. A more general online learning algorithm which makes no fixed-set assumption on neither the feature space nor the label space is necessary, as shown in Figure 4.2.

In this chapter, to initiate the research on this open problem, we begin our investigations by constructing several baseline models based on the state-of-the-art approaches in both fields. Specifically, the ensemble of ensemble OVA framework [11] for class evolution, and the passive-aggressive (PA) model [85] for feature evolution. The constructed baseline ensembles of PA models, where PA models are configured to function as OVA classifiers, seem to provide viable solutions. We then go beyond merely integrating state-of-the-art approaches by analyzing their limitations. Although PA methods are

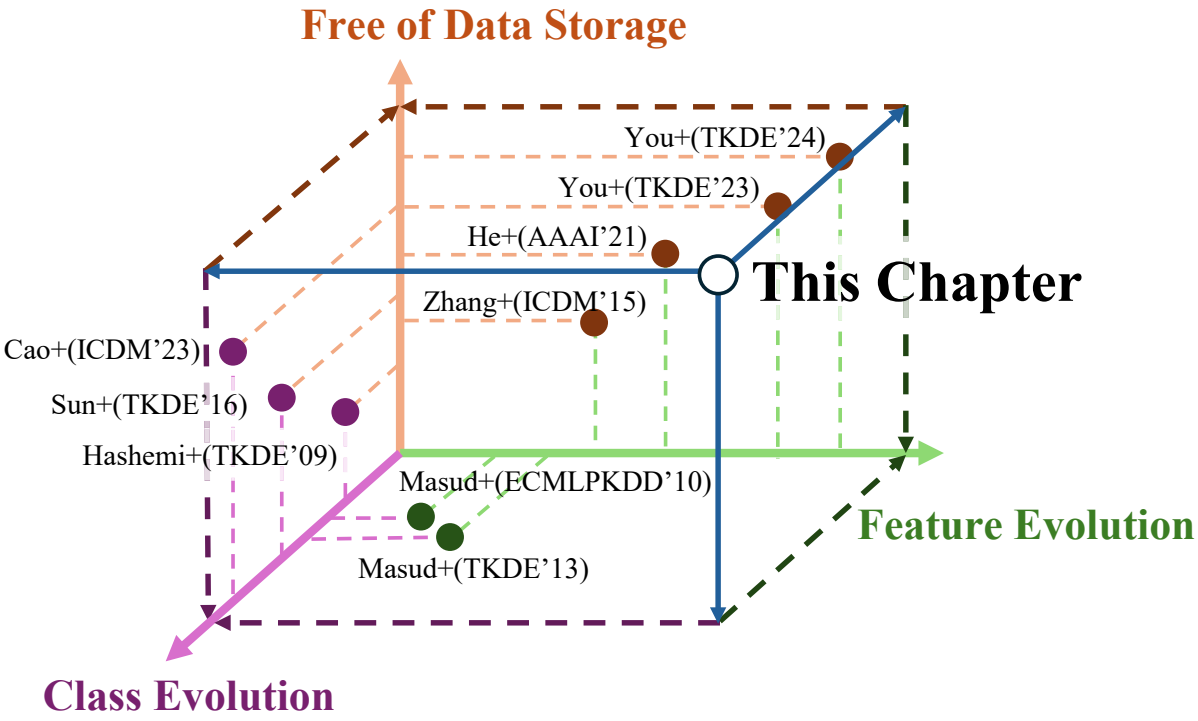


Figure 4.2: The projected positions of the papers along the axes are utilized solely to illustrate the relative order of their publication years. In this chapter, we introduce a novel model, EPAM, that is capable of handling class evolution and feature evolution while being free of data storage. Papers included in this figure are Masud et al. [59, 60], Hashemi et al. [38], Zhang et al. [89], He et al. [42], You et al. [85, 86], Sun et al. [75], and Cao et al. [11].

ubiquitous in solving online learning with feature evolution [39], the bias term of the linear model in these PA models is excluded from feature evolution, either by incorporating it into the weight vector as a dummy input attribute or simply assuming it to be 0. In other words, the bias term in these models is a fixed value for diverse features spaces and the accommodation of the bias term to the dynamic data space is generally ignored in previous work. In addition, previous online learning algorithms for handling class evolution primarily address the imbalance between the positive and negative classes in each OVA classifier [11, 75]. However, the negative class is not a singular entity but consists of multiple distinct classes. The imbalance of the error feedback among these classes is overlooked.

To address these limitations, a novel ensemble of passive-aggressive model (EPAM)

is proposed. This model manages feature evolution and class evolution using only a single available instance each time, eliminating the need to store any historical data. In EPAM, a novel feature contributed bias classifier (FCBC) serves as the base OVA classifier, with its bias term adapting dynamically to diverse feature spaces formed by varying feature combinations in the open data space, therefore delivering a more accurate decision boundary. Furthermore, a novel model adaptation strategy is designed to balance error feedback among classes that belong to the negative class of each FCBC, enhancing the multi-class classification performance of EPAM in open data space as the compositions of classes evolve. Comprehensive experimental studies on both synthetic and real-world data streams demonstrate the superiority of EPAM, and highlight the significance of the proposed components in managing data streams with open data space.

4.2 Problem Formulation

In this section, we present a formal formulation for online learning in open data space. Given a continuous data stream $\{(\boldsymbol{x}_t, y_t)\}_1^{+\infty}$, where $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$ is a feature vector of length d_t, and $y_t \in \mathscr{Y}_t$ is the associated class label. Here, $\mathscr{X}_t$ denotes the feature space at time t with $\mathscr{X}_t = \{f_i | f_i \text{ occurred at time } t\}$ and $|\mathscr{X}_t| = d_t$. $\mathscr{Y}_t$ denotes the label space at time t. Each element in $\mathscr{Y}_t$ is the class c_i with a positive prior probability at time t, i.e. $\mathscr{Y}_t = \{c_i | p_t^{(i)} > 0\}$. Each instance $(\boldsymbol{x}_t, y_t)$ arrived at time t follows the distribution $p_t(\boldsymbol{x}, y)$ at time t, i.e. $(\boldsymbol{x}_t, y_t) \sim p_t(\boldsymbol{x}, y)$. In the online learning, only one instance $\boldsymbol{x}_t$ is available at time t. The true label y_t of $\boldsymbol{x}_t$ always arrives after the prediction $\hat{y}_t$ is made before time $t+1$.

In the data stream with open data space, both feature space $\mathscr{X}_t$ and label space $\mathscr{Y}_t$ change over time, referred to as feature evolution and class evolution respectively. Taking class evolution as an example, based on how classes evolve, class evolution is categorized into three principal forms: 1) class emergence, 2) class disappearance, and 3) class

reoccurrence. A class c_{novel} emerges at timestamp t if $c_{novel} \in \mathcal{Y}_t$ and $c_{novel} \notin \bigcup_{i=1}^{t-1} \mathcal{Y}_i$. A class $c_{disppear}$ disappears at timestamp t if $c_{disppear} \notin \mathcal{Y}_t$ and $c_{disppear} \in \mathcal{Y}_{t-1}$. A class $c_{reoccur}$ disappears at timestamp d and reoccurs at timestamp t if $c_{reoccur} \in \mathcal{Y}_t$ and $c_{reoccur} \notin \bigcup_{i=d}^{t-1} \mathcal{Y}_i$ and $c_{reoccur} \in \mathcal{Y}_{d-1}$. The forms of feature evolution can also be categorized in a similar way.

The task of online learning in open data space is to construct an online classifier $\mathcal{F} : \mathcal{X}_t \rightarrow \mathcal{Y}_t$ that is capable of 1) making accurate prediction $\hat{y}_t$ for arriving instance $\boldsymbol{x}_t$ at time t, and 2) updating after receiving the true label y_t.

4.3 Baseline Ensemble of Passive-Aggressive Models

In this section, we first provide a brief overview of the general formulations in current research addressing class evolution and feature evolution. We then construct baseline models for online learning in open data space based on state-of-the-art online learning algorithms in class evolution and feature evolution. Finally, we analyze the limitations of existing approaches in managing data streams with open data space.

4.3.1 Preliminary

The prevailing approach of existing online learning algorithms in handling class evolution involves maintaining an ensemble of one-versus-all (OVA) classifiers [3, 69], which transform the n-class classification into n binary classifications. Each OVA classifier $\mathcal{F}_i$ in the ensemble predicts whether an incoming instance $\boldsymbol{x}_t$ belongs to class c_i (considered as the positive class $+1$ for $\mathcal{F}_i$) or not (considered as the negative class -1 for $\mathcal{F}_i$). The final prediction is classified to the class with the hightest prediction score:

$$\hat{y}_t = \arg\max_i \mathcal{F}_i(\boldsymbol{x}_t). \tag{4.1}$$

A new OVA classifier is initialized and added to the ensemble when a novel class emerges. The OVA classifier corresponding to any disappeared class is excluded from the classification process in Eq. (4.1). When a previously disappeared class reoccurs, its corresponding OVA classifier is reintroduced for classification [75].

The prevalent approaches of existing online learning algorithms in solving feature evolution are Passive-Aggressive (PA) models. Here, we present a general formulation of PA models as outlined in [39], using the notations in our study. Let $\boldsymbol{w}_t \in \bigcup_{i=1}^{t-1} \mathscr{X}_i \subseteq \mathbb{R}^{u_{t-1}}$ denotes the weight vector of the classifier at time t before update, where u_{t-1} represents the number of all features observed up to time $t-1$. At time t, the instance $\boldsymbol{x}_t \in \mathscr{X}_t$ arrives. The feature space of $\boldsymbol{x}_t$ can be categorized into three parts: common, disappeared, and novel feature spaces. In the common feature space, features are both observed previously and in $\boldsymbol{x}_t$, i.e. $f_c \in \mathscr{X}_t \cap (\bigcup_{i=1}^{t-1} \mathscr{X}_i)$. In the disappeared feature space, features are observed previously and absent in $\boldsymbol{x}_t$, i.e. $f_d \in (\bigcup_{i=1}^{t-1} \mathscr{X}_i) \setminus \mathscr{X}_t$. In novel feature space, features are all novel ones, i.e. $f_n \in \mathscr{X}_t \setminus (\bigcup_{i=1}^{t-1} \mathscr{X}_i)$. Let the superscripts "c", "d" and "n" denote the projection of a vector onto the common, disappeared, and novel feature spaces, respectively. The objective of PA models takes the form:

$$
\begin{aligned}
\boldsymbol{w}_{t+1} = \underset{\boldsymbol{w}=[\boldsymbol{w}^d, \boldsymbol{w}^c, \boldsymbol{w}^n]}{\arg\min} \quad & d(\boldsymbol{w}, \boldsymbol{w}_t) + \gamma ||\boldsymbol{w}^n||_p + C\xi \\
s.t. \quad & l(\boldsymbol{w}; (\boldsymbol{x}_t, y_t)) \leq \xi \\
& \xi \geq 0,
\end{aligned}
\tag{4.2}
$$

where $d(\cdot, \cdot)$ is the distance function to measure the update of the weight vector on previously seen features. The l_p-norm serves as a regularization term for the weight vector in the novel feature space. ξ is a non-negative slack variable that facilitates a soft-margin boundary [84]. $l(\boldsymbol{w}; (\boldsymbol{x}_t, y_t))$ represents the loss function of a classifier with the weight vector $\boldsymbol{w}$ when making the prediction on the instance $(\boldsymbol{x}_t, y_t)$. γ and C are trade-off parameters to control the influence of different terms in the objective function.

Generally, the hinge loss is the default choice for the loss function. The prediction is made in the common feature space, i.e. $\hat{y}_t = \text{sign}(\boldsymbol{w}_t^c \cdot \boldsymbol{x}_t^c)$.

4.3.2 Baseline Models

Note that PA models are binary classifiers. To develop online learning algorithms that manage data streams with open data space, a straight forward approach is to assemble PA models as base OVA classifiers. Following the state-of-the-art method in online learning with class evolution, EEOF [11], we adapt different PA models within the ensemble of ensemble framework, thereby constructing a group of baseline ensembles of PA models. In each baseline model, an ensemble of PA models is maintained as the OVA classifiers for each class, namely, $\mathscr{F}_i = \{\mathscr{F}_{i1}, \mathscr{F}_{i2}, ..., \mathscr{F}_{im}\}$. Here, m is the ensemble size. The prediction score for class c_i is the average prediction score across all PA models in $\mathscr{F}_i$, i.e. $\mathscr{F}_i(\boldsymbol{x}_t) = \frac{1}{m}\sum_{j=1}^{m}\mathscr{F}_{ij}(\boldsymbol{x}_t)$. The final prediction result $\hat{y}_t$ is determined as:

$$\hat{y}_t = \arg\max_i \frac{1}{m}\sum_{j=1}^{m}\mathscr{F}_{ij}(\boldsymbol{x}_t) \tag{4.3}$$

$$\mathscr{F}_{ij}(\boldsymbol{x}_t) = \boldsymbol{w}_{ij,t}^c \cdot \boldsymbol{x}_t^c, \tag{4.4}$$

where $\boldsymbol{w}_{ij,t}^c$ represents the projection of the weight vector $\boldsymbol{w}_{ij,t}$ for the j-th PA model corresponding to class c_i in the common feature space. In addition to the strategy of excluding disappeared classes and re-including reoccurred classes in the prediction process [75], the confidence-triggered fallback mode is also integrated, ensuring that disappeared classes are re-included when PA models for existing classes fail to provide reliable predictions. Details of the confidence-triggered fallback mode are omitted here due to page limitations and can be found in [11].

Class evolution leads to the dynamic changes of prior probabilities of classes. For each PA model in a n-class classification problem, one class is considered as the positive class while the remaining $n-1$ classes are considered as the negative class. Here, we

introduce the loss weight parameter $r_{ij,t}$ to modulate the loss on the arriving instance $(\boldsymbol{x}_t, y_t)$, thereby balancing the error feedback between classes for each PA model $\mathscr{F}_{ij}$. After integrating the PA model into the ensemble of ensemble framework and taking the loss weight into the consideration, the objective of the PA model $\mathscr{F}_{ij}$ takes the form:

$$\begin{aligned} \boldsymbol{w}_{ij,t+1} = & \underset{\boldsymbol{w}=[\boldsymbol{w}^d,\boldsymbol{w}^c,\boldsymbol{w}^n]}{\arg\min} \; d(\boldsymbol{w},\boldsymbol{w}_{ij,t}) + \gamma||\boldsymbol{w}^n||_p + C\xi \\ s.t. \quad & r_{ij,t} l(\boldsymbol{w};(\boldsymbol{x}_t, y_{i,t})) \leq \xi \\ & \xi \geq 0, \end{aligned} \tag{4.5}$$

The loss function is defined as:

$$l(\boldsymbol{w};(\boldsymbol{x}_t, y_{i,t})) = \max\{0, 1 - y_{i,t}(\boldsymbol{w}_{\parallel \boldsymbol{x}_t} \cdot \boldsymbol{x}_t)\}, \tag{4.6}$$

where $\boldsymbol{w}_{\parallel \boldsymbol{x}_t}$ represents the projection of the vector $\boldsymbol{w}$ onto the feature space of $\boldsymbol{x}_t$. The value of $y_{i,t}$ is determined by:

$$y_{i,t} = \begin{cases} +1, & y_t = c_i \\ -1, & y_t \neq c_i. \end{cases} \tag{4.7}$$

In EEOF [11], the prior probability $p_t^{(i)}$ of each class c_i at time t is estimated using the time decayed method [78, 79]:

$$\hat{p}_t^{(i)} = \beta \hat{p}_{t-1}^{(i)} + (1-\beta)\mathbb{1}[y_t == c_i], \tag{4.8}$$

where $\hat{p}_t^{(i)}$ is the estimated prior probability and β is the time decay factor. The estimated prior probability is then utilized to balance the error feedback between the positive and negative classes. Specifically, the loss weight $r_{ij,t}$ for the PA model $\mathscr{F}_{ij}$ is generated in the form:

$$r_{ij,t} \sim \begin{cases} Poisson(1), & y_t = c_i \\ Poisson(\hat{p}_t^{(i)}/(1-\hat{p}_t^{(i)})), & y_t \neq c_i. \end{cases} \tag{4.9}$$

The state-of-the-art method in online learning with feature evolution is OLI^{2}DS [85], which is developed based on the PA model and introduces the cumulative class ratio to address the class imbalance problem in binary classification. Furthermore, it employs the concept of informativeness to assign varying update weights to different features. OL$_{SF}$ [89] is the benchmark PA model in online learning with feature evolution. By plugging different implementations of these PA models into the objective function (4.5), we can construct several baseline models. Specifically, these baseline ensembles of PA models are:

- EEOF-OLI^{2}DS$_1$: Plugging OLI^{2}DS into the objective function (4.5) with the loss weight $r_{ij,t}$ to be a multiple of the cumulative class ratio, as defined in the OLI^{2}DS [85] model:

$$r_{ij,t} = \begin{cases} \delta * n^-_{ij,t}/(n^+_{ij,t} + n^-_{ij,t}), & y_t = c_i \\ \\ \delta * n^+_{ij,t}/(n^+_{ij,t} + n^-_{ij,t}), & y_t \neq c_i, \end{cases} \tag{4.10}$$

  where δ is the scaling factor, $n^+_{ij,t}$ and $n^-_{ij,t}$ are counts of instances from the positive and negative classes of $\mathscr{F}_{ij}$ until time t, respectively.

- EEOF-OLI^{2}DS$_2$: Same as EEOF-OLI^{2}DS$_1$ *with the exception* that the loss weight $r_{ij,t}$ is determined by the time-decayed dynamic class ratio in EEOF [11], as described in Eq. (4.8) and Eq. (4.9).

- EEOF-OL$_{SF}$: Plugging OL$_{SF}$ into the objective function (4.5), with the loss weight $r_{ij,t}$ determined by the time-decayed dynamic class ratio in EEOF [11], as described in Eq. (4.8) and Eq. (4.9).

4.3.3 Limitation Analysis

While the aforementioned construction strategy appears sound and the resulting baseline models are capable of handling online learning in open data space, several limitations still remain when addressing real-world data streams. In current PA models [6, 7, 31, 32, 39, 85, 86, 89, 90], the bias term is excluded from feature evolution, either by incorporating it into the weight vector by inventing a dummy input attribute $x_{t,0} = 1$ or simply assuming it to be 0. In other words, the bias term in these models is the same value or simply 0 for diverse feature spaces formed by different possible combinations of features. Consider a binary classification data stream in which classes are linearly separable with every single feature. In this scenario, we cannot guarantee that a zero or a fixed bias is appropriate for any linear model on any single feature. Moreover, even if an optimal bias can be identified for each linear model on individual features, the found bias is still unlikely to be perfectly suitable for a linear model operating in a feature space formed of any combination of these features. In real-world data stream with open data space, the number of possible feature spaces is uncountable and the feature evolution is unpredictable. The bias terms in PA models are required to be adaptable to the varying feature spaces for constructing more accurate decision boundary.

Additionally, current online learning algorithms for class evolution [11, 75] primarily focus on the imbalance between the positive and negative classes in each OVA classifier $\mathscr{F}_i$. However, the negative class is not a singular entity but encompasses multiple classes. Previous work applies the same sampling ratio [75] or loss weight [11] to all instances from different classes designated as the negative class, potentially assuming that classes belonging to the negative class are evenly distributed, which is overly idealistic in real-world data streams with class evolution. For a class c_k with a considerably small prior probability $p_t^{(k)}$, it serves as the negative class for the OVA classifier $\mathscr{F}_i$ with $i \neq k$. Generally, the number of instances in the positive class is smaller than that in the

negative class (i.e., the remaining classes). The error feedback of instances from the class c_k for $\mathscr{F}_i$ diminishes with the other classes belonging to the negative class due to the balancing techniques, leading to $\mathscr{F}_i$ to produce higher prediction scores for instances of c_k, which ultimately degrades the overall multi-class classification performance.

4.4 Proposed Algorithm: EPAM

In light of aforementioned limitations, we propose the Ensemble of Passive-Aggressive Model (EPAM) to tackle the problem of online learning in open data space. In this section, we will introduce the details of EPAM.

4.4.1 Feature Contributed Bias Classifier (FCBC)

As analyzed in the previous section, the bias term is required be capable of accommodating different feature spaces in the data stream with feature evolution. We begin with an example to motivate our proposed model. Consider a binary classification data stream distributed in a two-dimensional feature space $\{f_1, f_2\}$, which is linearly separable along either f_1 or f_2. In this case, we will obtain two classifiers, $\hat{y}_t = sign(w_1 x_1 + b_1)$ and $\hat{y}_t = sign(w_2 x_2 + b_2)$, perfectly perform the classification tasks on individual features f_1 and f_2, where x_1 and x_2 are the feature values along f_1 and f_2 respectively. To classify an instance with available values for both features, the classifier ought to be formed as $\hat{y}_t = sign(w_1 x_1 + w_2 x_2 + b_1 + b_2)$, where the bias term is composed of biases of the classifiers on f_1 and f_2. In general, the bias term can be viewed as the composition of biases from classifiers operating on different individual features within the current feature space.

In EPAM, the feature contributed bias classifier (FCBC) is designed as the base OVA classifier $\mathscr{F}_{ij}$ with an adaptive bias vector $\boldsymbol{b}_{ij,t}$ capable of accommodating to diverse feature spaces, offering more accurate decision boundary for classification. For every

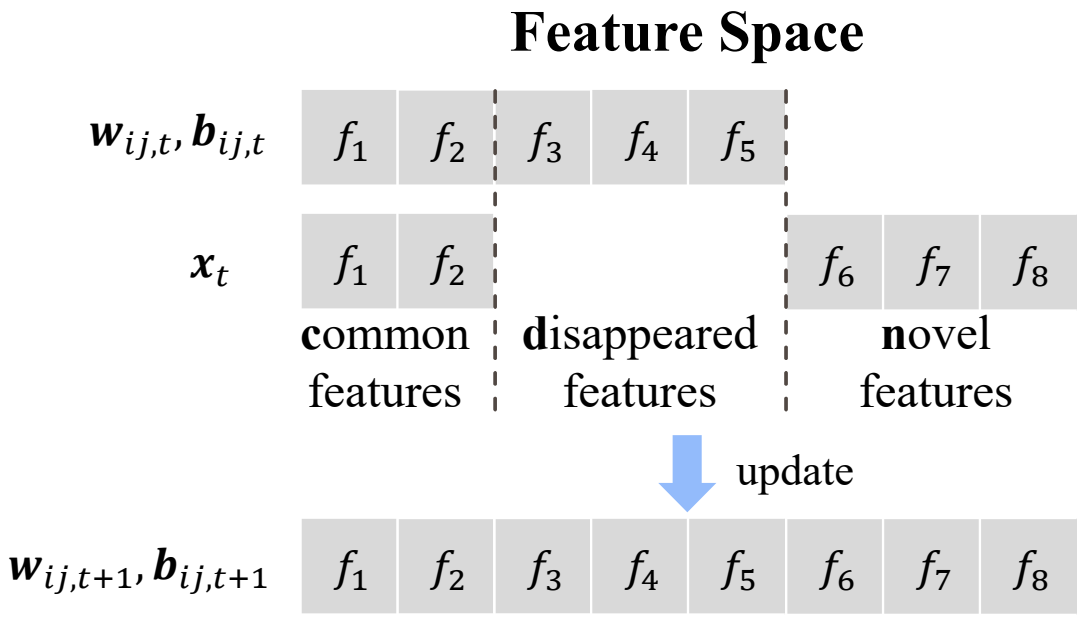


Figure 4.3: Before the update, $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ contains values only on the common feature space $\{f_1, f_2\}$ and the disappeared feature space $\{f_3, f_4, f_5\}$ compared with the feature space of $\boldsymbol{x}_t$. After the update, $\boldsymbol{w}_{ij,t+1}$ and $\boldsymbol{b}_{ij,t+1}$ are extended to include values on the novel feature space $\{f_6, f_7, f_8\}$.

observed feature, there is a corresponding value in $\boldsymbol{b}_{ij,t}$ which has the same shape as $\boldsymbol{w}_{ij,t}$, i.e. $\boldsymbol{b}_{ij,t} \in \bigcup_{i=1}^{t-1} \mathcal{X}_i \subseteq \mathbb{R}^{u_{t-1}}$, where u_{t-1} represents the number of all features observed up to time $t-1$. For an arriving instance $\boldsymbol{x}_t$, the weight vector and the bias vector can be divided into projections on the common feature space and the disappeared feature space, i.e. $\boldsymbol{w}_{ij,t} = [\boldsymbol{w}^d_{ij,t}, \boldsymbol{w}^c_{ij,t}]$ and $\boldsymbol{b}_{ij,t} = [\boldsymbol{b}^d_{ij,t}, \boldsymbol{b}^c_{ij,t}]$ (an example illustrated in Fig. 4.3). The prediction of FCBC is calculated on the common feature space:

$$\mathscr{F}_{ij}(\boldsymbol{x}_t) = \boldsymbol{w}^c_{ij,t} \cdot \boldsymbol{x}^c_t + \boldsymbol{b}^c_{ij,t} \cdot \mathbf{1}^c, \tag{4.11}$$

where $\mathbf{1}^c$ is the vector filled with ones in the common feature space. By employing the square of the euclidean distance as the $d(\cdot,\cdot)$ function along with $\gamma = 1$ into the objective function (4.5) and incorporating the update of the bias vector, the objective of FCBC $\mathscr{F}_{ij}$

is then designed in the following form:

$$
\begin{aligned}
(4.12) \qquad \boldsymbol{w}_{ij,t+1}, \boldsymbol{b}_{ij,t+1} = & \underset{\substack{\boldsymbol{w}=[\boldsymbol{w}^d, \boldsymbol{w}^c, \boldsymbol{w}^n] \\ \boldsymbol{b}=[\boldsymbol{b}^d, \boldsymbol{b}^c, \boldsymbol{b}^n]}}{\arg\min} \frac{1}{2}(\|\boldsymbol{w}^d - \boldsymbol{w}^d_{ij,t}\|^2 + \|\boldsymbol{w}^c - \boldsymbol{w}^c_{ij,t}\|^2 + \|\boldsymbol{w}^n\|^2) \\
& + \frac{A}{2}(\|\boldsymbol{b}^d - \boldsymbol{b}^d_{ij,t}\|^2 + \|\boldsymbol{b}^c - \boldsymbol{b}^c_{ij,t}\|^2 + \|\boldsymbol{b}^n\|^2) + C\xi \\
s.t. \quad & r_{ij,t} l_{i,t}(\boldsymbol{w}^c, \boldsymbol{w}^n, \boldsymbol{b}^c, \boldsymbol{b}^n) \leq \xi \\
& \xi \geq 0,
\end{aligned}
$$

where $r_{ij,t}$ is the loss weight, ξ is the slack variable, $A > 0$ adjusts the update trade-off between the weight vector and the bias vector, and $C > 0$ controls the trade-off between rigidness and slackness. The loss function is defined as:

$$
\begin{aligned}
(4.13) \qquad l_{i,t}(\boldsymbol{w}^c, \boldsymbol{w}^n, \boldsymbol{b}^c, \boldsymbol{b}^n) &= l(\boldsymbol{w}, \boldsymbol{b}; (\boldsymbol{x}_t, y_{i,t})) \\
&= \max\{0, 1 - y_{i,t}(\boldsymbol{w}^c \cdot \boldsymbol{x}^c_t + \boldsymbol{w}^n \cdot \boldsymbol{x}^n_t + \boldsymbol{b}^c \cdot \mathbf{1}^c + \boldsymbol{b}^n \cdot \mathbf{1}^n)\},
\end{aligned}
$$

where $y_{i,t}$ is the assigned label defined as in Eq. (4.7).

The objective function (4.12) of FCBC aims to achieve two goals: 1) to retain the information learned previously by minimizing the differences between the current and previous classifiers in terms of the weight and bias vectors in the disappeared and common feature spaces, and 2) to extend the classifier to the novel feature space (an example illustrated in Fig. 4.3) by minimizing its corresponding regularization terms, i.e. $\|\boldsymbol{w}^n\|^2$ and $\|\boldsymbol{b}^n\|^2$. To solve the objective function (4.12), we employ the Lagrangian function and the Karush-Kuhn-Tucker (KKT) conditions [9], and obtain:

$$
\begin{aligned}
(4.14) \qquad \mathscr{L}(\boldsymbol{w}, \boldsymbol{b}, \xi, \tau, \eta) = & \frac{1}{2}(\|\boldsymbol{w}^d - \boldsymbol{w}^d_{ij,t}\|^2 + \|\boldsymbol{w}^c - \boldsymbol{w}^c_{ij,t}\|^2 + \|\boldsymbol{w}^n\|^2) \\
& + \frac{A}{2}(\|\boldsymbol{b}^d - \boldsymbol{b}^d_{ij,t}\|^2 + \|\boldsymbol{b}^c - \boldsymbol{b}^c_{ij,t}\|^2 + \|\boldsymbol{b}^n\|^2) \\
& + C\xi + \tau(r_{ij,t} l_{i,t}(\boldsymbol{w}^c, \boldsymbol{w}^n, \boldsymbol{b}^c, \boldsymbol{b}^n) - \xi) - \eta\xi
\end{aligned}
$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{w}^d} = 0 \quad \Longrightarrow \quad \boldsymbol{w}^d = \boldsymbol{w}^d_{ij,t} \tag{4.15}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{w}^c} = 0 \quad \Longrightarrow \quad \boldsymbol{w}^c = \boldsymbol{w}^c_{ij,t} + \tau r_{ij,t} y_{i,t} \boldsymbol{x}^c_t \tag{4.16}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{w}^n} = 0 \quad \Longrightarrow \quad \boldsymbol{w}^n = \tau r_{ij,t} y_{i,t} \boldsymbol{x}^n_t \tag{4.17}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^d} = 0 \quad \Longrightarrow \quad \boldsymbol{b}^d = \boldsymbol{b}^d_{ij,t} \tag{4.18}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^c} = 0 \quad \Longrightarrow \quad \boldsymbol{b}^c = \boldsymbol{b}^c_{ij,t} + \frac{\tau r_{ij,t} y_{i,t}}{A} \mathbf{1}^c \tag{4.19}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^n} = 0 \quad \Longrightarrow \quad \boldsymbol{b}^n = \frac{\tau r_{ij,t} y_{i,t}}{A} \mathbf{1}^n \tag{4.20}$$

$$\frac{\partial \mathscr{L}}{\partial \xi} = 0 \quad \Longrightarrow \quad \eta = C - \tau, \tag{4.21}$$

where τ and η are Lagrange multipliers, and $\mathbf{1}^n$ is the vector filled with ones in the novel feature space. Plugging the Eq. (4.15, 4.16, 4.17, 4.18, 4.19, 4.20, 4.21) into Eq. (4.14) and setting the partial derivative of $\mathscr{L}(\tau)$ with respect to τ to zero, we obtain:

$$\mathscr{L}(\tau) = -\frac{\tau^2 r^2}{2}(\|\boldsymbol{x}^c_t\|^2 + \|\boldsymbol{x}^n_t\|^2 + \frac{\|\mathbf{1}^c\|^2 + \|\mathbf{1}^n\|^2}{A}) + \tau r_{ij,t} l_{i,t}(\boldsymbol{w}^c_{ij,t}, \boldsymbol{w}^n_{ij,t}, \boldsymbol{b}^c_{ij,t}, \boldsymbol{b}^n_{ij,t}) \tag{4.22}$$

$$\tau_{ij,t} = \min\{C, \frac{l_{i,t}(\boldsymbol{w}^c_{ij,t}, \boldsymbol{w}^n_{ij,t}, \boldsymbol{b}^c_{ij,t}, \boldsymbol{b}^n_{ij,t})}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}\}, \tag{4.23}$$

where d_t is the number of features occurred at time t, i.e. $d_t = |\mathscr{X}_t|$, and $\boldsymbol{w}^n_{ij,t}, \boldsymbol{b}^n_{ij,t} = \mathbf{0}^n$. Then the update strategy of FCBC is:

$$\begin{aligned} \boldsymbol{w}_{ij,t+1} &= [\boldsymbol{w}^d_{ij,t+1}, \boldsymbol{w}^c_{ij,t+1}, \boldsymbol{w}^n_{ij,t+1}] \\ &= [\boldsymbol{w}^d_{ij,t}, \boldsymbol{w}^c_{ij,t} + \tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}^c_t, \tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}^n_t] \end{aligned} \tag{4.24}$$

$$\begin{aligned} \boldsymbol{b}_{ij,t+1} &= [\boldsymbol{b}^d_{ij,t+1}, \boldsymbol{b}^c_{ij,t+1}, \boldsymbol{b}^n_{ij,t+1}] \\ &= [\boldsymbol{b}^d_{ij,t}, \boldsymbol{b}^c_{ij,t} + \frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^c}{A}, \frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^n}{A}]. \end{aligned} \tag{4.25}$$

Similar to previous studies of PA models [85, 89, 90], we introduce the sparsity strategy for FCBC, offering an alternative for FCBC managing high-dimensional data

streams. In the sparsity strategy, a proportion $B \in (0,1]$ of the parameters in FCBC is retained, while the remaining $(1-B)$ proportion is truncated to 0. In other words, a proportion B of the elements in $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ with the largest absolute values is retained. Furthermore, to mitigate the impact of truncating selected parameters, $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ are first projected onto a L_1 ball [77], thereby ensuring that the removal of the elements with smallest absolute values results in small changes to the weight and bias vectors. The projections are defined as:

$$\boldsymbol{w}_{ij,t+1} = \min\left\{1, \frac{\lambda}{\|\boldsymbol{w}_{ij,t+1}\|_1 + \|\boldsymbol{b}_{ij,t+1}\|_1}\right\} \boldsymbol{w}_{ij,t+1} \tag{4.26}$$

$$\boldsymbol{b}_{ij,t+1} = \min\left\{1, \frac{\lambda}{\|\boldsymbol{w}_{ij,t+1}\|_1 + \|\boldsymbol{b}_{ij,t+1}\|_1}\right\} \boldsymbol{b}_{ij,t+1}, \tag{4.27}$$

where $\lambda > 0$ is the regularization parameter.

At the end of this section, we introduce two *explanations* for the functionality of FCBC from other perspectives distinct from that we delivered at the beginning, aiming to provide readers with a deeper understanding of FCBC:

- FCBC can be further interpreted as an ensemble of classifiers operating on individual features. In other words, it functions as a feature-level ensemble of linear models, represented as $\mathscr{F}_{ij} = \{\mathscr{F}_{ij1}, \mathscr{F}_{ij2}, ..., \mathscr{F}_{iju_t}\}$, where $\mathscr{F}_{ijk}(\boldsymbol{x}_t) = w_{ijk,t}x_{t,k} + b_{ijk,t}$ and the subscript k denotes the value of a vector in the feature f_k. In the prediction process, the final output of $\mathscr{F}_{ij}$ is the average of the outputs from the feature-level base linear models corresponding to the available features in $\boldsymbol{x}_t$.

- FCBC can also be interpreted as an ensemble of two models 1) a model that considers only the occurrence of features, and 2) a linear model with zero bias. In other words, FCBC determines the final prediction by evaluating both the significance of a feature's occurrence in the classification and the contribution of its value to the classification process.

4.4.2 Model Adaptation

In EPAM, FCBC models are structured within an ensemble of ensemble framework. The inherent challenges include the dynamic imbalance between the positive and negative class, as well as the uneven distribution within the negative class for each FCBC model.

For the j-th FCBC model $\mathscr{F}_{ij}$ for class c_i, the cumulative loss is:

$$Loss_{ij} = \sum_{t,y_t=c_i} r_{ij,t} l_{ij,t} + \sum_{t,y_t\neq c_i} r_{ij,t} l_{ij,t} \tag{4.28}$$

where $l_{ij,t}$ is the loss value of $\mathscr{F}_{ij}$ on the instance $(\boldsymbol{x}_t, y_t)$ at time t, and $r_{ij,t}$ is the corresponding loss weight. The first component, $\sum_{t,y_t=c_i} r_{ij,t} l_{ij,t}$, represents the loss on instances from the positive class, i.e. instances from the class c_i. The second component, $\sum_{t,y_t\neq c_i} r_{ij,t} l_{ij,t}$, accounts for the loss on instances from the negative class, i.e. instances from the remaining classes $c_k (k \neq i)$. Previous work, such as EEOF [11], treats the negative class as a single entity, applying the same loss weight to all instances belonging to the negative class to address the imbalance between positive and negative classes. Specifically, in EEOF, Eq. (4.28) is transformed to:

$$Loss_{ij} = \sum_{t,y_t=c_i} r_{+,t} l_{ij,t} + \sum_{t,y_t\neq c_i} r_{-,t} l_{ij,t} \tag{4.29}$$

where $r_{+,t}$, $r_{-,t}$ are the loss weights for instances belonging to the positive and negative classes, respectively. To balance the loss feedback from the first and the second components in Eq. (4.29), the relationship between $r_{+,t}$ and $r_{-,t}$ is defined based on the prior probability of the positive class, c_i, as follows:

$$r_{+,t} p_t^{(i)} = r_{-,t} (1 - p_t^{(i)}), \tag{4.30}$$

where $p_t^{(i)}$ is the prior probability of class c_i at time t. As analyzed earlier, the negative class is not a singular entity but comprises multiple distinct classes, $\{c_k | k \neq i\}$. Assigning a uniform weight $r_{-,t}$ for different classes, without accounting for their individual prior probabilities, is insufficient, leading to inaccurate predictions in handling real-world data streams with class evolution.

In EPAM, the error feedback for each FCBC $\mathscr{F}_{ij}$ is initially balanced by treating the negative class as a set of distinct classes. The relationship between the loss weights for different classes is defined as follows:

$$r_{+,t}p_t^{(i)} = \sum_{c_k, c_k \neq c_i} r_{k,t}p_t^{(k)}, \tag{4.31}$$

where instances from the class $c_k (k \neq i)$ are assigned with a loss weight $r_{k,t}$. Furthermore, to balance the error feedback among classes that are designated as the negative class, the following condition must hold:

For any two classes $c_\alpha, c_\beta \in \{c_k | c_k \neq c_i\}$:

$$r_{\alpha,t}p_t^{(\alpha)} = r_{\beta,t}p_t^{(\beta)}. \tag{4.32}$$

Solving Eq. (4.31) and Eq. (4.32), we obtain:

$$r_{k,t} = \frac{r_{+,t}}{n_t - 1}\frac{p_t^{(i)}}{p_t^{(k)}}, \tag{4.33}$$

where n_t is the number of observed classes until time t.

Note that only the estimated prior probability $\hat{p}_t^{(i)}$ for class c_i is available, rather than the underlying true prior probability $p_t^{(i)}$. In addition, to introduce diversity within the group of FCBC models $\mathscr{F}_i = \{\mathscr{F}_{i1}, \mathscr{F}_{i2}, ..., \mathscr{F}_{im}\}$ for class c_i, the loss weight for each base FCBC model $\mathscr{F}_{ij}$ is generated using the Poisson distribution, a common technique in online learning algorithms [67]. Setting $r_{+,t} = 1$, the loss weight $r_{ij,t}$ for $\mathscr{F}_{ij}$ is generated by:

$$r_{ij,t} \sim \begin{cases} Poisson(1), & y_t = c_i \\ Poisson(\hat{p}_t^{(i)}/((n_t - 1) * \hat{p}_t^{(k)})), & y_t \neq c_i. \end{cases} \tag{4.34}$$

The model adaptation of EPAM is summarized in Algorithm 4.1. When an instance $(\boldsymbol{x}_t, y_t)$ arrives, the assigned label $y_{i,t}$ and the loss weight $r_{ij,t}$ are first computed for each FCBC $\mathscr{F}_{ij}$ (lines 3 to 10). Then, the weight and bias vectors in $\mathscr{F}_{ij}$ are updated (lines 11 to 13). If needed, an optional sparsity process is applied (lines 14 to 17).

Algorithm 4.1 Adaptation of EPAM

Input:

- $(\boldsymbol{x}_t, y_t)$: Instance with $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$
- $\mathscr{F}$: EPAM model with $\mathscr{F} = \{\mathscr{F}_{ij}\}$
- $\hat{\boldsymbol{p}}_t$: Estimated prior probability vector with $\hat{\boldsymbol{p}}_t = [\hat{p}_t^{(i)}]$
- n_t: Number of seen classes
- C: Tradeoff parameter between rigidness and slackness
- A: Tradeoff parameter between weight and bias vectors
- $sparsity$: Whether to process the sparsity steps
- λ: Regularization parameter
- B: Proportion of selected parameters

Output: Updated $\mathscr{F}$.

for $i \leftarrow 1 : n_t$ **do**
  for $j \leftarrow 1 : m$ **do**
    if $y_t == c_i$ **then**
      $r_{ij,t} \sim Poisson(1)$
      $y_{i,t} \leftarrow +1$
    else
      $k \leftarrow Find(k | c_k = y_t)$
      $r_{ij,t} \sim Poisson(\hat{p}_t^{(i)} / ((n_t - 1) * \hat{p}_t^k))$
      $y_{i,t} \leftarrow -1$
    end if
    Identify the common, disappeared, novel feature spaces
    Calculate $\tau_{ij,t}$ using Eq. (4.23)
    Update $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ in $\mathscr{F}_{ij}$ using Eq. (4.24, 4.25)
    if $sparsity$ is True **then**
      Project $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ to a L_1 ball using Eq. (4.26, 4.27)
      Truncate $\boldsymbol{w}_{ij,t}$ and $\boldsymbol{b}_{ij,t}$ by remaining a ratio of B elements with largest absolute values
    end if
  end for
end for

4.4.3 Overview of EPAM

After introducing the FCBC and the novel model adaptation method in EPAM, we can now integrate these techniques to address the problem of online learning in open data space. An overview of the proposed EPAM in the data stream is displayed in Algorithm 4.2.

In the data stream, EPAM first produces a prediction for each incoming instance, and then updates upon receiving the true label. The prediction process (line 6) incorporates the activation/deactivation strategy of OVA models in response to class reoccurrence and class disappearance, as well as the confidence-triggered fallback mode proposed in EEOF. The implementation details can be found in [11]. If the true label y_t of $\boldsymbol{x}_t$ corresponds to a novel class, m FCBC models are initialized with weight and bias vectors filled with 0s in the feature space $\mathscr{X}_t$ of $\boldsymbol{x}_t$ (line 11). These models are subsequently incorporated into the EPAM model $\mathscr{F}$ (line 12). In addition, the prior probability for this novel class is initialized (lines 13 and 14). At the end of each update process, the estimated prior probability vector $\hat{\boldsymbol{p}}_t$ and the EPAM model are updated (lines 16 and 17).

Algorithm 4.2 An overview of EPAM in data stream

$\mathscr{F} \leftarrow \emptyset$, $\hat{\boldsymbol{p}}_t \leftarrow \emptyset$ // Initialize EPAM model and estimated prior probability vector
while Instance $\boldsymbol{x}_t$ arrives **do**
  // *Predict*
  if $t == 1$ **then**
    $\hat{y}_t \leftarrow 0$ // *First instance in data stream*
  else
    Predict $\hat{y}_t$ using $\mathscr{F}$ // *Same method in EEOF*
  end if
  // *Update*
  Receive true label y_t
  Identify whether y_t is a novel, disappeared, or reoccurred class.
  if y_t is a novel class **then**
    Initialize m FCBC models $\mathscr{F}_{novel} = \{\mathscr{F}_{ij} | c_i = y_t, 1 \leq j \leq m\}$ for class y_t
    $\mathscr{F} \leftarrow \mathscr{F} \cup \mathscr{F}_{novel}$
    Initialize $\hat{p}_{novel}$ for class y_t
    $\hat{\boldsymbol{p}}_t \leftarrow \hat{\boldsymbol{p}}_t \cup \{\hat{p}_{novel}\}$
  end if
  Update $\hat{\boldsymbol{p}}_t$ with the occurrence of y_t using Eq. (4.8)
  Update $\mathscr{F}$ using Algorithm 4.1
end while

4.5 Theoretical Analysis

In this section, we first analysis the relative loss bounds [16] of FCBC in binary classification. Then we discuss the performance bound of the ensemble model, EPAM, for handling the multi-class classification. Our bounds essentially demonstrate that every FCBC component within EPAM cannot do much worse than the best fixed classifier chosen in hindsight for any given sequence of instances. This ensures the performance of EPAM in managing multi-class classification in data streams.

Let us begin by considering each FCBC model $\mathscr{F}_{ij}$, where $\mathscr{F}_{ij}$ represents the j-th classifier in EPAM, tasked with distinguishing whether an instance belongs to class c_i or not. We denote by $l_{ij,t}$ the loss suffered by $\mathscr{F}_{ij}$ at time t. In addition, we denote by $l^*_{ij,t}$ the loss suffered by an arbitrary fixed classifier on the same task. Formally, let $\boldsymbol{u}$

and $\boldsymbol{v}$ be two arbitrary vectors in $\mathbb{R}^{u_T}$, where u_T represents the number of all features observed up to time T, i.e. $\boldsymbol{u}, \boldsymbol{v} \in \bigcup_{t=1}^{T} \mathscr{X}_t \subseteq \mathbb{R}^{u_T}$. We define $l_{ij,t}$ and $l^*_{ij,t}$ as follows:

$$l_{ij,t} = l(\boldsymbol{w}_t, \boldsymbol{b}_t; (\boldsymbol{x}_t, y_{i,t}))$$

$$l^*_{ij,t} = l(\boldsymbol{u}, \boldsymbol{v}; (\boldsymbol{x}_t, y_{i,t}))$$

Then we have Lemma 1 as following:

Lemma 1. *Let* $(\boldsymbol{x}_1, y_{i,1}), ..., (\boldsymbol{x}_T, y_{i,T})$ *be a sequence of training instances for* $\mathscr{F}_{ij}$*, where* $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$ *and* $y_{i,t} \in [+1, -1]$ *for* $t \in [1, T]$*. Let* $r_{ij,t} > 0$ *be the loss weight and* $\tau_{ij,t} \in \{\frac{l_{ij,t}}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}, \min\{C, \frac{l_{ij,t}}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}\}\}$ *be the learning rate. The following bound holds for any* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{u_T}$*:*

$$\sum_{t=1}^{T} \tau_{ij,t} r_{ij,t} (2 l_{ij,t} - 2 l^*_{ij,t} - \tau_{ij,t} r_{ij,t} (\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})) \leq \|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2$$

Proof. Define $\Delta_t = \Delta_{1,t} + A\Delta_{2,t}$, where $\Delta_{1,t} = \|\boldsymbol{w}_t - \boldsymbol{u}_{\|\boldsymbol{w}_t}\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{\|\boldsymbol{w}_{t+1}}\|^2$ and $\Delta_{2,t} = \|\boldsymbol{b}_t - \boldsymbol{v}_{\|\boldsymbol{b}_t}\|^2 - \|\boldsymbol{b}_{t+1} - \boldsymbol{v}_{\|\boldsymbol{b}_{t+1}}\|^2$. We prove the lemma by summing Δ_t over all t and bounding this sum. Note that both $\sum_t \Delta_{1,t}$ and $\sum_t \Delta_{2,t}$ are telescopic sums. Then the sum $\sum_t \Delta_{1,t}$ collapses to $\sum_{t=1}^{T} \Delta_{1,t} = \|\boldsymbol{w}_1 - \boldsymbol{u}_{\|\boldsymbol{w}_1}\|^2 - \|\boldsymbol{w}_{T+1} - \boldsymbol{u}_{\|\boldsymbol{w}_{T+1}}\|^2$, where $\boldsymbol{w}_1$ is initialized as $\boldsymbol{0}$, and $\|\boldsymbol{w}_{T+1} - \boldsymbol{u}_{\|\boldsymbol{w}_{T+1}}\|^2 \geq 0$. Thus, we can upper bound the right-hand side of $\sum_{t=1}^{T} \Delta_{1,t}$ by $\|\boldsymbol{u}_{\|\boldsymbol{w}_1}\|^2$. Similarly, we have $\sum_{t=1}^{T} \Delta_{2,t} \leq \|\boldsymbol{v}_{\|\boldsymbol{b}_1}\|^2$. Then we conclude that:

$$\sum_{t=1}^{T} \Delta_t \leq \|\boldsymbol{u}_{\|\boldsymbol{w}_1}\|^2 + A\|\boldsymbol{v}_{\|\boldsymbol{b}_1}\|^2. \tag{4.35}$$

We now turn to bounding every $\Delta_{1,t}$ and $\Delta_{2,t}$ from below. For time t when the minimum margin requirement is not violated, $l_{ij,t} = 0$, $\tau_{ij,t} = 0$ and therefore $\Delta_{1,t} = \|\boldsymbol{w}_t - \boldsymbol{u}_{\|\boldsymbol{w}_t}\|^2 - \|[\boldsymbol{w}_t, \boldsymbol{0}] - \boldsymbol{u}_{\|\boldsymbol{w}_{t+1}}\|^2 = -\|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2$, and similarly $\Delta_{2,t} = -\|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2$. Then we have the following for $l_{ij,t} = 0$:

$$\Delta_t = -\|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 - A\|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2. \tag{4.36}$$

For time t when $l_{ij,t} > 0$, using the update strategy

$$\boldsymbol{w}_{t+1} = [\boldsymbol{w}_t^d, \boldsymbol{w}_t^c + \tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}_t^c, \tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}_t^n] \tag{4.37}$$

$$\boldsymbol{b}_{t+1} = [\boldsymbol{b}_t^d, \boldsymbol{b}_t^c + \frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^c}{A}, \frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^n}{A}], \tag{4.38}$$

we have

$$\begin{aligned}
\Delta_{1,t} =& \|\boldsymbol{w}_t - \boldsymbol{u}_{\|\boldsymbol{w}_t}\|^2 - \|\boldsymbol{w}_t^d - \boldsymbol{u}_{\|\boldsymbol{w}_t^d}\|^2 - \|\boldsymbol{w}_t^c + \tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}_t^c - \boldsymbol{u}_{\|\boldsymbol{w}_t^c}\|^2 \\
& - \|\tau_{ij,t} r_{ij,t} y_{i,t} \boldsymbol{x}_t^n - \boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 \\
=& -2\tau_{ij,t} r_{ij,t} y_{i,t} (\boldsymbol{x}_t^c \boldsymbol{w}_t^c - \boldsymbol{x}_t \boldsymbol{u}_{\|\boldsymbol{x}_t}) - \tau_{ij,t}^2 r_{ij,t}^2 \|\boldsymbol{x}_t\|^2 - \|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 \qquad (4.39) \\
\Delta_{2,t} =& \|\boldsymbol{b}_t - \boldsymbol{v}_{\|\boldsymbol{b}_t}\|^2 - \|\boldsymbol{b}_t^d - \boldsymbol{v}_{\|\boldsymbol{b}_t^d}\|^2 - \|\boldsymbol{b}_t^c + \frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^c}{A} - \boldsymbol{v}_{\|\boldsymbol{b}_t^c}\|^2 \\
& - \|\frac{\tau_{ij,t} r_{ij,t} y_{i,t} \mathbf{1}^n}{A} - \boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2 \\
=& -\frac{2\tau_{ij,t} r_{ij,t} y_{i,t}}{A} (\mathbf{1}^c \boldsymbol{b}_t^c - \mathbf{1}^{\boldsymbol{x}_t} \boldsymbol{v}_{\|\boldsymbol{x}_t}) - \frac{\tau_{ij,t}^2 r_{ij,t}^2}{A^2} \|\mathbf{1}^{\boldsymbol{x}_t}\|^2 - \|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2, \qquad (4.40)
\end{aligned}$$

where $\mathbf{1}^{\boldsymbol{x}_t}$ is the vector filled with ones and has the same number of features as $\boldsymbol{x}_t$. Note that $\|\mathbf{1}^{\boldsymbol{x}_t}\|^2 = d_t$.

Cause $l_{ij,t} > 0$, we have $l_{ij,t} = 1 - y_{i,t}(\boldsymbol{w}_t^c \cdot \boldsymbol{x}_t^c + \boldsymbol{b}_t^c \cdot \mathbf{1}^c)$. Also, $l_{ij,t}^* \geq 1 - y_{i,t}(\boldsymbol{u}_{\|\boldsymbol{x}_t} \cdot \boldsymbol{x}_t + \boldsymbol{v}_{\|\boldsymbol{x}_t} \cdot \mathbf{1})$ due to the definition of hinge loss. Using these two facts in Eq. (4.39) and Eq. (4.40) and the definition of Δ_t give:

$$\Delta_t \geq -2\tau_{ij,t} r_{ij,t} (l_{ij,t}^* - l_{ij,t}) - \tau_{ij,t}^2 r_{ij,t}^2 \|\boldsymbol{x}_t\|^2 - \frac{\tau_{ij,t}^2 r_{ij,t}^2 d_t}{A} - \|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 - A\|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2. \tag{4.41}$$

Summing over Δ_t over all t with Eq. (4.36) and Eq. (4.41), we obtain that

$$\begin{aligned}
\sum_{t=1}^{T} \Delta_t \geq & \sum_{t=1}^{T} \tau_{ij,t} r_{ij,t} (2l_{ij,t} - 2l_{ij,t}^* - \tau_{ij,t} r_{ij,t} \|\boldsymbol{x}_t\|^2 - \frac{\tau_{ij,t} r_{ij,t} d_t}{A}) \\
& - \sum_{t=1}^{T} \|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 - A \sum_{t=1}^{T} \|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2 \qquad (4.42)
\end{aligned}$$

Note that $\boldsymbol{w}_1$ and $\boldsymbol{b}_1$ are initialized with 0 and have the same size of $\boldsymbol{x}_1$. The common features vector $\boldsymbol{x}_1^n$ has a length of zero. Therefore, we have $\|\boldsymbol{u}_{\|\boldsymbol{w}_1}\|^2 + \sum_{t=1}^{T} \|\boldsymbol{u}_{\|\boldsymbol{x}_t^n}\|^2 = \|\boldsymbol{u}\|^2$,

and $\|\boldsymbol{v}_{\|\boldsymbol{w}_1}\|^2 + \sum_{t=1}^{T} \|\boldsymbol{v}_{\|\boldsymbol{x}_t^n}\|^2 = \|\boldsymbol{v}\|^2$. Using these two facts and comparing the lower bound in Eq. (4.42) with the upper bound in Eq. (4.35), we obtain:

$$\sum_{t=1}^{T} \tau_{ij,t} r_{ij,t} (2l_{ij,t} - 2l_{ij,t}^* - \tau_{ij,t} r_{ij,t} (\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})) \leq \|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2.$$

This lemma is proved. ■

In the following, we prove the upper bound of the cumulative squared loss of FCBC without the influence of the slack term $C\xi$ in the linear separable case.

Theorem 1. *Let* $(\boldsymbol{x}_1, y_{i,1}), ..., (\boldsymbol{x}_T, y_{i,T})$ *be a sequence of training instances for* $\mathscr{F}_{ij}$*, where* $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$ *and* $y_{i,t} \in [+1, -1]$*, and* $\|\boldsymbol{x}_t\|^2 \leq R^2$ *for all* $t \in [1, T]$*. Assume that there exists a classifier* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{u_T}$ *such that* $l_{ij,t}^* = 0$ *for all* t*. Let* $r_{ij,t} > 0$ *be the loss weight and the learning rate* $\tau_{ij,t} = \frac{l_{ij,t}}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}$*. The cumulative squared loss of FCBC on this sequence is bounded by:*

$$\sum_{t=1}^{T} l_{ij,t}^2 \leq (\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2)(R^2 + \frac{u_T}{A}).$$

Proof. Since $l_{ij,t}^* = 0$ for all t, Lemma 1 implies:

$$\sum_{t=1}^{T} \tau_{ij,t} r_{ij,t} (2l_{ij,t} - \tau_{ij,t} r_{ij,t} (\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})) \leq \|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2.$$

Using the definition of $\tau_{ij,t}$, $\|\boldsymbol{x}_t\|^2 \leq R^2$, and the fact that $d_t \leq u_T$, we have:

$$\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2 \geq \sum_{t=1}^{T} \frac{l_{ij,t}^2}{(R^2 + \frac{u_T}{A})}$$

We then obtain $\sum_{t=1}^{T} l_{ij,t}^2 \leq (\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2)(R^2 + \frac{u_T}{A})$. ■

We then extend the analysis from the separable case to the inseparable one, where no classifier with any parameters $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{u_T}$ can perfectly separate instances from two classes.

Theorem 2. *Let* $(\boldsymbol{x}_1, y_{i,1}),...,(\boldsymbol{x}_T, y_{i,T})$ *be a sequence of training instances for* $\mathscr{F}_{ij}$*, where* $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$ *and* $y_{i,t} \in [+1,-1]$*, and* $\|\boldsymbol{x}_t\|^2 = 1$ *for all* $t \in [1,T]$*. Let* $r_{ij,t} > 0$ *be the loss weight and the learning rate* $\tau_{ij,t} = \frac{l_{ij,t}}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}$*. For any vector* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{u_T}$*, the cumulative squared loss of FCBC on this sequence is bounded by:*

$$\sum_{t=1}^{T} l_{ij,t}^2 \leq \left((1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) + 2\sqrt{\sum_{t=1}^{T} (l_{ij,t}^*)^2} \right)^2 .$$

Proof. Using the definition of $\tau_{ij,t}$, $\|\boldsymbol{x}_t\|^2 = 1$, and the fact that $d_t \leq u_T$, Lemma 1 implies:

$$\sum_{t=1}^{T} l_{ij,t}^2 \leq (1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) + \sum_{t=1}^{T} 2 l_{ij,t} l_{ij,t}^*. \tag{4.43}$$

Denote

$$L_T = \sqrt{\sum_{t=1}^{T} l_{ij,t}^2}$$

$$U_T = \sqrt{\sum_{t=1}^{T} (l_{ij,t}^*)^2}.$$

By using the Cauchy-Schwartz inequality, we obtain $L_T^2 U_T^2 \geq (\sum_{t=1}^{T} l_{ij,t} l_{ij,t}^*)^2$. Then, Eq. (4.43) is upper bounded:

$$L_T^2 \leq (1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) + 2 L_T U_T.$$

To obtain an upper bound of L_T, we need to find the largest solution L_T^* of $L_T^2 - (1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) - 2 L_T U_T = 0$. This gives that:

$$L_T \leq L_T^* = U_T + \sqrt{U_T^2 + (1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2)}.$$

Using the fact that $\sqrt{\alpha + \beta} \leq \sqrt{\alpha} + \sqrt{\beta}$, we get:

$$L_T \leq (1 + \frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) + 2U_T$$

Then, we obtain

$$\sum_{t=1}^{T} l_{ij,t}^2 \leq \left((1+\frac{u_T}{A})(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2) + 2\sqrt{\sum_{t=1}^{T}(l_{ij,t}^*)^2} \right)^2 .$$

This theorem is proved. ■

Now, we turn to analyze the performance bound of the original FCBC with the slack term $C\xi$.

Theorem 3. *Let* $(\boldsymbol{x}_1, y_{i,1}), ..., (\boldsymbol{x}_T, y_{i,T})$ *be a sequence of training instances for* $\mathscr{F}_{ij}$*, where* $\boldsymbol{x}_t \in \mathscr{X}_t \subseteq \mathbb{R}^{d_t}$ *and* $y_{i,t} \in [+1, -1]$*, and* $\|\boldsymbol{x}_t\|^2 \leq R^2$ *for all* $t \in [1, T]$*. Let* $r_{ij,t}$ *be the loss weight with* $r_{ij,t} \geq r^* > 0$ *for all t, and the learning rate* $\tau_{ij,t} = \min\{C, \frac{l_{ij,t}}{r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A})}\}$*. For any vector* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{u_T}$*, the number of false predictions by FCBC is bounded by:*

$$M \leq \max\{\frac{1}{r^* C}, R^2 + \frac{u_T}{A}\}(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2 + 2C\sum_{t=1}^{T} r_{ij,t} l_{ij,t}^*).$$

Proof. If FCBC $\mathscr{F}_{ij}$ produces a false prediction at time t, the suffered loss $l_{ij,t} \geq 1$. Considering the fact that $d_t \leq u_T$, then we have

$$\sum_{t=1}^{T} \tau_{ij,t} r_{ij,t} l_{i,t} \geq \sum_{t=1}^{T} \min\{r^* C, \frac{l_{ij,t}}{R^2 + \frac{u_T}{A}}\} l_{ij,t} \geq \min\{r^* C, \frac{1}{R^2 + \frac{u_T}{A}}\} M, \tag{4.44}$$

where M is the total number of false predictions.

Note that $\tau_{ij,t} l_{ij,t}^* \leq C l_{ij,t}^*$ and $\tau_{ij,t} r_{ij,t}(\|\boldsymbol{x}_t\|^2 + \frac{d_t}{A}) \leq l_{ij,t}$. Plugging these two inequalities into Lemma 1 gives

$$\sum_{t=1}^{T} (\tau_{ij,t} r_{ij,t} l_{ij,t} - 2 r_{ij,t} C l_{ij,t}^*) \leq \|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2. \tag{4.45}$$

Combining Eq. (4.44) and Eq. (4.45), we obtain that:

$$\min\{r^* C, \frac{1}{R^2 + \frac{u_T}{A}}\} M \leq \|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2 + 2C\sum_{t=1}^{T} r_{ij,t} l_{ij,t}^*$$

Multiplying both sides with $\max\{\frac{1}{r^* C}, R^2 + \frac{u_T}{A}\}$, we obtain that

$$M \leq \max\{\frac{1}{r^* C}, R^2 + \frac{u_T}{A}\}(\|\boldsymbol{u}\|^2 + A\|\boldsymbol{v}\|^2 + 2C\sum_{t=1}^{T} r_{ij,t} l_{ij,t}^*).$$

This theorem is proved. ■

The proposed model, EPAM, is an ensemble of FCBC models. For an instance $(\boldsymbol{x}_t, y_t)$, the loss of the prediction result originate from each group of classifiers $\mathscr{F}_i = \{\mathscr{F}_{ij} | c_i = y_t, 1 \leq j \leq m\}$. The ideal prediction score for $\mathscr{F}_i$ is 1 when $c_i = y_t$ and -1 when $c_i \neq y_t$. Using hinge loss for each $\mathscr{F}_i$, the loss of EPAM can be describe as:

$$Loss(\boldsymbol{x}_t) = \sum_{i, c_i = y_t} \max\{0, 1 - (+1)\mathscr{F}_i(\boldsymbol{x}_t)\} + \sum_{i, c_i \neq y_t} \max\{0, 1 - (-1)\mathscr{F}_i(\boldsymbol{x}_t)\}.$$

Using the definition $\mathscr{F}_i(\boldsymbol{x}_t) = \frac{1}{m}\sum_j \mathscr{F}_{ij}(\boldsymbol{x}_t)$ and the convexity of the function $\max\{0, x\}$, we have:

$$\begin{aligned} Loss(\boldsymbol{x}_t) \leq & \frac{1}{m} \sum_{i, c_i = y_t} \sum_j \max\{0, 1 - (+1)\mathscr{F}_{ij}(\boldsymbol{x}_t)\} + \frac{1}{m} \sum_{i, c_i \neq y_t} \sum_j \max\{0, 1 - (-1)\mathscr{F}_{ij}(\boldsymbol{x}_t)\} \\ = & \frac{1}{m} \sum_i \sum_j l_{ij,t}. \end{aligned}$$

This demonstrates EPAM's loss is bounded by the performance of each FCBC model. The relative loss bounds of FCBC ensure that the performance of EPAM cannot do much worse than the best fixed classifier in handling data streams with multi-class classification.

4.6 Experimental Study

In this section, we conduct a comprehensive experimental study to demonstrate the superiority of EPAM in addressing the problem of online learning in open data space. These experiments are designed to answer the following research questions (RQs):

- **RQ1**. Why is it necessary to develop online learning methods for handling the data stream with open data space? Do the constructed online learning methods, including EPAM and baseline methods, show better performance compared to the method that rely on extensive data storage for model adaptation?

- **RQ2**. Does the proposed online learning model, EPAM, outperform the baseline methods constructed from the state-of-the-art online learning approaches in both fields of class evolution and feature evolution?
- **RQ3**. Do the novel components proposed in EPAM enhance the performance in handling real-world data streams with open data space?

For RQ1/RQ2, we evaluate models on a variety of synthetic and real-world data streams. For RQ3, we conduct an ablation study to assess the significance of different components in EPAM on handling real-world data streams.

4.6.1 Experimental Settings

1) Data Stream

The data streams used in the following experiments cover three primary types based on the synthetic properties:

- Synthetic class evolution and synthetic feature evolution
- Real-world class evolution and synthetic feature evolution
- Real-world data streams with open data space.

For the first type, we generate data streams from three tabular datasets, namely Letter (Letter Recognition) [72], Statlog (Landsat Satellite), and Covertype [8]. For the second type, we begin with three real-world data streams, KDDCUP99 data stream [76] Poker-hand data stream [13] and UNSW-NB15 data stream [63], which exhibit only class evolution, and simulate feature evolution within them. For the third type, we use four preprocessed data streams of TwitterCrawl data stream [50], specifically Tweet Stream - A/B/C/20 classes as outlined in [75]. In these data streams, each instance represents a tweet, with the label assigned based on the hashtag describing its topic. Each feature

corresponds to an individual word. In [59, 60], Tweet data is also utilized to investigate the coexistence of class evolution and feature evolution. While studies on class evolution [11, 75] also use Tweet data, they assume that all dimensions are available from the outset of the data stream. However, it is impossible to ascertain which words will emerge in the subsequent continuously generated tweets.

For generating class evolution, we adopt the same approach as in the former work [75], and we similarly consider three-class scenarios. In the class emergence scenarios, one class is designated as the emerging class, while the other two are existing classes. In the scenarios of class disappearance and reoccurence, one class is the disappeared and reoccurred class, with the other two remaining as existing classes. For generating feature evolution, we simulate in the forms of capricious data streams as in the former work [40], in which feature spaces change randomly, and trapezoidal data streams as in the former work [89], a special case in which novel features emerge without any features disappearing. The key characteristics of all data streams with open data space are displayed in Table 4.1.

2) Evaluation Metrics

As referred in Section 4.2, the task of online learning in open data space is to make accurate predictions for instances in the data stream. Considering the issue of class evolution in the data stream, we utilize the same evaluation metrics to assess the multi-class classification performance of different algorithms, as described in Eq. (2.6). To reduce the impact of random variations, each model is independently run 10 times on each data stream. Differences between algorithms are evaluated using the Wilcoxon rank-sum test [28] to assess statistical significance.

3) Compared Approaches

Table 4.1: Characteristics of data streams with open data space in experiments.

Data Stream	Instances	Features	Classes	Class Evolution Types	Feature Evolution Types
Letter(EC)	4500	16	3	Class Emergence	Capricious
Letter(DC)				Class Disappearance and Reoccurrence	Capricious
Letter(ET)				Class Emergence	Trapezoidal
Letter(DT)				Class Disappearance and Reoccurrence	Trapezoidal
Statlog(EC)		36		Class Emergence	Capricious
Statlog(DC)				Class Disappearance and Reoccurrence	Capricious
Statlog(ET)				Class Emergence	Trapezoidal
Statlog(DT)				Class Disappearance and Reoccurrence	Trapezoidal
Covertype(EC)		54		Class Emergence	Capricious
Covertype(DC)				Class Disappearance and Reoccurrence	Capricious
Covertype(ET)				Class Emergence	Trapezoidal
Covertype(DT)				Class Disappearance and Reoccurrence	Trapezoidal
KDDCUP99(C)	494021	117	23	Class Emergence, Disappearance and Reoccurrence	Capricious
KDDCUP99(T)					Trapezoidal
Poker-hand(C)	829201	25	10		Capricious
Poker-hand(T)					Trapezoidal
UNSW-NB15(C)	1087136	199	10		Capricious
UNSW-NB15(T)					Trapezoidal
Tweet Stream - A	36865	242	3	Class Emergence	Capricious
Tweet Stream - B	13926	244	3	Class Disappearance and Reoccurrence	Capricious
Tweet Stream - C	63285	242	4	Class Emergence	Capricious
Tweet Stream - 20 classes	135623	524	20	Class Emergence, Disappearance and Reoccurrence	Capricious

* The letters in (·) denote the synthetic properties. E: class emergence; D: class disappearance and reoccurrence; C: capricious; T: trapezoidal. For example, Letter(EC) is the data stream generated from the Letter dataset, with synthetic class emergence for class evolution and synthetic capricious for feature evolution. KDDCUP99(C) is the data stream generated from KDDCUP99 data stream, with synthetic capricious for feature evolution, while already exhibiting inherent class evolution.

To the best of our knowledge, only two models, DXMiner [60] and MCM [59], manage class evolution and feature evolution simultaneously, with none of the existing approach designed to process data streams with open data space in an online learning manner. Therefore, the latter work MCM, which processes the data stream in a chunk-by-chunk manner, is selected for comparison. To further demonstrate the superiority of EPAM, we construct baseline methods based on the latest advancements in the domains of online learning with class evolution and online learning with feature evolution. This results in the construction of the aforementioned baseline models, EEOF-OLI^{2}DS$_1$ and EEOF-OLI^{2}DS$_2$, which integrate state-of-the-art techniques from both domains, along with the baseline model EEOF-OL$_{SF}$.

To ensure a fair comparison, the hyperparameters in MCM, EEOF-OLI^{2}DS$_1$, EEOF-OLI^{2}DS$_2$ and EEOF-OL$_{SF}$ are either set to the recommended values or determined through grid search within the suggested value ranges in their original papers. EPAM utilizes the same settings of hyperparameters as EEOF-OL$_{SF}$, with the additional parameter A selected from the set $[1,5,10,50,100,500,1000]$. The size of initial offline dataset for grid search, as well as the chunk size for MCM is set to 1000.

4.6.2 Comparisons Analysis on synthetic and real-world data streams

We examine models on data streams with different data distributions on feature space, and different dynamics in class evolution and feature evolution. The performance of EPAM and the aforementioned compared models across diverse synthetic and real-world data streams is presented in Table 4.2.

1) To answer RQ1

As observed in the columns for MCM and the baseline models, namely EEOF-

Table 4.2: The average of G-mean results (mean/std) of models over 10 runs on synthetic data streams and real-world data streams with open data space

Data Stream	**EPAM**	**EEOF-OLI^{2}DS$_1$**	**EEOF-OLI^{2}DS$_2$**	**EEOF-OL$_{SF}$**	**MCM**
Letter(EC)	**0.7015/0.0105**	0.4398/0.0174	0.4964/0.0127	0.5174/0.0226	0.2047/0.0206
Letter(DC)	**0.7412/0.0143**	0.4618/0.0198	0.5345/0.0222	0.5469/0.0249	0.3990/0.0156
Letter(ET)	**0.8326/0.0471**	0.8122/0.0508	0.8141/0.0432	0.8146/0.0435	0.3744/0.0283
Letter(DT)	**0.8619/0.0302**	0.8106/0.0480	0.8367/0.0431	0.8372/0.0426	0.6309/0.0900
Statlog(EC)	**0.8731/0.0140**	0.5128/0.0150	0.5525/0.0166	0.5633/0.0160	0.2905/0.0153
Statlog(DC)	**0.9118/0.0174**	0.5343/0.0217	0.5942/0.0250	0.5992/0.0244	0.5694/0.0306
Statlog(ET)	**0.9013/0.0153**	0.7746/0.0234	0.7649/0.0187	0.7650/0.0187	0.4262/0.0040
Statlog(DT)	**0.9447/0.0114**	0.7919/0.0240	0.7973/0.0323	0.7974/0.0324	0.8414/0.0444
Covertype(EC)	**0.5636/0.0110**	0.5323/0.0159	0.5502/0.0120	0.5626/0.0103	0.2155/0.0101
Covertype(DC)	0.5636/0.0160	0.5068/0.0118	0.5587/0.0073	**0.5684/0.0142**	0.4401/0.0397
Covertype(ET)	**0.6476/0.0448**	0.6227/0.0410	0.6455/0.0351	0.6304/0.0415	0.2729/0.0183
Covertype(DT)	**0.6328/0.0478**	0.5952/0.0407	0.6286/0.0252	0.6261/0.0273	0.4968/0.0833
KDDCUP99(C)	**0.1747/0.0038**	0.0837/0.0027	0.0801/0.0023	0.0683/0.0027	0.0186/0.0006
KDDCUP99(T)	**0.2951/0.0490**	0.2324/0.0351	0.0968/0.0158	0.0892/0.0130	0.1676/0.0572
Poker-hand(C)	**0.2866/0.0025**	0.2194/0.0030	0.2551/0.0035	0.2253/0.0043	0.0350/0.0010
Poker-hand(T)	**0.3678/0.0289**	0.2625/0.0168	0.1531/0.0138	0.2648/0.0265	0.0556/0.0133
UNSW-NB15(C)	**0.8443/0.0004**	0.8405/0.0007	0.8397/0.0005	0.8405/0.0007	0.8297/0.0000
UNSW-NB15(T)	**0.8380/0.0058**	0.8325/0.0082	0.8255/0.0080	0.8260/0.0110	0.8345/0.0044
Tweet Stream - A	**0.6967/0.0058**	0.6809/0.0000	0.6674/0.0070	0.6804/0.0080	0.2014/0.0043
Tweet Stream - B	**0.7920/0.0021**	0.7792/0.0000	0.7916/0.0028	0.7846/0.0015	0.2472/0.0246
Tweet Stream - C	**0.6593/0.0053**	0.6019/0.0000	0.6356/0.0054	0.6542/0.0066	0.1852/0.0067
Tweet Stream - 20 classes	**0.0723/0.0014**	0.0708/0.0000	0.0451/0.0018	0.0686/0.0016	0.0000/0.0000
Win/Tie/Loss	-	0/4/18	0/6/16	0/7/15	0/1/21

* The highest mean results are highlighted in bold. The Win/Tie/Loss counts represent the number of cases in which the selected model significantly outperforms, shows no significantly difference, or performs significantly worse compared with EPAM, respectively (Wilcoxon rank sum test at 95% confidence level).

OLI^{2}DS$_1$, EEOF-OLI^{2}DS$_2$ and EEOF-OL$_{SF}$, MCM generally underperforms across most cases, with better performance only on Statlog(DC), Statlog(DT), and KDDCUP99(T). In addition, as indicated by the Win/Tie/Loss metric, MCM performs significantly worse than EPAM in almost all cases.

In MCM, instances are initially gathered in chunks. In each chunk, a lossless homogenizing conversion is applied to create a unified feature space for all instances within. A base classifier is subsequently built by utilizing all all observed features and classes in this chunk. Although this approach may seem effective, it is unsuitable for managing the dynamic data stream with continuous evolving sets of features and classes. MCM requires substantial data collection before it can adapt to novel classes and novel features, leading to delays in tracking the latest distribution in the data stream and resulting in suboptimal predictions. In particular, on the Tweet Stream - 20 classes, classes and features emerge abruptly and disappear rapidly, meaning those classes and features encountered in one chunk may disappear and never reoccur in subsequent chunks. As a result, MCM fails to achieve multi-class classification within any individual sliding window under the G-mean metric. Online learning models, which processes instances one-by-one, have the advantange of closely tracking dynamics of class evolution and feature evolution, demonstrating their powerful performance on the dynamic data stream with open data space.

2) To answer RQ2

As observed in the columns for EPAM and other online learning models, the Win/Tie/Loss results indicate that EPAM demonstrates significantly superior performance in most cases and comparable performance in a few.

Specifically, compared to the highest results among the baseline online learning models, EPAM exhibits substantial improvements, achieving gains of 35.6% on Let-

ter(EC), 35.5% on Letter(DC), 55.0% on Statlog(EC), 52.2% on Statlog(DC), 12.3% on Poker-hand(C), and 109% on KDDCUP99(C). All aforementioned data streams have the capricious type of feature evolution. We attribute these significant improvements to the employment of FCBC in EPAM. In capricious data streams, features with different data distributions emerge and disappear randomly. The integration of our proposed adaptive bias vector in FCBC offers a more comprehensive solution by catering a multitude of potential feature combinations and identifying the optimal decision boundary between the positive class and the negative class, thereby enhancing the multi-class classification performance of EPAM. The Covertype data streams, including Covertype(EC), Covertype(DC), Covertype(ET), and Covertype(DT), are special cases where linear models with zero biases effectively separate classes. For these data streams, higher values of the parameter A, which balance the trade-off between weight and bias vectors in the objective function (4.12), are selected via grid search on the initial offline dataset. This helps prevent EPAM from overfitting, resulting in comparable or slightly improved performance on the Covertype data streams. In trapezoidal data streams, the combinations of features are limited. In cases such as Letter(ET), Letter(DT), Statlog(ET), and Statlog(DT), EPAM still demonstrates notable improvements, albeit to a lesser degree.

The primary distinction between EEOF-OLI^{2}DS$_1$ and EEOF-OLI^{2}DS$_2$ lies in the generation of the loss weight. A comparison between these models reveals that the time-decayed dynamic class ratio in Eq. (4.8) is more effective than the cumulative class ratio in Eq. (4.10) at tracking dynamic class imbalance and enabling online learning models to adapt to class evolution. However, models employing the time-decayed dynamic class ratio, specifically EEOF-OLI^{2}DS$_2$ and EEOF-OL$_{SF}$, both fail on the KDDCUP99(T) data stream. In KDDCUP99, there are many small classes with the tiniest class containing only two instances across the whole data stream. For a small class c_i, it is treated as the negative class for OVA classifiers $\mathscr{F}_k(k \neq i)$. In models with the time-decayed

Table 4.3: Integrated components in EPAM and its variants for ablation study

Algorithm	**Model Adaptation**			**OVA Classifier**		
	Ori.	*Singular*	*Static*	*Ori.*	*Zero-bias*	*Scalar-bias*
EPAM	✓	-	-	✓	-	-
EPAM-A1	-	✓	-	✓	-	-
EPAM-A2	-	-	✓	✓	-	-
EPAM-C1	✓	-	-	-	✓	-
EPAM-C2	✓	-	-	-	-	✓

dynamic class ratio, instances from class c_i tend to provide lower loss feedback for $\mathscr{F}_k(k \neq i)$, as their loss weights decreases in conjunction with the other classes that are considered as the negative class. In the absence of adequate loss feedback from the small class c_i treated as the negative class, instances in data streams are more prone to being misclassified as class c_i, resulting in suboptimal performance in multi-class classification. The model EEOF-OLI^{2}DS$_1$ achieves better performance, however, it compromises the recalls of small classes, which only appear briefly. In this model, the cumulative class ratio struggles to adapt swiftly to the dynamic class ratios for small classes. The OVA model $\mathscr{F}_i$ for the small class c_i will receive excessive loss feedback from instances of the negative class. Consequently, fewer or even no instances, are predicted as class c_i, leading to a significant decline in the recall of class c_i. The critical factor here is the appropriate assignment of loss weights for different classes designated as the negative class. EPAM addresses this problem through its innovative model adaptation strategy in Eq. (4.33), achieving better performance in multi-class classification.

4.6.3 Ablation Study

To assess the importance of each novel component proposed in EPAM for handling real-world data streams with open data space, we conduct an ablation study from two perspectives: 1) the model adaptation approach and 2) the integrated OVA classifier. Specifically, we investigate four variants of EPAM, as displayed in Table 4.3.

For clarity and conciseness, the following abbreviations are adopted: *Ori.* (Original) denotes the original version of either the novel model adaptation method or the novel OVA classifier (FCBC) in EPAM. *Singular* refers to the model adaptation method in the state-of-the-art method in online learning with class evolution, EEOF [11], as defined in Eq. (4.9), in which classes designated as the negative class are treated as a singular entity. *Static* represents the model adaptation method in the state-of-the-art method in online learning with feature evolution, OLI^{2}DS [85], which employs the cumulative class ratio, as defined in Eq. (4.10). *Zero-bias* and *Scalar-bias* correspond to the variants of FCBC in which the feature contributed bias vector is eliminated, and the bias term is assigned with a value of 0 or a scalar value for diverse feature spaces, respectively. To ensure a fair comparison, the hyperparameters of these variants are either set to match those used in EPAM or determined via grid search. Experimental results are displayed in Table 4.4 and Table 4.5, respectively.

As shown in the columns for EPAM and EPAM-A1 in Table 4.4 and Table 4.5, EPAM consistently achieves significantly better performance across all data streams. Both methods monitor the dynamic class imbalance using the time-decayed approach. However, they differ in whether they address the imbalance issue among classes designated as the negative class, which we regard as the main factor contributing to the observed improvements. EPAM also achieves significantly better or comparable performance comparing with the variant EPAM-A2. Although the cumulative class ratio method adopted in EPAM-A2 significantly outperforms the time-decayed dynamic class ratio employed in EPAM-A1 on KDDCUP99 data streams, this method compromises the recalls of minority classes, which is consistent with our previous findings. In contrast, the novel model adaptation method in EPAM takes all classes into account, thereby demonstrating its superiority in the task of multi-class classification. The comparison between EPAM and two variants, EPAM-C1 and EPAM-C2, in Table 4.4 and Table

Table 4.4: The average of G-mean results (mean/std) of EPAM and its variants over 10 runs on real-world data streams with open data space (Same hyperparameter as EPAM)

Data Stream	**EPAM**	**EPAM-A1**	**EPAM-A2**	**EPAM-C1**	**EPAM-C2**
KDDCUP99(C)	**0.1747/0.0038**	0.0976/0.0030	0.1145/0.0024	0.1228/0.0029	0.1650/0.0036
KDDCUP99(T)	0.2951/0.0490	0.1141/0.0124	0.1878/0.0270	0.2807/0.0525	**0.2964/0.0480**
Poker-hand(C)	**0.2866/0.0025**	0.2071/0.0088	0.1792/0.0015	0.1979/0.0015	0.2304/0.0011
Poker-hand(T)	**0.3678/0.0289**	0.2618/0.0187	0.1774/0.0152	0.3245/0.0324	0.3601/0.0269
UNSW-NB15(C)	**0.8443/0.0004**	0.8412/0.0005	0.8406/0.0006	0.8423/0.0004	0.8425/0.0004
UNSW-NB15(T)	**0.8380/0.0058**	0.8319/0.0072	0.8349/0.0049	0.8344/0.0048	0.8379/0.0058
Tweet Stream - A	**0.6967/0.0058**	0.6788/0.0059	0.6848/0.0000	0.6961/0.0068	0.6763/0.0088
Tweet Stream - B	**0.7920/0.0021**	0.7851/0.0011	0.7753/0.0000	0.7890/0.0021	0.7906/0.0033
Tweet Stream - C	**0.6593/0.0053**	0.6542/0.0045	0.6055/0.0000	0.6574/0.0019	0.6568/0.0072
Tweet Stream - 20 classes	0.0723/0.0014	0.0672/0.0013	0.0228/0.0000	**0.0728/0.0016**	0.0203/0.0009
Win/Tie/Loss	-	0/0/10	0/1/9	0/5/5	0/5/5

* The highest mean results are highlighted in bold. The Win/Tie/Loss counts represent the number of cases in which the selected model significantly outperforms, shows no significantly difference, or performs significantly worse compared with EPAM, respectively (Wilcoxon rank sum test at 95% confidence level).

Table 4.5: The average of G-mean results (mean/std) of EPAM and its variants over 10 runs on real-world data streams with open data space (Hyperparameters set via Grid search)

Data Stream	**EPAM**	**EPAM-A1**	**EPAM-A2**	**EPAM-C1**	**EPAM-C2**
KDDCUP99(C)	**0.1747/0.0038**	0.0440/0.0059	0.1085/0.0025	0.1395/0.0028	0.1685/0.0035
KDDCUP99(T)	0.2951/0.0490	0.1141/0.0124	0.2528/0.0415	0.2345/0.0472	**0.2964/0.0480**
Poker-hand(C)	**0.2866/0.0025**	0.2381/0.0042	0.2259/0.0017	0.2542/0.0028	0.2797/0.0017
Poker-hand(T)	**0.3678/0.0289**	0.2905/0.0181	0.3002/0.0135	0.3564/0.0375	0.3212/0.0117
UNSW-NB15(C)	**0.8443/0.0004**	0.8418/0.0007	0.8412/0.0006	0.8413/0.0006	0.8425/0.0004
UNSW-NB15(T)	**0.8380/0.0058**	0.8315/0.0062	0.8372/0.0044	0.8346/0.0044	0.8379/0.0058
Tweet Stream - A	**0.6967/0.0058**	0.6788/0.0059	0.6777/0.0000	0.6961/0.0068	0.6763/0.0088
Tweet Stream - B	**0.7920/0.0021**	0.7418/0.0052	0.7766/0.0000	0.7885/0.0019	0.7906/0.0033
Tweet Stream - C	**0.6593/0.0053**	0.6542/0.0045	0.6183/0.0000	0.6574/0.0019	0.6568/0.0072
Tweet Stream - 20 classes	0.0723/0.0014	0.0686/0.0015	0.0223/0.0000	**0.0739/0.0013**	0.0221/0.0009
Win/Tie/Loss	-	0/0/10	0/2/8	1/4/5	0/4/6

* The highest mean results are highlighted in bold. The Win/Tie/Loss counts represent the number of cases in which the selected model significantly outperforms, shows no significantly difference, or performs significantly worse compared with EPAM, respectively (Wilcoxon rank sum test at 95% confidence level).

4.5 also indicates that the feature contributed bias enhances performance, yielding results that are comparable or superior. In the real-world data streams with capricious feature evolution like KDDCUP99(C) and Poker-hand(C), the improvements from the incorporation of the feature contributed bias are remarkable. As a summary, the default framework of EPAM is generally recommended to address real-world data streams with open data space.

4.7 Summary

In this chapter, we introduce EPAM, a novel model designed to address online learning in open data space. To the best of our knowledge, this is the first study to tackle this problem, addressing challenges of feature evolution, class evolution, free of data storage for model adaptation meanwhile. We approach this problem by constructing several baseline models grounded on state-of-the-art online learning techniques for class evolution and feature evolution, analyzing their limitations, and highlighting the motivations for developing EPAM. In EPAM, a novel feature contributed bias classifier (FCBC) is incorporated with an adaptive bias vector capable of accommodating to diverse feature spaces. An innovative model adaptation strategy in EPAM is designed to manage the imbalance among classes designated as the negative class for each FCBC model. Extensive experiments on synthetic and real-world data streams demonstrate the superiority of EPAM on handling the problem of online learning in open data space.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

This thesis explores advancements in utilitarian online learning algorithms to address challenges posed by class evolution in real-world data streams. In practical applications, data streams often exhibit unpredictable behaviors, leading to issues of: 1) unexpected prior probability of the novel class, 2) limited availability of true labels, and 3) data distributed in open data space. While most existing algorithms require extensive data storage for model adaptation, existing online learning algorithms are not sufficient to manage aforementioned challenges. Particularly, these methods 1) rely on underlying assumptions about the prior probabilities of classes, which hold in stationary settings but fail in dynamic data streams, 2) presume the availability of all instances in data streams, and 3) demand a static feature space throughout. Recognizing these limitations, this thesis proposes three innovative algorithms to tackle these challenges in online learning with class evolution.

First, EEOF is proposed to handle data streams with class evolution, accommodat-

ing a broader spectrum of prior probabilities for classes, including scenarios where a novel class emerges with a dominant amount. The ensemble of ensemble structure is constructed in EEOF to introduce diversity within each OVA classifier. A novel adaptive model adaptation strategy is integrated to balance the error feedback between classes. Furthermore, a confidence-triggered fallback mode is designed to prevent incorrect recognition of disappearing classes. All these techniques ensure that EEOF outperforms in diverse data streams with class evolution.

Second, SOELF is proposed to handle data streams with class evolution under incomplete supervision. To leverage unlabeled instances, a data generated module is designed based on an ensemble online clustering model, generating diverse instances for model adaptation. Pseudo-labels for these instances are determined using the decay G-mean metric. Additionally, an inverse-prior-ratio model adaptation strategy is designed to balance the exploitation between classes, enhancing performance in multi-class classification tasks.

Finally, EPAM is proposed to address the problem of online learning with class evolution and feature evolution simultaneously, where instances are distributed in open data space. In EPAM, FCBC is integrated as the base OVA classifier with an adaptive bias vector that is capable of adapting to feature evolution. Furthermore, a model adaptation strategy is designed to balance the error feedback among the negative class of each FCBC model, thereby enhancing the multi-class classification performance in data streams with class evolution.

In each study, comprehensive experiments on a variety of synthetic and real-world data streams are conducted, demonstrating the superiority of these algorithms as utilitarian solutions to real-world challenges.

5.2 Future Work

Despite the advancements achieved in this research, there still remains numerous opportunities for future work and further improvements.

1. **Multi-label Classification.** In multi-label classification, a single instance belongs to more than one classes at the same time [88]. For example, a tweet may contains multiple rather than one popular topics to be categorized into [46]. While several studies have developed online learning algorithms [66, 96], they assume that the class set remains the same throughout the data stream. Future work could focus on developing algorithms to address class evolution in the multi-label classification.

2. **Non-linearly Separable in Open Data Space.** In real-world scenarios, instances from different classes may be non-linearly separable. The popular passive-aggressive model is not sufficient. Several non-linear online learning models [7, 52] have been proposed to address feature evolution, yet they neglect the issue of class evolution. Designing algorithms for non-linearly separable cases in open data space is a valuable endeavor but it remains an open problem.

3. **Temporal Dependency in Data Streams.** The temporal property is one of the important characteristics that distinguishes data streams from offline datasets [5]. The issue of temporal dependency in the data streams means that the other common assumption, independently-distributed assumption, is no longer valid. However, most data stream mining techniques overlook this problem. The combined challenge of class evolution and temporal dependency remains unexplored.

BIBLIOGRAPHY

[1] C. C. AGGARWAL, P. S. YU, J. HAN, AND J. WANG, *A framework for clustering evolving data streams*, in Proceedings 2003 VLDB Conference, Morgan Kaufmann, 2003, pp. 81–92.

[2] T. AL-KHATEEB, M. M. MASUD, L. KHAN, C. AGGARWAL, J. HAN, AND B. THURAISINGHAM, *Stream classification with recurring and novel class detection using class-based ensemble*, in 2012 IEEE 12th International Conference on Data Mining, 2012, pp. 31–40.

[3] E. L. ALLWEIN, R. E. SCHAPIRE, AND Y. SINGER, *Reducing multiclass to binary: A unifying approach for margin classifiers*, J. Mach. Learn. Res., 1 (2001), pp. 113–141.

[4] R. A. ARIYALURAN HABEEB, F. NASARUDDIN, A. GANI, I. A. TARGIO HASHEM, E. AHMED, AND M. IMRAN, *Real-time big data processing for anomaly detection: A survey*, International Journal of Information Management, 45 (2019), pp. 289–307.

[5] N. BECK, J. N. KATZ, AND R. TUCKER, *Taking time seriously: Time-series-cross-section analysis with a binary dependent variable*, American Journal of Political Science, 42 (1998), pp. 1260–1288.

[6] E. BEYAZIT, J. ALAGURAJAH, AND X. WU, *Online learning from data streams with varying feature spaces*, Proceedings of the AAAI Conference on Artificial Intelligence, 33 (2019), pp. 3232–3239.

[7] E. BEYAZIT, M. HOSSEINI, A. MAIDA, AND X. WU, *Learning simplified decision boundaries from trapezoidal data streams*, in Artificial Neural Networks and Machine Learning – ICANN 2018, 2018, pp. 508–517.

[8] J. BLACKARD, *Covertype*.
UCI Machine Learning Repository, 1998.
DOI: https://doi.org/10.24432/C50K5N.

[9] S. BOYD AND L. VANDENBERGHE, *Convex Optimization*, Cambridge University Press, 2004.

[10] F. CAO, M. ESTERT, 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, SIAM, 2006, pp. 328–339.

[11] Z. CAO, S. ZHANG, AND C.-T. LIN, *Online ensemble of ensemble OVA framework for class evolution with dominant emerging classes*, in 2023 IEEE International Conference on Data Mining (ICDM), 2023, pp. 968–973.

[12] G. CASALINO, G. CASTELLANO, AND C. MENCAR, *Incremental adaptive semi-supervised fuzzy clustering for data stream classification*, in 2018 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), 2018, pp. 1–7.

[13] R. CATTRAL, F. OPPACHER, AND D. DEUGO, *Evolutionary data mining with automatic rule generalization*, Recent Advances in Computers, Computing and Communications, 1 (2002), pp. 296–300.

[14] J. CHEN, M. ABBOD, AND J.-S. SHIEH, *Integrations between autonomous systems and modern computing techniques: A mini review*, Sensors, 19 (2019).

[15] W. CHMIELNICKI AND K. STAÃSSPOR, *Combining one-versus-one and one-versus-all strategies to improve multiclass SVM classifier*, in Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015, R. Burduk, K. Jackowski, M. Kurzyński, M. Woźniak, and A. Żołnierek, eds., Cham, 2016, Springer International Publishing, pp. 37–45.

[16] K. CRAMMER, O. DEKEL, J. KESHET, S. SHALEV-SHWARTZ, AND Y. SINGER, *Online passive-aggressive algorithms*, Journal of Machine Learning Research, 7 (2006), pp. 551–585.

[17] E. R. DE FARIA, A. C. PONCE DE LEON FERREIRA CARVALHO, AND J. GAMA, *MINAS: multiclass learning algorithm for novelty detection in data streams*, Data Mining and Knowledge Discovery, 30 (2016), pp. 640–680.

[18] T. G. DIETTERICH, *Ensemble methods in machine learning*, in Multiple Classifier Systems, Berlin, Heidelberg, 2000, Springer Berlin Heidelberg, pp. 1–15.

[19] S. U. DIN AND J. SHAO, *Exploiting evolving micro-clusters for data stream classification with emerging class detection*, Information Sciences, 507 (2020), pp. 404–420.

[20] S. U. DIN, J. SHAO, J. KUMAR, C. B. MAWULI, S. M. H. MAHMUD, W. ZHANG, AND Q. YANG, *Data stream classification with novel class detection: A review, comparison and challenges*, Knowledge and Information Systems, 63 (2021), pp. 2231–2276.

[21] S. U. DIN, A. ULLAH, C. B. MAWULI, Q. YANG, AND J. SHAO, *A reliable adaptive prototype-based learning for evolving data streams with limited labels*, Information Processing & Management, 61 (2024), p. 103532.

[22] G. DITZLER, G. ROSEN, AND R. POLIKAR, *Incremental learning of new classes from unbalanced data*, in The 2013 International Joint Conference on Neural Networks (IJCNN), 2013, pp. 1–8.

[23] J. DONG, Y. CONG, G. SUN, T. ZHANG, X. TANG, AND X. XU, *Evolving metric learning for incremental and decremental features*, IEEE Transactions on Circuits and Systems for Video Technology, 32 (2022), pp. 2290–2302.

[24] D. FONTANEL, F. CERMELLI, M. MANCINI, AND B. CAPUTO, *On the challenges of open world recognition under shifting visual domains*, IEEE Robotics and Automation Letters, 6 (2021), pp. 604–611.

[25] J. GAMA, R. SEBASTIÃO, AND P. P. RODRIGUES, *On evaluating stream learning algorithms*, Machine Learning, 90 (2013), pp. 317–346.

[26] Y. GAO, S. CHANDRA, Y. LI, L. KHAN, AND T. BHAVANI, *SACCOS: A semi-supervised framework for emerging class detection and concept drift adaption over data streams*, IEEE Transactions on Knowledge and Data Engineering, 34 (2022), pp. 1416–1426.

[27] N. A. GHANI, S. HAMID, I. A. TARGIO HASHEM, AND E. AHMED, *Social media big data analytics: A survey*, Computers in Human Behavior, 101 (2019), pp. 417–428.

[28] J. GIBBONS AND S. CHAKRABORTI, *Nonparametric Statistical Inference: Revised and Expanded*, CRC Press, 2014.

[29] J. D. GIBBONS AND S. CHAKRABORTI, *Nonparametric Statistical Inference*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 977–979.

[30] H. M. GOMES, J. P. BARDDAL, F. ENEMBRECK, AND A. BIFET, *A survey on ensemble learning for data stream classification*, ACM Comput. Surv., 50 (2017).

[31] S. GU, T. LUO, M. HE, AND C. HOU, *Online learning with incremental feature space and bandit feedback*, IEEE Transactions on Knowledge and Data Engineering, 35 (2023), pp. 12902–12916.

[32] S. GU, Y. QIAN, AND C. HOU, *Incremental feature spaces learning with label scarcity*, ACM Trans. Knowl. Discov. Data, 16 (2022).

[33] M. HAHSLER AND M. BOLAÑOS, *Clustering data streams based on shared density between micro-clusters*, IEEE Transactions on Knowledge and Data Engineering, 28 (2016), pp. 1449–1461.

[34] M. HAN, Z. CHEN, M. LI, H. WU, AND X. ZHANG, *A survey of active and passive concept drift handling methods*, Computational Intelligence, 38 (2022), pp. 1492–1535.

[35] A. HAQUE, L. KHAN, AND M. BARON, *Semi supervised adaptive framework for classifying evolving data stream*, in Advances in Knowledge Discovery and Data Mining, 2015, pp. 383–394.

[36] ——, *SAND: Semi-supervised adaptive novel class detection and classification over data stream*, Proceedings of the AAAI Conference on Artificial Intelligence, 30 (2016).

[37] A. HAQUE, L. KHAN, M. BARON, B. THURAISINGHAM, AND C. AGGARWAL, *Efficient handling of concept drift and concept evolution over stream data*, in

2016 IEEE 32nd International Conference on Data Engineering (ICDE), 2016, pp. 481–492.

[38] S. HASHEMI, Y. YANG, Z. MIRZAMOMEN, AND M. KANGAVARI, *Adapted one-versus-all decision trees for data stream classification*, IEEE Transactions on Knowledge and Data Engineering, 21 (2009), pp. 624–637.

[39] Y. HE, C. SCHRECKENBERGER, H. STUCKENSCHMIDT, AND X. WU, *Towards utilitarian online learning: A review of online algorithms in open feature space*, in Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23, 2023.

[40] Y. HE, B. WU, D. WU, E. BEYAZIT, S. CHEN, AND X. WU, *Online learning from capricious data streams: A generative approach*, in Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI'19, 2019, pp. 2491–2497.

[41] Y. HE, B. WU, D. WU, E. BEYAZIT, S. CHEN, AND X. WU, *Toward mining capricious data streams: A generative approach*, IEEE Transactions on Neural Networks and Learning Systems, 32 (2021), pp. 1228–1240.

[42] Y. HE, X. YUAN, S. CHEN, AND X. WU, *Online learning in variable feature spaces under incomplete supervision*, Proceedings of the AAAI Conference on Artificial Intelligence, 35 (2021), pp. 4106–4114.

[43] S. C. HOI, D. SAHOO, J. LU, AND P. ZHAO, *Online learning: A comprehensive survey*, Neurocomputing, 459 (2021), pp. 249–289.

[44] B.-J. HOU, L. ZHANG, AND Z.-H. ZHOU, *Learning with feature evolvable streams*, in Advances in Neural Information Processing Systems, vol. 30, 2017.

[45] B.-J. HOU, L. ZHANG, AND Z.-H. ZHOU, *Prediction with unpredictable feature evolution*, IEEE Transactions on Neural Networks and Learning Systems, 33 (2022), pp. 5706–5715.

[46] M. JABREEL AND A. MORENO, *A deep learning-based approach for multi-label emotion classification in tweets*, Applied Sciences, 9 (2019).

[47] J. KIVINEN, A. SMOLA, AND R. WILLIAMSON, *Online learning with kernels*, IEEE Transactions on Signal Processing, 52 (2004), pp. 2165–2176.

[48] B. KRAWCZYK, L. L. MINKU, J. GAMA, J. STEFANOWSKI, AND M. WOŹNIAK, *Ensemble learning for data stream analysis: A survey*, Information Fusion, 37 (2017), pp. 132–156.

[49] Y. LECUN, C. CORTES, AND C. BURGES, *Mnist handwritten digit database*, ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, 2 (2010).

[50] R. LI, S. WANG, H. DENG, R. WANG, AND K. C.-C. CHANG, *Towards social user profiling: unified and discriminative influence model for inferring home locations*, in KDD, 2012, pp. 1023–1031.

[51] X. LI, Y. ZHOU, Z. JIN, P. YU, AND S. ZHOU, *A classification and novel class detection algorithm for concept drift data stream based on the cohesiveness and separation index of mahalanobis distance*, Journal of Electrical and Computer Engineering, 2020 (2020), p. 4027423.

[52] H. LIAN, D. WU, B.-J. HOU, J. WU, AND Y. HE, *Online learning from evolving feature spaces with deep variational models*, IEEE Transactions on Knowledge and Data Engineering, 36 (2024), pp. 4144–4162.

[53] P. LINDSTROM, S. J. DELANY, AND B. MAC NAMEE, *Handling concept drift in a text data stream constrained by high labelling cost*, in Twenty-Third International FLAIRS Conference, 2010, pp. 1–6.

[54] Y. LIU, X. FAN, W. LI, AND Y. GAO, *Online passive-aggressive active learning for trapezoidal data streams*, IEEE Transactions on Neural Networks and Learning Systems, 34 (2023), pp. 6725–6739.

[55] 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, 31 (2019), pp. 2346–2363.

[56] P. K. MALLAPRAGADA, R. JIN, AND A. JAIN, *Non-parametric mixture models for clustering*, in Structural, Syntactic, and Statistical Pattern Recognition, E. R. Hancock, R. C. Wilson, T. Windeatt, I. Ulusoy, and F. Escolano, eds., Berlin, Heidelberg, 2010, Springer Berlin Heidelberg, pp. 334–343.

[57] M. M. MASUD, T. M. AL-KHATEEB, L. KHAN, C. AGGARWAL, J. GAO, J. HAN, AND B. THURAISINGHAM, *Detecting recurring and novel classes in concept-drifting data streams*, in 2011 IEEE 11th International Conference on Data Mining, 2011, pp. 1176–1181.

[58] M. M. MASUD, Q. CHEN, J. GAO, L. KHAN, J. HAN, AND B. THURAISINGHAM, *Classification and novel class detection of data streams in a dynamic feature space*, in Proceedings of the 2010th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II, ECMLPKDD'10, 2010, pp. 337–352.

[59] M. M. MASUD, Q. CHEN, L. KHAN, C. C. AGGARWAL, J. GAO, J. HAN, A. SRIVASTAVA, AND N. C. OZA, *Classification and adaptive novel class detection*

of feature-evolving data streams, IEEE Transactions on Knowledge and Data Engineering, 25 (2013), pp. 1484–1497.

[60] M. M. MASUD, J. GAO, L. KHAN, J. HAN, AND B. THURAISINGHAM, *Classification and novel class detection in data streams with active mining*, in Advances in Knowledge Discovery and Data Mining, 2010, pp. 311–324.

[61] M. MOHAMMADI AND A. AL-FUQAHA, *Enabling cognitive smart cities using big data and machine learning: Approaches and challenges*, IEEE Communications Magazine, 56 (2018), pp. 94–101.

[62] M. MOHAMMADI, A. AL-FUQAHA, S. SOROUR, AND M. GUIZANI, *Deep learning for IoT big data and streaming analytics: A survey*, IEEE Communications Surveys & Tutorials, 20 (2018), pp. 2923–2960.

[63] N. MOUSTAFA AND J. SLAY, *The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set*, Information Security Journal: A Global Perspective, 25 (2016), pp. 18–31.

[64] X. MU, K. M. TING, AND Z.-H. ZHOU, *Classification under streaming emerging new classes: A solution using completely-random trees*, IEEE Transactions on Knowledge and Data Engineering, 29 (2017), pp. 1605–1618.

[65] A. M. MUSTAFA, G. AYOADE, K. AL-NAAMI, L. KHAN, K. W. HAMLEN, B. THURAISINGHAM, AND F. ARAUJO, *Unsupervised deep embedding for novel class detection over data stream*, in 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 1830–1839.

[66] T. T. NGUYEN, T. T. T. NGUYEN, A. V. LUONG, Q. V. H. NGUYEN, A. W.-C. LIEW, AND B. STANTIC, *Multi-label classification via label correlation and first order feature dependance in a data stream*, Pattern Recognition, 90 (2019), pp. 35–51.

[67] N. C. OZA AND S. J. RUSSELL, *Online bagging and boosting*, in Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics, T. S. Richardson and T. S. Jaakkola, eds., vol. R3 of Proceedings of Machine Learning Research, PMLR, 04–07 Jan 2001, pp. 229–236.

[68] J. PENG, J. GUO, Q. YANG, J. LU, AND J. SHAO, *A general framework for mining concept-drifting data streams with evolvable features*, in 2021 IEEE International Conference on Data Mining (ICDM), 2021, pp. 1276–1281.

[69] R. RIFKIN AND A. KLAUTAU, *In defense of one-vs-all classification*, J. Mach. Learn. Res., 5 (2004), pp. 101–141.

[70] S. J. RUSSELL AND P. NORVIG, eds., *Artificial Intelligence: A Modern Approach*, Prentice-Hall, 1995.

[71] H. SAHLI, *An Introduction to Machine Learning*, John Wiley & Sons, Ltd, 2020, ch. 7, pp. 61–74.

[72] D. SLATE, *Letter Recognition*.
UCI Machine Learning Repository, 1991.
DOI: https://doi.org/10.24432/C5ZP40.

[73] L. SONG AND L. L. MINKU, *A procedure to continuously evaluate predictive performance of just-in-time software defect prediction models during software development*, IEEE Transactions on Software Engineering, 49 (2023), pp. 646–666.

[74] A. SRINIVASAN, *Statlog (Landsat Satellite)*.
UCI Machine Learning Repository, 1993.

DOI: https://doi.org/10.24432/C55887.

[75] Y. SUN, K. TANG, L. L. MINKU, S. WANG, AND X. YAO, *Online ensemble learning of data streams with gradually evolved classes*, IEEE Transactions on Knowledge and Data Engineering, 28 (2016), pp. 1532–1545.

[76] M. TAVALLAEE, E. BAGHERI, W. LU, AND A. A. GHORBANI, *A detailed analysis of the KDD CUP 99 data set*, in 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.

[77] J. WANG, P. ZHAO, S. C. HOI, AND R. JIN, *Online feature selection and its applications*, IEEE Transactions on Knowledge and Data Engineering, 26 (2014), pp. 698–710.

[78] S. WANG, L. L. MINKU, AND X. YAO, *A learning framework for online class imbalance learning*, in 2013 IEEE Symposium on Computational Intelligence and Ensemble Learning (CIEL), 2013, pp. 36–45.

[79] S. WANG, L. L. MINKU, AND X. YAO, *Online class imbalance learning and its applications in fault detection*, International Journal of Computational Intelligence and Applications, 12 (2013), p. 1340001.

[80] S. WANG, L. L. MINKU, AND X. YAO, *Resampling-based ensemble methods for online class imbalance learning*, IEEE Transactions on Knowledge and Data Engineering, 27 (2015), pp. 1356–1368.

[81] 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, 29 (2018), pp. 4802–4821.

[82] Z. WANG, Z. KONG, S. CHANGRA, H. TAO, AND L. KHAN, *Robust high dimensional stream classification with novel class detection*, in 2019 IEEE 35th International Conference on Data Engineering (ICDE), 2019, pp. 1418–1429.

[83] J. XIE, Z. SONG, Y. LI, Y. ZHANG, H. YU, J. ZHAN, Z. MA, Y. QIAO, J. ZHANG, AND J. GUO, *A survey on machine learning-based mobile big data analysis: Challenges and applications*, Wireless Communications and Mobile Computing, 2018 (2018), p. 8738613.

[84] X. XU, I. W. TSANG, AND D. XU, *Soft margin multiple kernel learning*, IEEE Transactions on Neural Networks and Learning Systems, 24 (2013), pp. 749–761.

[85] D. YOU, J. XIAO, Y. WANG, H. YAN, D. WU, Z. CHEN, L. SHEN, AND X. WU, *Online learning from incomplete and imbalanced data streams*, IEEE Transactions on Knowledge and Data Engineering, 35 (2023), pp. 10650–10665.

[86] D. YOU, H. YAN, J. XIAO, Z. CHEN, D. WU, L. SHEN, AND X. WU, *Online learning for data streams with incomplete features and labels*, IEEE Transactions on Knowledge and Data Engineering, 36 (2024), pp. 4820–4834.

[87] P. ZAREMOODI, S. KAMALI SIAHROUDI, AND H. BEIGY, *Concept-evolution detection in non-stationary data streams: A fuzzy clustering approach*, Knowledge and Information Systems, 60 (2019), pp. 1329–1352.

[88] M.-L. ZHANG AND Z.-H. ZHOU, *A review on multi-label learning algorithms*, IEEE Transactions on Knowledge and Data Engineering, 26 (2014), pp. 1819–1837.

[89] Q. ZHANG, P. ZHANG, G. LONG, W. DING, C. ZHANG, AND X. WU, *Towards mining trapezoidal data streams*, in 2015 IEEE International Conference on Data Mining, 2015, pp. 1111–1116.

[90] Q. ZHANG, P. ZHANG, G. LONG, W. DING, C. ZHANG, AND X. WU, *Online learning from trapezoidal data streams*, IEEE Transactions on Knowledge and Data Engineering, 28 (2016), pp. 2709–2723.

[91] Z. ZHANG, Y. LI, Z. ZHANG, C. JIN, AND M. GAO, *Adaptive matrix sketching and clustering for semisupervised incremental learning*, IEEE Signal Processing Letters, 25 (2018), pp. 1069–1073.

[92] Z.-Y. ZHANG, P. ZHAO, Y. JIANG, AND Z.-H. ZHOU, *Learning with feature and distribution evolvable streams*, in Proceedings of the 37th International Conference on Machine Learning, vol. 119 of Proceedings of Machine Learning Research, 2020, pp. 11317–11327.

[93] X. ZHENG, P. LI, X. HU, AND K. YU, *Semi-supervised classification on data streams with recurring concept drift and concept evolution*, Knowledge-Based Systems, 215 (2021), p. 106749.

[94] D.-W. ZHOU, Y. YANG, AND D.-C. ZHAN, *Learning to classify with incremental new class*, IEEE Transactions on Neural Networks and Learning Systems, 33 (2022), pp. 2429–2443.

[95] Z.-H. ZHOU, *A brief introduction to weakly supervised learning*, National Science Review, 5 (2017), pp. 44–53.

[96] Y. ZOU, X. HU, AND P. LI, *Online multi-label classification with scalable margin robust to missing labels*, in 2023 IEEE International Conference on Knowledge Graph (ICKG), 2023, pp. 274–281.