UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Adaptive Learning under Concept Drift for Multiple Data Streams

by En Yu

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy in Computer Science

under the supervision of Prof. Jie Lu and Guangquan Zhang

University of Technology Sydney
Faculty of Engineering and Information Technology

April 2024

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, *En Yu*, 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.

Production Note:
Signature removed prior to publication.

SIGNATURE: _____

[En Yu]

DATE: April, 2020

PLACE: Sydney, Australia

Abstract

In real-world intelligent systems, it is common to encounter scenarios where multiple data streams are generated simultaneously. Each of these data streams is prone to changes in its underlying distribution, leading to concept drift. Despite being associated with the same task, these data streams often exhibit distinct distributions due to varying data sources. Hence, there's a rising interest in developing efficient learning techniques for analyzing multiple data streams with concept drift in non-stationary environments. This research focuses on two main tasks under multiple data streams: i) multistream classification, which aims to predict the unlabeled target stream by adaptively transferring knowledge from labeled source streams in the face of non-stationary processes with concept drifts. ii) exploring the concept drift problem in more complex intelligent systems.

For task i), current works reveal three issues in multistream classification: scarcity of labels, covariate shift, and asynchronous drift. Despite the growing research outcomes in this area, there has been a notable oversight regarding the following challenges: 1) how to model the dependency among data streams and exploit the positive information to help decision-making; 2) how to exploit robust and guaranteed joint representations of different streams 3) how to deal with real-world high-dimensional data efficiently; 4) how to handle the class-changing problem. To tackle these challenges, four innovative adaptive learning algorithms are proposed to provide solutions for each of the identified challenges. Specifically, Challenge 1) is solved by developing an *Online Boosting Adaptive*

Learning (OBAL) method. OBAL aligns the covariate shift as well as investigates a new dynamic correlation issue between source and target streams, which enhances positive knowledge transfer and prevents negative transfer effects. To solve Challenge 2), a *Fuzzy Shared Representation Learning* (FSRL) method is proposed to learn robust and compact common features for different streams. FSRL not only addresses the drifting problem but also mitigates the impact of redundant features and uncertainties. A *Learn-to-Adapt framework* (L2A) is designed to address the challenge 3). L2A contributes to dealing with more complex and multiple high-dimensional data streams by exploiting an efficient deep neural network. The challenge 4) is addressed by a *Calibrated Source-Free Adaptation* (CSFA) method, which exploits a calibrated prototype classifier to overcome the class incremental problem as well as adapting the distribution shift by a source-free adaptation strategy.

For task ii), there are many practical scenarios where concept drift issues exist. For example, autonomous vehicles need to navigate in dynamically changing environments. Hence, 5) expanding the real-world applications of the concept drift problem and designing autonomous learning algorithms to address it have also become imperative demands. Thus, this research further investigates the impact of adverse weather conditions or day-night transitions on visual restoration tasks. And, a transformer-based model is designed to adaptively adjust the restoration capabilities of the system in dynamic environments.

To conclude, this thesis contributes effective adaptive learning methods for multi-stream classification, thereby enhancing the performance and adaptability of multi-stream classification systems. In addition, it explores concept drift in complex real-world applications, advancing adaptive learning in non-stationary environments.

Dedication

To myself . . .

Acknowledgments

It is a memorial and exciting journey at University of Technology Sydney (UTS) for pursuing my Ph.D. degree in the past three and half years. I extend my sincerest thanks to all those who have contributed to help and inspire me, with immense gratitude and heartfelt appreciation.

First and foremost, I extend my sincerest appreciation to my principal supervisor, Distinguished Professor Jie Lu, for her unwavering support and guidance throughout my Ph.D. journey. Professor Lu's patience, encouragement, and mentorship have been instrumental in my academic growth and success. Her trust in my abilities and dedication to nurturing my research interests have empowered me to explore and innovate. Professor Lu's profound insights and vast expertise have continually motivated me to delve further into my research pursuits. Her guidance and insightful perspectives have provided invaluable clarity during moments of uncertainty. Moreover, her unwavering confidence and infectious enthusiasm have motivated me to persevere through challenges and adversities. I am deeply honored to have been mentored by such a distinguished researcher and compassionate supervisor. The knowledge acquired and the insights gained during her guidance have enriched my Ph.D. journey, serving as invaluable assets that will undoubtedly endure throughout my lifetime.

Meanwhile, I express my sincere appreciation to my co-supervisors, Professor Guangquan Zhang, Dr. Yiliao Song, and Dr. Kaihao Zhang. They have played an integral role in guiding me from being a novice researcher to becoming a proficient one. They consistently

provided me with the right direction in research and shared invaluable knowledge and experiences, which have been crucial in shaping my research skills. Each discussion with them has offered me new perspectives and insights, significantly enhancing my productivity and broadening my research horizons. Their guidance has instilled confidence in me and has been a source of hope during challenging times, both academically and personally.

I also want to express my sincere appreciation to all the individuals in the Decision Systems & e-Service Intelligence (DeSI) lab at the Australian Artificial Intelligence Institute (AAII) for their consistent support, camaraderie, and stimulating intellectual discussions. It has been a remarkable experience to spend my Ph.D. journey alongside these dedicated researchers and friends. In particular, I am grateful to Dr. Bin Zhang, Dr. Anjin Liu, Dr. Keqiuyin Li, Dr. Kun Wang, Zihe Liu, Wei Duan, Wenting Zhang, Mengjing Wu, Pengqian Lu, Ming Zhou, and Guangzhi Ma for their invaluable support and assistance in both my research and personal life. Their steadfast support and companionship have served as a constant source of motivation and inspiration.

Finally, I want to express my sincere gratitude to my wife, parents, and all my family members. Their unwavering love, encouragement, and sacrifices have served as my anchor and source of strength throughout this journey. Their belief in me has propelled me forward, and I am forever grateful for their enduring support.

PUBLICATIONS

1. En Yu, Yiliao Song, Guangquan Zhang and Jie Lu. "Learn-to-adapt: Concept drift adaptation for hybrid multiple streams". Neurocomputing 496 (2022): 121-130. DOI: 10.1016/j.neucom.2022.05.025

2. En Yu, Jie Lu, Bin Zhang and Guangquan Zhang. "Online Boosting Adaptive Learning under Concept Drift for Multistream Classification". Proceedings of the AAAI Conference on Artificial Intelligence. 2024, 38(15): 16522-16530. DOI: 10.1609/aaai.v38i15.29590

3. En Yu, Jie Lu and Guangquan Zhang. "Fuzzy Shared Representation Learning for Multistream Classification". Accepted by IEEE Transactions on Fuzzy Systems. DOI: 10.1109/TFUZZ.2024.3423024.

4. En Yu, Jie Lu and Guangquan Zhang. "Generalized Incremental Learning under Concept Drift via Calibrated Source-Free Adaptation". IEEE Transactions on Cybernetics, Under Review.

5. En Yu, Jie Lu, Kaihao Zhang and Guangquan Zhang. "Enhancing Outdoor Vision: Binocular Desnowing with Dual-Stream Temporal Transformer". Pattern Recognition, Under Review.

Table of Contents

List of Figures

Figure **Page**

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background and Motivations

In various real-world scenarios, such as auto-driving systems, weather forecasts, and industrial production, data is continuously and sequentially generated over time, which is referred as data streams or streaming data Lu et al. (2018a); Zhou et al. (2023c); Wang et al. (2022d). These data streams are susceptible to changes in their underlying distribution, resulting in concept drift. Consequently, classifiers trained on historical data may fail to predict subsequent samples, leading to a performance decrease Li et al. (2022); Xu et al. (2023). Researchers have shown significant interest in devising effective learning techniques to analyze streaming data in non-stationary environments. The objective of this research is to address the challenges posed by concept drift and enhance the model's adaptability to the continuously evolving data distributions in dynamic real-world environments. The phenomena of concept drift manifest in numerous real-world scenarios, for example, the stock market exhibits instability due to changing external conditions as shown in Figure 1.1(a), e.g., different economic conditions, regulations, or

trading behaviors. In retail, Figure 1.1(b) shows that product demand and sales patterns sometimes shift over time, resulting in distribution changes due to unexpected buying behaviors or economic fluctuations, and incorporating new products or opening new stores in a retail chain also introduces new data distributions.

(a) Movement of the Dow Jones Industrial Average (DJIA) between 01/2017 and 12/2020, showing the pre-crash high on 12/02/2020, and the subsequent crash during the COVID-19 pandemic and recovery to new highs to close 2020.

(b) US Holiday Season Retail Sales Growth, 2002-2019 Articles.

Figure 1.1: Examples of real-world applications under concept drift.

Under the circumstance of concept drift occurring, the prediction or decision outcomes of the model trained via offline learning strategy will be destroyed. In other words, traditional machine learning is processed by a lot of collected data in an offline mode, and then the learned model can be utilized to get the predictions for unseen inputs from the distribution as same as training data directly or via simple fine-tuning. Nevertheless, making decisions under concept drift for streaming data using a traditional learning strategy is not practical due to changing distributions and data forms. Therefore, it requires that the predictive models should have the ability to detect drift in time and adapt to newly arriving data over time, meanwhile, the models should also be robust to noise and be suitable to memory and time-consuming. As shown in Figure 1.2, assuming given the input stream data X, the prediction model can be defined as $y = L(X)$. The learning procedure under concept drift can be summarized as: (1) Prediction: When

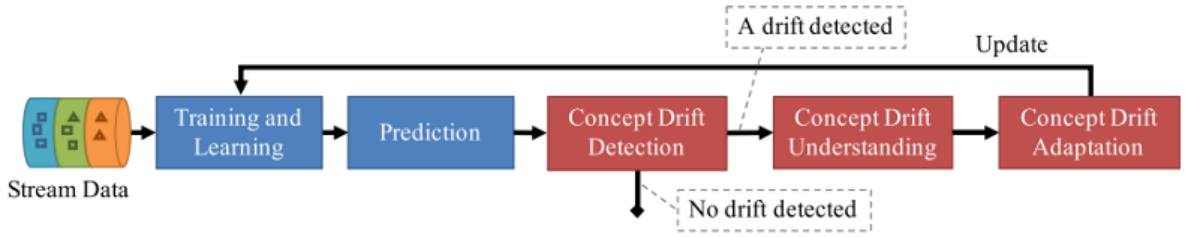


Figure 1.2: The framework of adaptive learning under concept drift Lu et al. (2018a).

a new sample X_t comes, we can get the prediction $\hat{y_t}$ using the current model L_t; (2) Concept Drift Detection: When we get the true label y_t after some time, the detection methods should be utilized to determine whether the drift happens or not; (3) Concept Drift Adaptation: If the drift happens, a thorough analysis of the drift is conducted (i.e., concept drift understanding), and the model should be updated to get the L_{t+1} using the newly collected data.

Existing studies have provided empirical evidence regarding the efficacy of concept drift adaptation methods for handling dynamic distributions in data streams Jiao et al. (2022a); Liu et al. (2020a); Miyaguchi and Kajino (2019); Hinder et al. (2023); Bai et al. (2023). Research in this domain predominantly falls into two categories: informed and blind. Informed methods typically leverage supervised or unsupervised drift detection strategies to dynamically monitor data streams. Upon detecting a drift, an adaptation mechanism will be triggered, enabling the model to quickly adapt to the new concept Abadifard et al. (2023); Gama and Castillo (2006). In contrast, blind methods continuously update the model with new incoming data without explicitly employing drift detection mechanisms Yang et al. (2021); Renchunzi and Pratama (2022). It's important to note, however, that a considerable number of these approaches are specifically designed for single-stream scenarios with delayed labels.

In practice, data are always generated from multiple non-stationary processes simultaneously Chandra et al. (2016); Yu et al. (2024a). These data streams, despite being

associated with the same task, often exhibit distinct distributions due to varying data sources Zhou et al. (2023b). For example, in healthcare systems, various clinical sites generate distinct data streams, which often exhibit underlying correlations. Training models using data collected from multiple clinics, rather than relying solely on smaller, single-stream data from each clinic, can significantly enhance healthcare decision-making processes Zhang et al. (2022a). This approach can enable the development of more robust and generalized models that effectively capture the complexities and nuances present in diverse clinical settings, thereby promoting improved healthcare outcomes and patient care. In addition, while data collection is straightforward, the labeling process incurs high time and labor costs, leading to the hybrid multiple streams where massive labeled and unlabeled streams arrive simultaneously Yu et al. (2022a). For instance, social media (like TikTok) always handles more complex multiple data streams (like images or videos) from different users Yue et al. (2019) Ji et al. (2016), each user generating one data stream. Real-time sentiment prediction in social media requires users' true sentiment to train the model. However, some users are not willing to provide the ground-truth. Therefore, the predicting method needs to handle labeled and unlabeled streams at the same time. When using existing single-stream aimed methods to deal with such multi-stream tasks, they cannot predict the unlabeled streams because they only handle each stream with delayed labels separately Song et al. (2020).

To address this research gap, numerous studies have been proposed to analyze and explore the correlation of multiple data streams, aiming to develop more general and robust predictive algorithms Yu et al. (2023). Among these, the most prominent task is multi-stream classification Haque et al. (2017); Yu et al. (2024a); Pratama et al. (2019). This task focuses on developing algorithms capable of effectively classifying data or predicting outcomes based on information extracted from multiple interconnected data streams. By leveraging the interdependencies among diverse streams, multistream classification

methods offer valuable insights and enhanced predictive capabilities, contributing to advancements in various areas, including healthcare, finance, auto-driving, and so on. Specifically, this task involves both labeled and unlabeled data streams generated in a non-stationary process, which are named source and target streams, respectively. It aims at predicting the class labels of target instances with the label information in the source stream Haque et al. (2017); Chandra et al. (2016). In other words, it requires that a model can be flexibly transferred from labeled source streams to the unlabeled target stream while employing online detection and adaptation working principles. This not only enables the model to adapt to new and unlabeled data streams but also mitigates the expenses and logistical challenges.

The multistream classification problem features three major challenges that have to be tackled simultaneously Haque et al. (2017); Yoon et al. (2022): *a) Scarcity of labels:* this arises from the absence of labels specifically for the target stream, while the source streams possess labeled data; *b) Covariate shift:* this implies that any two data streams exhibit distinct distributions, whether they are different source streams or a source stream and a target stream; and *c) Asynchronous drift:* the source and target streams are susceptible to independent concept drift, which occurs at varying time periods and results in unique effects on the model performance. Although some works have focused on the analysis and study of multistream classification, there still are several unsolved and challenging problems, i.e., 1) how to model the dependency among data streams and exploit the positive information to help decision-making; 2) how to exploit guaranteed joint representations of different streams; 3) how to develop guaranteed and efficient adaptation methods to deal with real-world high-dimensional data; 4) how to handle the class-changing problem with the data streams evolving.

In addition to the multistream classification task, there are many practical scenarios where concept drift issues exist. For example, autonomous vehicles need to navigate in

dynamically changing environments, such as bad weather conditions, obstacles, emergency events, and other vehicles breaking down. Hence, 5) expanding the real-world applications of the concept drift problem and designing autonomous learning algorithms to address it have also become imperative demands.

This study aims to give a comprehensive analysis and solutions to all the above-mentioned challenges. Chapter 3 targets challenge 1); Chapter 4 provides a solid solution for challenges 2); Chapter 5 and 6 aim to address challenge 3) and 4), respectively; and Chapter 7 widens the concept drift problem into a real-world scenario to investigate challenge 5).

1.2 Research Questions and Objectives

1.2.1 Research Questions

The concept drift adaptation problem for multiple streams is still an open question with numerous unresolved challenges. Specifically, this study focuses on the following research questions:

RESEARCH QUESTION 1 (RQ1): *How to model the temporal dynamic correlations and exploit the positive information to help decision-making?*

Multistream classification aims to transfer the knowledge from labeled source to unlabeled target streams while employing online detection and adaptation working principles, and it poses three main challenges: scarcity of labels, covariate shift, and asynchronous drift. However, any drift occurring within each stream has the potential to alter the correlation between the source and target streams. There has been a notable oversight regarding the temporal dynamic relationships between these streams, leading to the issue of negative transfer arising from irrelevant data. It is crucial for the predictive model to adapt promptly, extracting valuable insights from relevant source streams while

avoiding the assimilation of irrelevant knowledge from other source streams.

RESEARCH QUESTION 2 (RQ2) *How to exploit robust and guaranteed joint representations of different streams?*

Transferring knowledge among multiple non-stationary data streams presents additional challenges that are not considered in current works. Firstly, most current methodologies construct classifiers based on information from the original feature space, overlooking the impact of redundant or low-quality features, which can detrimentally affect the final decision-making process. Therefore, a method to exploit guaranteed joint representations of different streams becomes particularly crucial during knowledge transfer. Secondly, various data streams with asynchronous drifts inevitably contain inherent uncertainties. Consequently, a method to address these uncertainties and provide a reasonable interpretability analysis is also a crucial focus of this task.

RESEARCH QUESTION 3 (RQ3): *How to develop guaranteed and efficient adaptation methods to deal with real-world high-dimensional data?*

Current drift adaptation methods often rely on hand-crafted features, which are typically low-dimensional and may not be suitable for handling complex data. However, as the demand for more sophisticated solutions increases, there is a pressing need to shift focus towards addressing concept drift in high-dimensional data streams. In many situations, it requires that the model can continuously process vast amounts of visual data captured by cameras. In this context, traditional machine learning methods based on hand-crafted features may not fully capture the complexity of the visual environment with dynamics. Therefore, developing deep learning-based concept drift adaptation methods that are capable of handling high-dimensional data is crucial for ensuring the robustness and reliability of more complex systems in real-world scenarios. These methods should be able to automatically extract relevant features from raw data, adapt to evolving environments, and effectively deal with the emergence of new concepts and

challenges over time.

RESEARCH QUESTION 4 (RQ4): *How to make concept drift adaption towards open world scenarios, including both distribution changing and class involving problems?*

Existing multistream concept drift adaptation methods typically assume that the streams share a common label space and are designed to recognize a limited number of classes. However, real-world applications often involve streaming data with the arrival of new classes over time. For example, consider a scenario where a robot must continually learn about new objects while also retaining knowledge of those it has encountered previously. The ability to continuously acquire and accumulate new knowledge, including new distributions and classes, is a key characteristic of general intelligence. Therefore, addressing the challenge of concept drift in multi-stream environments requires methods that can adapt to evolving data distributions and accommodate the emergence of new classes in real-time applications.

RESEARCH QUESTION 5 (RQ5): *How to widen the real-world applications of the concept drift problem and design autonomous learning algorithms to solve it?*

In addition to the multistream classification task, there are many practical scenarios where concept drift issues exist. For example, autonomous vehicles need to navigate in dynamically changing environments, such as bad weather conditions, obstacles, emergency events, and other vehicles breaking down. Hence, expanding the real-world applications of the concept drift problem and designing autonomous learning algorithms to address it have also become imperative demands. Consider a scenario where an autonomous vehicle encounters bad weather conditions or day-night altering, which significantly affects visibility and road conditions. In such situations, the vehicle's perception system must quickly adapt to the changing environment, accurately detect obstacles, and adjust its navigation strategy to ensure safe operation. Thus, the expansion of real-world applications of concept drift adaptation extends beyond traditional domains, encompassing

areas such as autonomous vehicles, finance, healthcare, and environmental monitoring. Addressing these challenges requires the development of autonomous learning algorithms capable of continuously learning from evolving data streams and adapting to changing conditions in real time. These algorithms must exhibit robustness, scalability, and adaptability to ensure their effectiveness in diverse and dynamic environments.

1.2.2 Research Objectives

To answer these research questions, we set up the corresponding Research Objectives (RO) as follows:

RESEARCH OBJECTIVE 1 (RO1): *To provide a comprehensive definition and devise an adaptive algorithm that enhances positive knowledge transfer by exploiting dynamic correlations for multistream classification.* (Aims to answer RQ1)

Existing concept drift adaptation works focus on single or multiple streams with delayed labels. In this research, we give an expansive definition of more hybrid drifting problems in multiple data streams. Compared to these existing settings, it places fewer limitations on data streams—streams can be stationary (non-drifting) or non-stationary (drifting), and streams can be labeled or unlabeled. There is a significant gap in multistream classification, where the dynamic relationships between streams have largely been overlooked. This oversight can often result in the issue of negative transfer stemming from irrelevant data. To overcome the issue, we propose a novel online boosting adaptive learning method that effectively addresses this limitation by adaptively learning the dynamic correlation among different streams.

RESEARCH OBJECTIVE 2 (RO2): *To develop a robust and efficient joint representation learning method that guarantees enhanced performance while effectively handling uncertainties.*

In contemporary multistream classification approaches, the process of drift detection

and adaptation predominantly operates within the confines of the original feature space. Nonetheless, the inclusion of redundant information and uncertainties substantially exerts influence on the outcome, also undermining the efficiency of the methodology. Since fuzzy systems have shown robust learning capabilities and transparent interpretability for knowledge transfer, this research proposes a novel fuzzy shared representation learning method to learn compact high-quality features. When a shared fuzzy space is established, we can map any newly arrived sample into this space using the learned transformations, and then proceed with detection and adaptation. This method efficiently circumvents the detrimental effects caused by redundant or low-quality features.

RESEARCH OBJECTIVE 3 (RO3): *To explore more complex multistream classification problems with high-dimensional data and develop a deep learning-based adaptation method.*

Current concept drift adaptation methods often rely on hand-crafted features, which are typically low-dimensional. In addition, these methods often employ fine-tuning or retraining of classifiers or predictors to adapt to concept drift. Nevertheless, it's important to recognize that the learning and representation of features play a crucial role in the overall performance. Given the widespread success of deep neural networks across diverse areas, they offer a promising avenue for seamlessly integrating feature learning and final prediction within a unified framework. Deep neural networks have the capability to learn hierarchical representations of data, allowing them to automatically extract relevant features directly from the raw input. In light of these advancements, this research proposes a novel deep learning-based concept drift adaptation method. This method not only focuses on adapting the classifiers to changing concepts but also integrates the feature extractor and classifiers into a feedback loop. By doing so, our approach enables the simultaneous refinement of both feature extraction and classification processes, resulting in a more robust and adaptive model for handling

concept drift in dynamic environments.

RESEARCH OBJECTIVE 4 (RO4): *To develop a method to address both distribution-changing and class-incremental problems.* (Aims to answer RQ3)

Data streams often encounter dynamic environments, exhibiting characteristics of both distributional shifts and class evolution. In traditional drift adaptation scenarios, instances are typically associated with single or multiple labels. However, previous research has predominantly assumed a static class space throughout the learning process, disregarding the continuous evolution of classes in response to changing environmental conditions. In practice, as data exploration deepens, the number of classes tends to increase gradually. To bridge these research gaps, this study introduces a novel framework that encompasses both distribution-changing and class-incremental challenges. This framework aims to address the dynamic nature of data streams more comprehensively. Specifically, we propose a stable few-shot class incremental learning method to effectively handle the class-incremental problem. Moreover, to tackle the distribution shift, we introduce a test domain adaptation strategy to adapt to changing distributions while maintaining robust performance. This holistic approach offers a more nuanced understanding of the complexities inherent in multiple data streams, paving the way for more effective and adaptive learning algorithms in dynamic environments.

RESEARCH OBJECTIVE 5 (RO5): *To investigate the concept drift problem in more complex intelligent systems and design adaptive frameworks to overcome drifting scenarios.* (Aims to answer RQ5)

In addition to multistream classification, our research delves deeper into the phenomenon of concept drift in more complex intelligent tasks. Specifically, we investigate the impact of adverse weather conditions or day-night transitions on visual restoration tasks, focusing on their implications for autonomous driving systems. To address these challenges, we introduce a transformer-based model designed to adaptively adjust the

restoration capabilities of the system in dynamic environments. This approach aims to provide robust and high-quality data support for the ultimate intelligent decision-making system. Our research not only advances our understanding of concept drift in challenging environments but also offers practical solutions for enhancing the resilience and performance of intelligent systems in real-world applications.

1.3 Research Contributions

This thesis aims to present a comprehensive analysis of various drifting situations in multiple data streams and reveal the main challenges exposed in different drifting situations. Correspondingly, a series of adaptive learning algorithms are proposed to overcome these challenges. The main contributions of this study are summarised as follows:

1) Defining challenges in multistream classification.

Expansive definitions of multistream classification task and challenges are proposed to present research objectives as well as the characteristics of different drifting situations. Addressing these challenges can ensure accurate and reliable classification results in dynamic and complex multistream environments, providing clear direction for algorithm design and development.

2) A new online adaptive learning method to explore dynamic correlations.

- This study presents a new online ensemble approach (OBAL) for multi-source data stream classification. With the capability to dynamically detect and adapt to concept drift, OBAL demonstrates enhanced effectiveness and stability. Moreover, it offers effortless extensibility in managing diverse data streams.

- A novel algorithm (AdaCOSA) is proposed to align the covariate shift as well as investigate a new dynamic correlation issue between source and target streams.

It further enhances positive knowledge transfer and prevents negative transfer effects.

- It designs a simple yet effective GMM-based module to adapt the asynchronous drift. It orchestrates an ensemble of both historical classifiers and newly trained classifiers on weighted source samples. By accumulating abundant source knowledge, the proposed approach achieves improved prediction accuracy for the target stream.

3) A new method to learn robust common representations based on fuzzy systems.

- This study proposes a novel fuzzy shared representation learning (FSRL) method for multistream classification, addressing uncertainties and interpretability during knowledge transfer.
- An advanced optimization method for multistream joint distribution adaptation is introduced to learn the consequent parameters of the TSK fuzzy system. This addresses inter-stream shifts, simultaneously mitigating the impact of redundant features and enhancing model robustness.
- A window-based and a GMM-based online adaptation strategies are designed to address the asynchronous drifts over time. They provide a robust solution for asynchronous drift adaptation, and ensuring continuous model relevance in evolving data landscapes.

4) A new deep learning-based method to handle high-dimensional streaming data.

- This study proposes a learn-to-adapt framework (L2A) to address the challenges of multistream classification in an end-to-end way.

- The framework adapts knowledge from a labeled drifting stream to unlabeled target streams, but also contributes to dealing with more complex and multiple high-dimensional data streams.
- This study paves a new way to address the problem of concept drift in multiple streams while broadening the practical applications of meta-learning.

5) A new incremental method to overcome both class evolving and distribution changes.

- This study introduces a more practical setting, Generalized Incremental Learning under Concept Drift (GILCD), aimed at investigating the challenges posed by class incremental and concept drift in dynamically evolving environments.
- A novel framework, Calibrated Source-Free Adaptation (CSFA) is proposed to solve the GILCD problem. It employs training-free calibrated prototypes to incrementally learn novel classes with limited labeled source samples. Additionally, we propose a source-free adaptation method to address distribution shifts in target streams.
- During adaptation, a reliable source-free adaptation algorithm (RGSM) is proposed. It can filter samples with large entropy out of adaptation, ensuring the model converges to a flat region with enhanced adaptation and generalization capabilities.

6) Investigate the impact of concept drift in the visual restoration task.

- To further explore the impact of concept drift in various real-world applications, this study investigates the effects of adverse weather conditions or day-night transitions on visual restoration tasks. Experimental validation and analysis were conducted based on the desnow task.
- This study proposes to advance snow removal by considering the shared information from different views, which is more challenging and practical in computer vision.

To solve this, we design a novel snow removal framework, named Dual-Stream Temporal Transformer (DSTT), to extract enhanced spatial-temporal features across different views and contribute to recovering high-quality clean videos.

- This study further analyzes the performance of DSTT under drifting scenarios and proposes a novel adaptive DSTT model (AdaDSTT) to address the concept drift problem in the video content restoration task.

1.4 Research Significance

The theoretical and practical significance of this thesis is summarized as follows:

Theoretical significance: This research gives an expansive and standardized definition of the concept drift problem in multistream scenarios and develops a series of concept drift adaptation methods to handle the proposed research questions. The comprehensive definition supplements critical issues overlooked by the current multistream classification researches. It also provides direction for adaptive learning in drifting multistream scenarios. The proposed methods enrich the theoretical analysis of multistream classification and provide significant solutions for newly exposed challenges. This research explores the dynamic correlations among various non-stationary streams and gives insight to enhance positive knowledge transfer. Furthermore, it reveals robust and compact shared representations can improve learning efficiency and model stability by integrating Fuzzy systems. This research also marks the first indication that employing meta-learning techniques can train deeper neural networks to achieve rapid adaptation to concept drift. Meanwhile, it also offers research directions for addressing the issue of class evolution in multistream classification tasks. In addition, this study further explores and substantiates the concept drift issue in increasingly complex scenarios, providing a foundation for future research endeavors.

Practical signifcance: The findings of this thesis hold significance for real-world applications involving the concurrent generation of data streams from multiple non-stationary processes. Through this study, a suite of adaptive learning methods has been developed to effectively address the myriad challenges present in handling multiple data streams, thereby enhancing adaptation efficiency and prediction accuracy. The methods devised in this thesis undergo rigorous validation using real-world datasets and tasks. Experimental results unequivocally demonstrate the superior performance of the proposed method compared to the majority of existing approaches. These findings are pivotal in resolving real-world challenges pertaining to online decision-making for multiple data streams. Moreover, this research has broader implications, as it lays the groundwork for addressing similar challenges in a multitude of online applications.

1.5 Thesis Structure

The structure of the thesis is shown in Figure 1.3 and the chapters are organized as follows:

- CHAPTER 2: This chapter provides a comprehensive literature review related to this research. It begins by introducing the fundamental concepts and types of concept drift. Subsequently, it delineates the basic processes and mainstream methods for concept drift detection and adaptation. Furthermore, it offers a special focus on mainstream approaches for concept drift adaptation in the context of multiple streams. Finally, it covers various related research domains such as online transfer learning, meta-learning, class incremental learning, and test-time adaptation which are relevant to this thesis.

- CHAPTER 3: This chapter proposes a novel online boosting adaptive learning (OBAL) method that can adaptively learn the dynamic correlation among different

streams. Specifically, OBAL operates in a dual-phase mechanism, in the first of which we design an adaptive covariate shift adaptation (AdaCOSA) algorithm to construct an initialized ensemble model using archived data from various source streams, thus mitigating the covariate shift while learning the dynamic correlations via an adaptive re-weighting strategy. During the online process, we employ a Gaussian Mixture Model-based weighting mechanism, which is seamlessly integrated with the acquired correlations via AdaCOSA to effectively handle asynchronous drift. This chapter addresses RQ1 to achieve RO1.

- CHAPTER 4: This chapter proposes an interpretable Fuzzy Shared Representation Learning (FSRL) method based on the Takagi-Sugeno-Kang (TSK) fuzzy system. FSRL accomplishes the non-linear transformation of individual streams by learning the fuzzy mapping with the antecedents of the TSK fuzzy system, thereby effectively preserving discriminative information for each original stream in an interpretable way. Then, a multistream joint distribution adaptation algorithm is proposed to optimize the consequent part of the TSK fuzzy system, which learns the final fuzzy shared representations for different streams. Hence, this method concurrently investigates both the commonalities across streams and the distinctive information within each stream. Following that, window-based and GMM-based online adaptation strategies are designed to address the asynchronous drifts over time. The former can directly demonstrate the effectiveness of FSRL in knowledge transfer across multiple streams, while the GMM-based method offers an informed way to overcome the asynchronous drift problem by integrating drift detection and adaptation. This chapter addresses RQ2 to achieve RO2.

- CHAPTER 5: This chapter discusses concept drift adaptation in multiple high-dimensional data streams. It rethinks the concept drift problem by adopting a meta-learning approach and introduces a learn-to-adapt framework (L2A). The

L2A framework simultaneously 1) makes adaptations for drifting labeled streams, and 2) leverages knowledge from labeled drifting streams to make adaptations for unlabeled stream prediction. In L2A, a meta-representor with an adapter in the meta-training stage is designed to learn the invariant representations for drifting streams, enabling the model to quickly produce a good generalization of new concepts with limited training samples. In the online stage, the meta-representor will be adapted continually under the control of the adapter and will contribute to adapting the classifiers for unlabeled drifting stream prediction. L2A adapts the feature extractor and classifier in a feedback process, which is advanced in dealing with more complex and high-dimensional data streams. This chapter addresses RQ3 to achieve RO3.

- CHAPTER 6: This chapter introduces a novel research setting called Generalized Incremental Learning under Concept Drift (GILCD) where both classes and distributions evolve over time. Our objective is to incrementally learn new class concepts while adapting to distribution shifts for accurate prediction. To tackle this challenge, this chapter proposes the novel Calibrated Source-Free Adaptation (CSFA) framework. It employs a training-free calibrated prototype strategy to incrementally learn new classes with limited labeled samples, enhancing prediction stability by fusing new class prototypes with base prototypes. This strategy requires no extra optimization procedures once the base training finished. In addition, this chapter proposes a Reliable Surrogate Gap Sharpness-aware Minimization (RGSM) algorithm to address the concept drift problem. RGSM minimizes perturbation loss and surrogate gap while integrating a reliable indicator function to filter out high-entropy samples. This approach aids in distribution alignment between source and target streams and improves generalization for changing target distributions. Therefore, CSFA enhances the robustness and adaptability of adaptive learning

under concept drift, ensuring effectiveness in dynamic data streams. This chapter addresses RQ4 to achieve RO4.

- CHAPTER 7: This chapter focuses on the task of video content restoration within the context of intelligent visual systems. For instance, in auto-driving systems, different cameras capture video streams from various views, generating multiple video streams. In outdoor environments, the collection of videos often results in background drifts inevitably, which can significantly impact the restoration of video content. One of the most noticeable drifts is the transition from day to night, which serves as a clear instance of concept drift. To verify this hypothesis, this chapter first designs a robust baseline for visual snow removal in a static environment, which is a critical research task in video content restoration. Subsequently, testing is conducted in dynamic environments with day-night transitions to validate the significant impact of such drifts on the model. Additionally, this chapter proposes a novel adaptive model to address the concept drift problem in video content restoration tasks. This chapter addresses RQ5 to achieve RO5.

- CHAPTER 8 summarises the findings of this thesis and points to directions for future work.

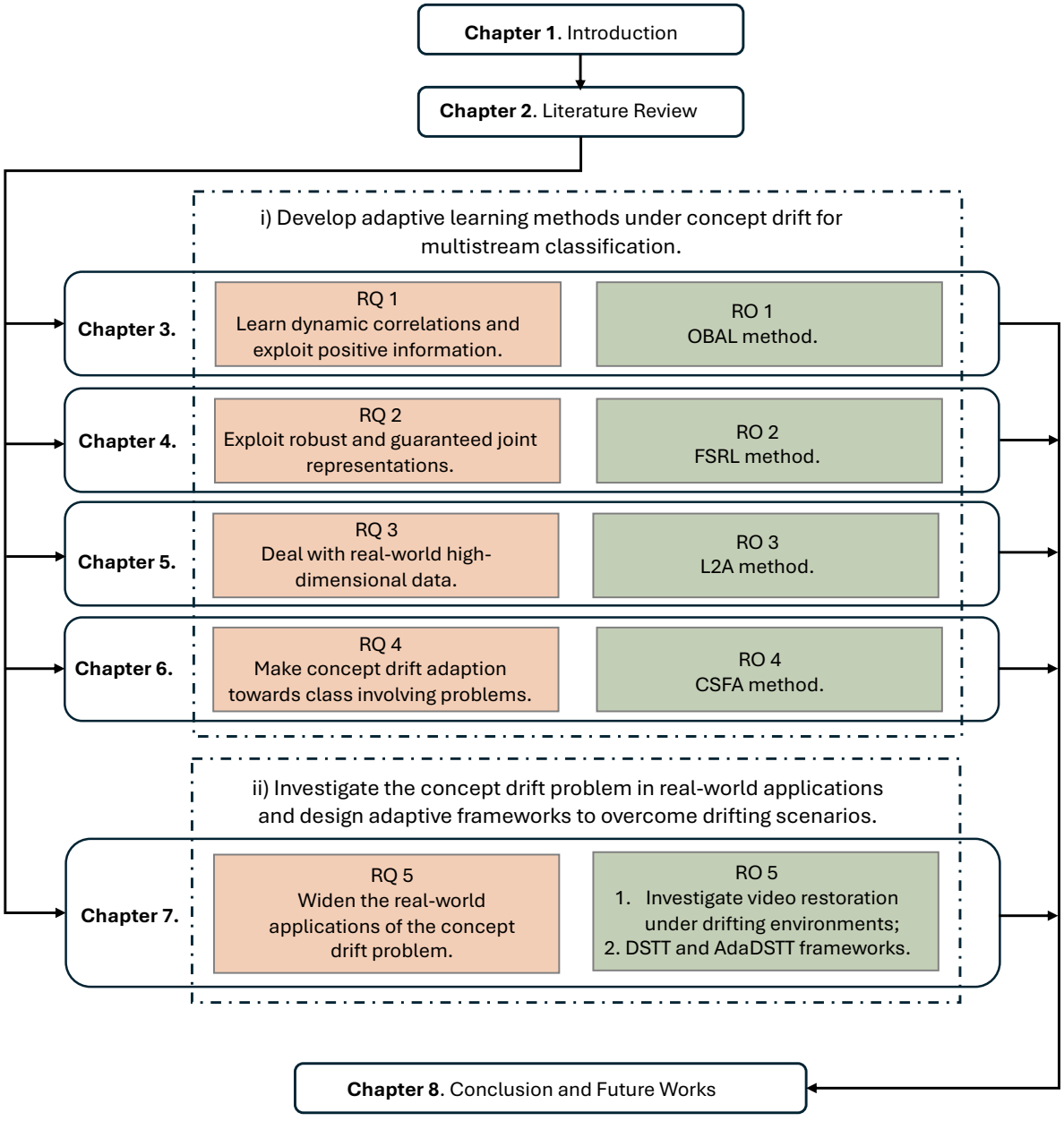


Figure 1.3: Thesis Structure.

CHAPTER 2

LITERATURE REVIEW

This chapter offers a comprehensive survey of the literature related to this research. As depicted in Figure 2.1, Section 2.1 delves into the formulation of the concept drift problem, encompassing its definition and various types. Subsequently, both concept drift detection and adaptation techniques are thoroughly introduced in Sections 2.2 and 2.3, respectively. Additionally, Section 2.4 provides an in-depth introduction to multistream classification along with associated methodologies. Finally, Section 2.5 reviews other works pertinent to this research, including meta-learning, class-incremental learning, test-time adaptation, and visual restoration methods.

2.1 Definitions of Concept Drift

Concept drift is a pervasive issue in numerous real-world scenarios, posing a threat to the predictive capabilities of models trained offline. In order to conduct a thorough investigation into this problem, this section presents fundamental definitions and terminologies of concept drift, along with an overview of common types of drift occurrences.

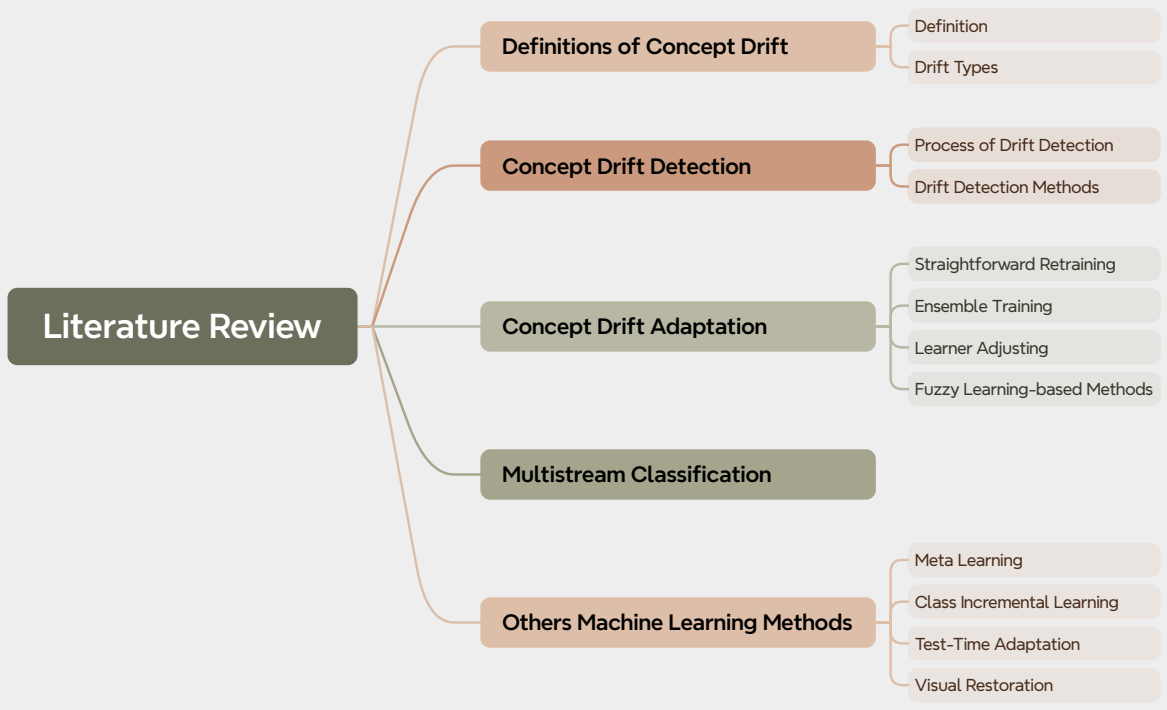


Figure 2.1: Overview of literature review.

2.1.1 Definition

Concept drift is referred to the change of underlying distributions of streaming data, also there are some different names or terminologies given by other authors, such as data shift Storkey (2009), covariate shift Huang et al. (2006), conditional change Gao et al. (2007) and so on. Here, we refer to the description in Lu et al. (2018a) and define the problem as:

Definition 2.1 ***Concept drift.*** *Given a time period from* 0 *to* t*, and data stream* $S_{0,t} = \{(X_{0,0}, y_{0,0}), ..., (X_{i,t}, y_{i,t}),\}$*, where* $X_{i,t}$ *is the data instance and* $y_{i,t}$ *is the label. Concept drift occurs at* $t+1$*, if joint distribution* $p_{t+1}(X, y) \neq p_t(X, y)$*.*

Specifically, the prior probabilities of classes $p(y)$ may change, and the conditional probabilities $p(X|y)$ may change, thus the posterior probabilities $p(y|X)$ will change, which affects the obtained model. As shown in Figure 2.2, we can also distinguish two

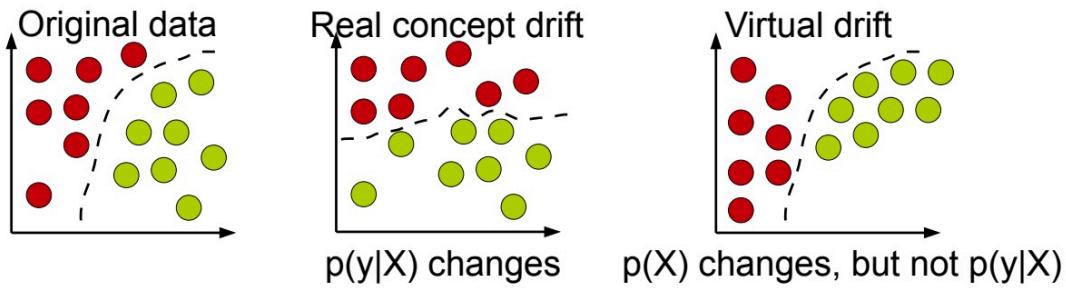


Figure 2.2: Overview of literature review.

categories of drifts according to the above definition: (1) Real Drift happens when $p(y|X)$ changes, while (2) Virtual Drift only changes the data distribution $p(X)$ without affecting the conditional probability $p(y|X)$.

2.1.2 Concept Drift Types

As introduced in Lu et al. (2018a), concept drift in single streams can be categorized into four types: (1) Sudden Drift: switching from one concept to another suddenly. (2) Incremental Drift: changing to the new concept slowly over a period of time. (3) Gradual Drift: the new concept replaces the old concept gradually over a period of time. (4) Reoccurring Drift: drifts may introduce new concepts that were not seen before, or previously seen concepts may reoccur after some time. Each drift type is illustrated in Figure 2.3.

2.2 Concept Drift Detection

2.2.1 Process of Drift Detection

Concept drift detection is the preparation work for the adaptive update, which plays an essential role in data streaming mining. It aims at discriminating when or where the data distribution changes. And referring to the analysis in Lu et al. (2018a), we regard

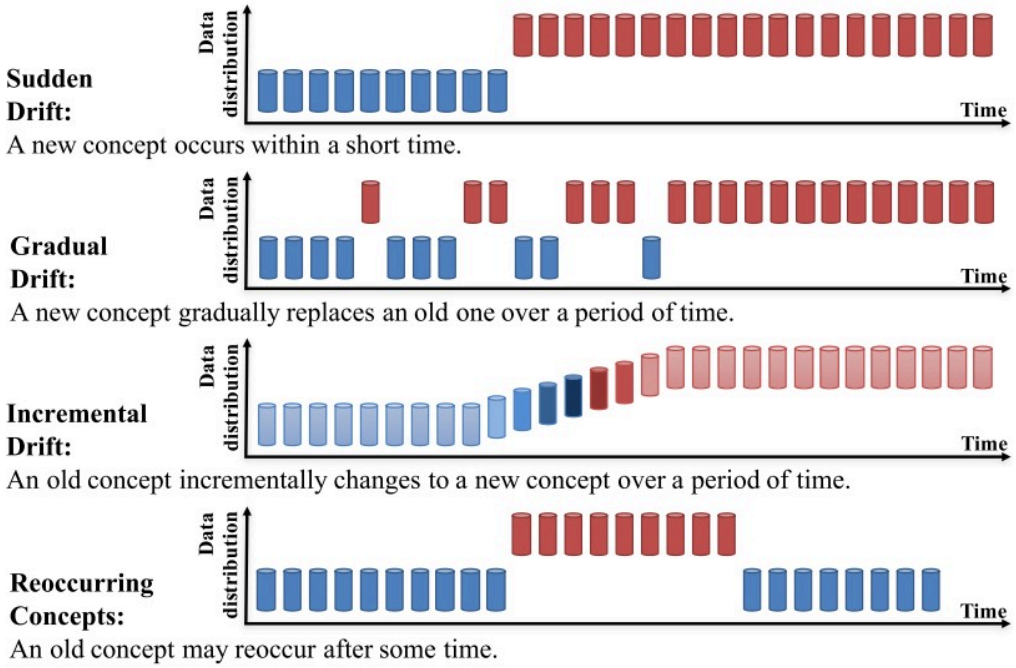


Figure 2.3: Illustration of concept drift types Lu et al. (2018a).

the general process of concept drift detection as three stages:

Stage 1 (Data Management): As we all know, a single sample may not be abundant with information to get the distribution shift. Therefore, one crucial operation of data management is data sampling, which means that we always presuppose a strategy to get the data chunks with fixed or adaptive window sizes and striding steps. Another optional operation is feature selection, which aims at selecting the key features with discriminative information that contribute to impacting the data pattern, such as active learning Settles (2009), dimensionality reduction Van Der Maaten et al. (2009) and so on. It is also used to meet requirements of limited storage and fast learning speed.

Stage 2 (Dissimilarity Measurement): Intuitively, when the data chunks are given over time, we should design an accurate and robust strategy to quantify the statistical difference between them. Several methods are commonly used to measure these differences: Distributional distance metrics Dasu et al. (2006) (e.g., Kullback-Leibler

Divergence) assess divergence between probability distributions across data chunks; Feature statistics based methods Jiao et al. (2022b) compare specific metrics like mean and variance; and Model-based methods Gama et al. (2004) involve training a supervised model (e.g., a classifier) to monitor changes in model performance over time. It is an important and challenging stage of drift detection, and also considered to be an open research area.

Stage 3 (Hypothesis Test): After stage 2, although the test statistics can be observed, they can not identify the confidence interval which means whether the change is concept drift or noise. Thus, some specific hypothesis tests should be used to evaluate the statistical attribute of the dissimilarity and then we can get the drift detection accuracy. And there are lots of hypothesis tests, such as distribution estimation, bootstrapping, bound identification based on Hoeffding's inequality, and so on.

2.2.2 Drift Detection Methods

Drift detection refers to the strategies and mechanisms for identifying the changing points and quantifying the severity. This setting defines the true values for the targets that will come with a delay or are not available at all. According to the difference between hypothesis tests, we overview the most representative drift detection methods from Onefold and Ensemble two categories.

2.2.2.1 Onefold Drift Detection

As mentioned above, if the true value of the target comes immediately or in some time, it is reasonable that the error rate can be applied for drift detection, which is the most common category of detection algorithms. These algorithms aim at monitoring the online error rate of existing classifiers. If the error rate changes dramatically and is statistically significant analyzed by stage 3 in Section 2.2.1, the classifier will be updated.

Drift Detection Method (DDM) Gama et al. (2004), one of the most representative drift detection methods, proposes two thresholds of confidence (i.e., warning level and drift level) to distinguish the drift according to the error rate. Specifically, DDM manages the data samples using a dynamic time window and calculates the error rate with the data coming. If the observed error rate increases dramatically and reaches the warning level thresholds, DDM still uses the old learner for prediction but it will also build a new model in reserve. When the change exceeds the thresholds of the drift level, the old learner will be dropped by the new learner for the next prediction.

There also are some similar methods extended from DDM. For example, Dynamic Extreme Learning Machine (DELM) Xu and Wang (2017) just replaces the base learner with a hidden layer neural network and maintains the same detection methods as DDM, which focuses on improving the adaptation performance. Fuzzy Windowing Drift Detection Method (FW-DDM) Liu et al. (2017) aims at improving the data management stage utilizing a fuzzy time window, which especially contributes to solving the gradual drift problem. Early Drift Detection Method (EDDM) Baena-Garcıa et al. (2006) improves the robustness of the drift detection by applying a new dissimilarity measurement between two correct predictions. Heoffding's inequality-based Drift Detection Method (HDDM) Frias-Blanco et al. (2014) applies Heoffding's inequality for the Hypothesis test, which is introduced in Section 2.2.1 Stage 3.

In contrast to the single window-based detection method, there are still some multiple window-based methods, such as Statistical Test of Equal Proportions Detection (STEPD) Nishida and Yamauchi (2007). STEPD designs two-time windows, i.e., an overall time window and a most recent time window, and it mainly depends on comparing the error rate that comes from these two-time windows to detect the drift according to the given threshold of warning and drift levels similar to DDM. In addition, ADaptive WINdowing (ADWIN) Bifet and Gavalda (2007) also uses two-time windows. Especially, different

from STEPD, ADWIN can get the optimal sub-window size adaptively. It is also the most popular method for the state-of-the-art adaptation methods to refer to.

Unsupervised drift detection refers to the distribution or density-based methods without using the true values of the target variables. This kind of method aims at calculating the distribution dissimilarity between the old data and coming data via some specific distance measurement methods. And then the update mechanism will be triggered when the drift is determined. Commonly, the distribution-based detection methods always require pre-defined time windows, thus two windows-based methods are always applied.

Kifer et al. (2004) firstly propose a criterion of distance measurement for distribution discrepancy analysis, and it is defined as: $TV(P_1, P_2) = 2\sup_{E\epsilon\varepsilon}|P_1(E) - P_2(E)|$. Equivalently, it can be re-defined as the density form: $\text{dist}_{L^1} = \int |f_1(x) - f_2(x)|\,\mathrm{d}x$. Another representative detection method based on distribution is the Information-Theoretic Approach (ITA) Dasu et al. (2006). It first divides the old and new data into a series of bins using kdqTress, and then applies the Kullback-Leibler divergence to measure the density of each bin, and finally the concept drift can be confirmed by the bootstrapping hypothesis test.

2.2.2.2 Ensemble Drift Detection

Different from the above methods based on a single hypothesis test, ensemble-based drift detection methods utilize multiple hypothesis tests. The ensemble mechanism can be further divided as bagging and boosting strategies.

Drift detection based on a bagging strategy means the parallel structure of hypothesis tests. Just-In-Time adaptive classification (JIT) Alippi and Roveri (2008a) uses this strategy for drift detection. Specifically, JIT first uses the Principal Component Analysis (PCA) to map the raw features into different feature spaces, and then detect the drift in all the combinations of feature spaces as well as the raw feature spaces. Similarly, Linear

Four Rate drift detection (LFR) Wang and Abraham (2015) detects the changes in True Positive rate (TP), False Positive rate (FP), True Negative rate (TN) and False Negative rate (FN). IV-Jac Zhang et al. (2017) integrates the Information Value and Jaccard similarity into a three-layer network for drift detection, which processes the problem with prior probability ($p(X)$), posterior probability ($p(y)$) and conditional probability ($p(y|X)$) jointly. Ensemble of Detectors (e-Detector) Du et al. (2015) proposes using heterogeneous detectors to observe the change and using the diversity measurement for drift warning, which means any of the base detectors find a drift, the system will update.

In addition, boosting strategy (Hierarchical) based drift detection always defines a normal detection layer and a second validation layer in a tandem way. The first layer is mainly for drift detection and the second layer is another hypothesis test for reduplicate detection. Hierarchical Change-Detection Tests (HCDTs) Alippi et al. (2016) is the first method for concept drift detection based on Hierarchical architecture. It indicates that we can use any base detectors for the detection layer, and then use the validation layer to test the result obtained from the detection layer. Inspired by HCDTs, Yu et al. Yu and Abraham (2017) propose the Hierarchical Linear Four Rate (HLFR) method. The detection layer of HLFR is the LRF and the validation layer is defined as a zero-one loss. If the zero-one loss is more than the given threshold, the drift will be confirmed and trigger the update progress.

There also are some request and reverify-based Hierarchical detection methods, such as, Hierarchical Hypothesis Testing with Classification Uncertainty (HHT-CU) Yu et al. (2018). HHT-CU utilizes Heoffding inequality in the detection layer to find the change of prediction, and similar to HLFR, the validation layer is defined as a zero-one loss. Besides, Yu et al. Yu et al. (2018) Hierarchical Hypothesis Testing with Attribute-wise Goodness-of-fit (HHT-AG). Kolmogorov-Smirnov (KS) based hypothesis test is used in the detection layer. Then, the true labels of new coming data are used to validate the

drift and it performs d independent 2-dimensional Kolmogorov-Smirnov test with each feature-label distribution, which only uses a few true labels and outperforms related supervised detection methods.

2.3 Concept Drift Adaptation

In this section, we focus on the techniques and mechanisms for drift adaptation, which means the process of updating existing learning models based on the drifts. Straightforward retraining, ensemble retraining, and learner adjusting are the three key categories of drift adaptation methods that attempt to manage various forms of drift. By the way, It is inevitable to introduce some detection methods at the same time due to the blurred boundary between concept drift detection and adaptation.

2.3.1 Straightforward Retraining

This is the most simple strategy that retraining a new model with the new coming data when the drift is observed. Thus, one of the most important steps is to detect when the model should be retrained. An intuitionistic strategy is just using a window-based detection method to manage the coming and old data. For example, Paired Learners Bach and Maloof (2008) proposes using two base learners, i.e., stable learner and reactive learner, to confirm whether the model should be retrained. In particular, upon identifying a new concept, if the stable learner consistently misclassifies instances that the reactive learner correctly identifies, the stable learner will be replaced by the reactive learner. This approach is straightforward to comprehend and can be easily extended to other problems in data streams.

However, the biggest issue of the window-based method is that we have to estimate the window size. If the window size is too large, it will cause memory overload, otherwise, a small window size can not get abundant information. Therefore, ADWIN Bifet and

Gavalda (2007) is proposed and it is not necessary to pre-define a fixed window size. Instead, it considers all potential window cuts and calculates the best sub-window sizes based on the rate of change among sub-windows. After that, a new model will be retrained using the data from the latest window and realize the model updating. Compared to direct retraining, some other methods integrate drift detection and adaptation mechanisms into an adaptive framework. For example, DELM Xu and Wang (2017) develops an adaptive hidden layer, in which the nodes can be adjusted according to whether the drift occurs or not. Similarly, the ELM extended method, FP-ELM Liu et al. (2016), integrates the forgetting mechanism into the adaptive model.

In addition, Instance-Based lazy learners have also been used broadly in recent years. For example, Just-In-Time adaptive classification (JIT) Alippi and Roveri (2008a,b) converts the standard CUSUM test into a format that does not require the pdf. After that, the detection method is combined with a kNN classifier. Old instances are excluded from the case base when a principle drift is observed Alippi and Roveri (2008b). To further identify and distinguish the drift instances and noise instances, Lu et al. Lu et al. (2014) also propose Stepwise Redundancy Removal (SRR), which reduces the redundant cases.

2.3.2 Ensemble Retraining

With the emergence and development of Ensemble Learning Dietterich et al. (2002); Polikar (2012), many researchers have also applied the mechanism of ensemble learning to the concept drift problem, especially for the recurring situation. Reoccurring drift refers to the changing of new concepts and old concepts alternatively, which requires the model should acquire new knowledge from the latest data samples for the new concept as well as preserve the ability of accurate prediction for the old concept. Ensemble learning-based methods include a group of base learners, which could be homogeneous or

heterogeneous, and the final predictions are decided by the ensemble of all base learners via some voting strategies Polikar (2012).

There are three main research points: the selection of base learners, ensemble patterns, and voting roles. Correspondingly, ensemble learning-based adaptive methods under concept drift also focus on improving or exploring new strategies to make the final decision from the above factors. Here, we mainly introduce the Ensemble Retraining methods that focus on the improvement of ensemble patterns and voting roles.

Boosting, Bagging, and Random Forests as the most representative architectures in ensemble learning are always used to make a better decision for addressing the concept drift problem in an unstable environment Gomes et al. (2017a). For example, Chu et al. Chu and Zaniolo (2004) propose an adaptive learning method based on boosting. It is an error-based method by a hypothesis test, which assumes that if there is no drift, the classification error should be Gaussian distribution. For the bagging extension, online bagging Oza and Russell (2001) is first developed and it replaces batch-based bagging with an instance-based strategy. Similarly, the proposed method in Bifet et al. (2010) further integrates ADWIN for drift detection. Recently, A random forest extended method, Adaptive Random Forest (ARF) Gomes et al. (2017b), is proposed to combine with drift detection algorithms to realize the appropriate retraining and a similar method using Hoeffding bound for noise identification is also proposed in Li et al. (2015).

Besides the above methods that focus on ensemble patterns, there still are many ensemble methods designed for concept drift problems by innovating voting rules. One of the most popular methods is proposed in Kolter and Maloof (2007), named Dynamic Weighted Majority (DWM). Specifically, DWM controls all the classifiers based on the accuracy of the classification individually and globally. Globally, if the ensemble can not classify the sample correctly, DWM will add a new classifier to the ensemble. Individually, if any base learner misclassifies an instance, the weight of the learner will be reduced and

even will be dropped when the weight drops below the predefined threshold. However, DWM is not so efficient for gradual drift due to the high-frequency new classifier training. In order to avoid this issue, Elwell et al. Elwell and Polikar (2011) propose the well-known Learn++NES, which just considers the error rate of the most recent data batch for base learner weighting with an incremental learning strategy. Furthermore, the Optimal Weights Adjustment (OWA) method Zhang et al. (2008) not only considers the base learner weighting but also considers the instance weighting.

2.3.3 Learner Adjusting

Intuitively, retraining a new model completely will cause heavy costs of time or memory, thus many current kinds of research focus on how to update the model partially. The most popular method is the Very Fast Decision Tree (VFDT) proposed in Domingos and Hulten (2000), which is an online decision tree method and spits the node by calculating the Hoeffding bound. Due to the outstanding attribute of the decision tree, VFDT just only can process the instance only once and does not need to store it in memory, which performs more efficiently. CVFDT Hulten et al. (2001), the extension of VFDT, utilizes a sliding window to save the new data and an optional sub-tree is learned using the data in this window. If the optional sub-tree performs better than its raw counterpart, it will be adopted for the next prediction and the old sub-tree will be pruned. VFDTc Gama et al. (2003) regards the naive Bayes as the classifiers in the tree and uses more attributes for detection and adaptation. In addition, compared with CVFDT, it uses the distribution-based detection mechanism instead of classification error.

2.3.4 Drift Adaptation Based on Fuzzy Techniques

Fuzzy systems have gained significant traction in machine learning research Deng et al. (2017); Lei et al. (2020); Yu et al. (2024b); Shi et al. (2024). Notably, the TSK fuzzy

system Takagi and Sugeno (1985) has emerged as a prominent model within the realm of fuzzy logic, distinguished by its data-driven approach and utilization of IF-THEN fuzzy rules Zhang et al. (2023). This methodology not only enables the construction of models adept at learning intricate data patterns but also excels in delivering transparent and interpretable results. The versatility of the TSK fuzzy system is evident from its successful application across diverse domains, spanning from control systems to pattern recognition and beyond. Leveraging its robust learning capabilities and transparent interpretability, fuzzy systems have recently found application in transfer learning research Lu et al. (2015); Li et al. (2023a), owing to their demonstrated effectiveness in knowledge transfer Lu et al. (2019); Li et al. (2023a); Xu et al. (2019).

Furthermore, fuzzy logic has been utilized to address the concept drift problem. For instance, Pratama et al. proposed an evolving recurrent fuzzy neural network in Pratama et al. (2016) to make adaptations incrementally. This method incorporates a generalized interval type-2 fuzzy rule, allowing for automatic generation, pruning, merging, and recall within a single-pass learning model. Additionally, Song et al. applied a fuzzy c-means clustering technique in Song et al. (2017, 2019) during the learning of a regression model. It can identify the most relevant data instances pertaining to the latest pattern in the training set. Fuzzy methods offer advantages in designing windowing techniques for concept drift adaptation, such as the fuzzy windowing technique proposed by Liu et al. (2017), which provides more flexible drift detection.

2.4 Multistream Classification

However, most methods are designed for a single-labeled stream, which can not be used for the multi-stream scenario. To fill the research blank, Chandra et al. Chandra et al. (2016) introduce a multi-stream classification framework that utilizes ensemble classifiers for each data stream and incorporates Kernel Mean Matching to reduce

the disparity between source and target streams. They further propose the FUSION algorithm Haque et al. (2017) to leverage the Kullback Leibler Importance Estimation Procedure for density ratio estimation and covariate shift handling. In addition, some neural-network-based models are proposed to deal with high-dimensional data Yoon et al. (2022). For example, Autonomous Transfer Learning (ATL) Pratama et al. (2019) is an online domain adaptation strategy that employs both generative and discriminative phases, combined with Kullback Leibler divergence-based optimization. Moreover, Yu et al. Yu et al. (2022b) propose a meta-learning-based framework to learn the invariant features of drifting data streams and then update the meta model in an online fashion.

In addition, multi-source stream classification is proposed to enhance the robustness by considering the complementary information from different source streams simultaneously. For example, Du et al. Du et al. (2019) introduced Melanie, which employs a weighted ensemble classifier to transfer knowledge from multiple source streams. It is the first approach capable of simultaneously transferring knowledge from various source streams with concept drift. However, Melanie is a supervised method, which cannot be used for unlabeled data prediction.

Hence, the AutOmatic Multi-Source Domain Adaptation (AOMSDA) Renchunzi and Pratama (2022) incorporates a central moment discrepancy-based regularizer to leverage the complementary information from multi-source streams, and employs a node weighting strategy to tackle the covariate shift. AOMSDA is a chunk-based method, which means it lacks the ability to dynamically detect the changes in data streams. To address this limitation, Jiao et al. Jiao et al. (2022b) propose a reduced-space Multi-stream Classification based on Multi-objective Optimization (MCMO). It seeks a common feature subset to minimize the distribution shift and then uses a GMM to detect and adapt asynchronous drift. However, all these methods determine the correlation between each individual source and target stream as fixed, which does not fully exploit temporal

dynamic correlations.

Adaptive learning for multiple streams under concept drift has attracted a lot of attention and many methods are emerging in recent years. The most representative adaptive framework for multistream classification is proposed in Chandra et al. (2016). It defines two independent data streams, one of which is the labeled data stream generated in a non-stationary process (named as source stream), and another is the unlabeled data stream from another dynamic process (named target stream). It aims at predicting the class labels of target instances with the label information in the source stream. Correspondingly, as depicted in this thesis, the drifts may occur at different times with different types in two streams, which causes unstable prediction errors. Therefore, the authors propose a weighted ensemble method for classification and using probability density ratio to eliminate the covariate shift. For drift detection, this method applies the supervised window and unsupervised window-based methods to analyze the source stream and target stream, respectively. In the later work, Ahsanul et al. Haque et al. (2017) also extended the basic adaptive framework to an online version, which reduces the execution overhead and enhances usability.

In addition, Ahsanul et al. Haque et al. (2018) propose a regression setting in a non-stationary data stream. Similarly, the setting defines a source stream with true response values and a target stream without any response value, which also may have asynchronous concept drifts. It integrates the direct density ratio estimation into an online regression model, which focuses on bias correction and asynchronous drift detection simultaneously. However, the above methods for multistream analysis assume that the data streams are independent. Yiliao et al. Song et al. (2020) propose a multistream regression model based on a fuzzy drift variance matrix (FDA), which is used for measuring the correlated drifts among all streams.

2.5 Other Machine Learning Methods

2.5.1 Meta-learning

Meta-learning shows promising performance in its fast adaptation ability to new tasks with limited data samples Hospedales et al. (2020); Yao et al. (2021). It gains accumulated knowledge by training the models over multiple learning tasks and improves future performance for new tasks. This 'learning-to-learn' method also has various benefits such as data and compute efficiency. For instance, MAML Finn et al. (2017) trains a model on a variety of sub-tasks to obtain the corresponding prior and then it can adapt the benefit of the prior to new tasks with limited training samples. However, the sampled tasks in meta-learning are assumed to be from the same distribution, which is not applicable in a changing distribution environment.

Inspired by meta-learning, Liu et al. (2020c) proposes to make adaptation in an online and continually evolving environment. An Evolution Adaptive Meta-Learning (EAML) framework is proposed to solve this problem without forgetting. However, it requires adequate source labels for meta-train and it focuses on learning a meta-model without forgetting (In other words, it only works for incremental drift) Yang et al. (2021). Therefore, it is not realistic for streaming learning due to the limited amount of labels and it will also cause the old knowledge to be overwritten when facing gradual or reoccurring drifts.

In addition, the most recent work OSAKA Caccia et al. (2020) also proposes a task-incremental setting, where previous tasks may reoccur and new tasks will also appear. In order to solve the OSAKA problem, the authors propose Continual-MAML for online adaptation based on the learning-to-learn strategy. However, the essential distinction between OSAKA and *HMS-CDA* is that the new arriving data will get delayed labels in OSAKA while there may not be labels in our proposed setting. Therefore, our proposed

setting is more challenging and the Continual-MAML is not suitable.

More recently, Marcus de Carvalho et al. (2021) proposed ACDC, an online adversarial unsupervised domain adaptation framework that handles two data streams with a self-evolving neural network structure. It can make adaptation for multiple unlabeled data streams under the distribution-changing environment, especially for real-world high-dimensional data. However, there are many limitations compared to our research. Firstly, from the perspective of research questions, ACDC mainly focuses on handling the marginal probability distribution, independent asynchronous drift, and contrasting throughput problems, which only is one of our research objectives. Our research is more expansive and challenging because we aim to address the dependency of drifting situations, label evolving, and drift understanding.

2.5.2 Class Incremental Learning

Class-incremental learning (CIL) endeavors to construct a comprehensive classifier encompassing all encountered classes over time Masana et al. (2022); Zhou et al. (2023a). The primary challenge in CIL, known as catastrophic forgetting, arises when optimizing the network with new classes leads to the loss of knowledge pertaining to previous classes, resulting in irreversible performance degradation. Thus, to effectively mitigate the catastrophic forgetting problem, there are three major categories: replay-based methods Gu et al. (2022); Isele and Cosgun (2018), knowledge distillation Rebuffi et al. (2017); Kang et al. (2022), and model expansion Wang et al. (2024a, 2022a). Replay-based approaches offer an intuitive means of leveraging previous data for rehearsal, enabling the model to revisit former classes and resist forgetting. Alternatively, some methodologies incorporate regularization terms with additional data to guide optimization direction and mitigate catastrophic forgetting. Knowledge distillation-based CIL methods aim to establish mappings between old and new models, thereby preserving the character-

istics of the old model during the updating process. Additionally, recent studies have demonstrated the effectiveness of model expansion in CIL Yan et al. (2021b). A notable method involves preserving a single backbone and freezing it for each incremental task, effectively alleviating the catastrophic forgetting problem.

CIL methods often depend on abundant labeled data for supervised learning, presenting a significant challenge, especially in real-world data streams where runtime and labeled data for model updates are limited. In response to this practical concern, a more realistic incremental learning paradigm, Few-Shot Class-Incremental Learning (FSCIL), has been introduced. FSCIL aims to efficiently tackle the class-incremental learning problem with only limited labeled data available. This paradigm has demonstrated effectiveness in addressing the challenges of class-incremental learning under resource constraints Wang et al. (2024a); Zhang et al. (2021a). For instance, Tao et al. Tao et al. (2020) propose a neural gas network to preserve the topology of features in both base and new classes for the FSCIL task. In Zhang et al. (2021a), they introduce a continually evolved classifier for few-shot incremental learning, utilizing an adaptation module to update classifier weights based on a global context of all sessions. Furthermore, Wang et al. Wang et al. (2024a) propose a simple yet effective training-free prototype calibration strategy (TEEN) for biased prototypes of new classes. However, all these methods are designed for static scenarios and do not address the problem of distribution changes.

2.5.3 Test-Time Adaptation

Test-Time adaptation (TTA), a vital aspect of machine learning, has garnered significant attention in recent years due to its potential to enhance model performance in real-world non-stationary scenarios Liang et al. (2023); Wang et al. (2024b). A similar paradigm is source-free domain adaptation (SFDA), which also requires no access to the training (source) data Liang et al. (2020). This technique involves adapting a trained model

during inference time to better suit the characteristics of the current data distribution. Various approaches have been proposed to achieve Test-Time Adaptation across different domains.

Current TTA methods can be categorized as Test-Time Training (TTT) and Fully Test-Time Adaptation based on whether they alter the training process. TTT methods jointly optimize a source model with both supervised and self-supervised losses, such as contrastive-based objectives Chen et al. (2020a), rotation prediction Gidaris et al. (2018), and so on. It then conducts self-supervised learning at test time to fit the new test distributions Sun et al. (2020). Fully Test-Time Adaptation methods do not modify the training process and can be applied to any pre-trained model. These techniques involve adapting various aspects such as statistics in batch normalization layers Zhao et al. (2022); Wang et al. (2020), unsupervised entropy minimization Zhang et al. (2022d); Niu et al. (2023), prediction consistency maximization Chen et al. (2022), top-k classification boosting Niu et al. (2022) and so on.

2.5.4 Visual Restoration

In this thesis, we also focus on binocular video desnowing in dynamic environments, which is a completely new research task. So far, only a few studies have examined how to remove snow from videos. However, rain removal has been researched more thoroughly and since it is closely related to snow removal, we have also included some of the current rain removal techniques. In addition, the review covers a few highly relevant studies on image restoration using Transformer models.

Snow Removal: Current snow removal methods can be divided into two main categories, i.e., traditional methods and deep learning methods. Traditional snow removal methods are based on hand-crafted features. For example, Bossu et al. (2011) outlined how to use the Histogram of Orientations of Streaks (HOS) based on the Histogram of

Oriented Gradient (HOG) features for snow removal. This technique starts with a classic background subtraction method, where a Mixture of Gaussians (MoG) is used to separate the background from the foreground. The snowflakes are then detected and removed from the foreground according to the obtained HOS. Alternatively, Xu et al. (2012) proposed an approach where a refined guidance image is calculated by employing color information and a guided filter is used to remove the snow. Rajderkar and Mohod (2013) introduced a frequency decomposition method. Their technique decomposes the snow image into low-frequency and high-frequency parts using a smoothing filter. However, all these approaches focus on designing hand-crafted features, *e.g.*, HOG Dalal and Triggs (2005), color assumption Chen et al. (2014), or frequency space decomposition Li et al. (2016), which often results in a weak generalization ability.

More recently, methods based on deep learning have shown promise with snow removal tasks due to their powerful representation abilities and the fact that they can incorporate feedback mechanisms. Liu *et al.* Liu et al. (2018) propose the first snow removal framework based on deep learning. Called DesnowNet, the framework consists of a translucency recovery module and a residual generation module to remove opaque snow particles and recover the image details. Another recent proposal is the Multi-scale Stacked Densely Connected Convolutional Network (MS-SDN) Zhang et al. (2021b), which simultaneously detects then removes the snowflakes from an image. Lastly, the all-in-one bad weather removal method based on a framework called the Network Architecture Search (NAS) also includes a snow removal process Li et al. (2020).

Rain Removal: Removing rain from images or videos, which is similar to snow removal, is a relatively explored task in the computer vision area Patil et al. (2022). However, snowflakes are opaque and they create different shapes as they fall making snow removal a more challenging problem than rain removal. In past years, many hand-crafted deraining methods have been proposed to recover clean images. For example,

Kang et al. (2011) developed a technique for detecting rain that relies on frequency space decomposition, where rain streaks are removed from high-frequency channels via sparse representations. Luo et al. (2015) propose a method that also involves separating the rain layer and deraining the image with an algorithm based on dictionary learning.

As with many fields, advancements in deep learning have also benefitted rain removal with several CNN/RNN-based models showing promise Huang and Zhang (2022); Gao et al. (2021). For instance, Fu et al. (2017) proposed a CNN-based model named DerainNet to remove rain particles. In this framework, the model directly learns a mapping relationship between the rainy image and the clean image using a convolutional neural network given high-frequency content. Luo et al. (2015) use an attention mechanism to learn depth-attentional features, regressing a residual map based on the attention weights to remove rain streaks. Similarly, some studies also use an attention mechanism to learn regions of heavy rain to improve deraining performance Li et al. (2019); Zhang and Patel (2018). However, all these methods only consider single images, which can not perform well with videos or temporal information.

To consider both temporal and spatial information in videos, Chen et al. (2018) use superpixel (SP) segmentation to separate the scene into depth-consistent units. They then align the content at the SP level, noting that this approach is robust to camera motion. Liu et al. (2019) develop D3RNet, which aggregates the temporal information extracted from the recurrent units with the spatial features extracted by a residual network. Zhang et al. (2022b) design a SICM module based on ConvLSTM to extract temporal information, which helps to improve the quality of the deraining. To our best knowledge, Zhang et al. (2022c) is the only team to explore deraining with binocular videos. In this work, they propose PRRNet for stereo image deraining, which leverages semantic information as well as the visual deviation between the two views for rain removal.

Vision Transformer: Recently, Transformer-based models have shown great success and gained much attention in the computer vision area, particularly for tasks like image classification Dosovitskiy et al. (2020), object detection Zhu et al. (2021) and semantic segmentation Zheng et al. (2021). Transformers typically have a strong ability to explore long-range interactions between different visual regions. For example, Wu *et al.*'s architecture uses ResNet as a backbone but replaces the last convolutional stage with a vision Transformer Wu et al. (2020). Dosovitskiy et al. (2020) designed a pure Transformer, named Vision Transformer (ViT) that directly applies Transformer architecture to the sequences of image patches for image classification tasks. In addition, there are plenty of studies that augment a conventional Transformer block or a self-attention layer with convolution simply because convolution performs so well at extracting local information Han et al. (2022). Along these lines, Chu et al. (2021) proposed a conditional positional encoding (CPE) method that combines the convolutional operations for fine-level feature encoding.

In addition to the above high-level tasks, Transformer has been investigated for some low-level tasks. Image restoration methods developed a pre-trained Transformer model (IPT) for a range of image restoration tasks, such as denoising, super-resolution, and deraining Chen et al. (2021). Liang et al. (2021) put forward a Swin Transformer-based image restoration model called SwinIR. Wang et al. (2022e) contributed a U-shape framework (Uformer) for image restoration that not only handles local contexts but also efficiently captures long-range dependencies. However, as image restoration methods, these techniques do not consider temporal information. With regard to video restoration, Cao et al. (2021) proposed a model called video super-resolution or VSR, which is based on a Transformer model.

CHAPTER 3

ONLINE BOOSTING ADAPTIVE LEARNING FOR MULTISTREAM CLASSIFICATION

RQ1 mentions that there has been a notable oversight regarding the temporal dynamic relationships between various streams, leading to the issue of negative transfer arising from irrelevant data. To solve this problem and achieve RO1, this chapter proposes a novel **O**nline **B**oosting **A**daptive **L**earning (OBAL) method that effectively addresses this limitation by adaptively learning the dynamic correlation among different streams. In addition, it employs a Gaussian Mixture Model-based weighting mechanism, which is seamlessly integrated with the acquired correlations to effectively handle asynchronous drift. In this chapter, the detailed analysis of motivations and challenges is introduced in Section 3.1. Definitions, notations, and proposed OBAL are listed and explained in Section 3.2. The proposed method is validated by experimental evaluations in Section 3.3. Section 3.4 concludes this chapter.

3.1 Introduction

In various real-world scenarios, such as auto-driving systems, weather forecasts, and industrial production, data is continuously and sequentially generated over time, which is referred to as data streams or streaming data Lu et al. (2018a); Zhou et al. (2023c); Wang et al. (2022d). These data streams are susceptible to changes in their underlying distribution, resulting in concept drift. Consequently, classifiers trained on historical data may fail to predict subsequent samples, leading to a performance decrease Li et al. (2022); Xu et al. (2023). Thus, it attracts many researchers to develop efficient learning techniques capable of analyzing streaming data with concept drift in non-stationary environments. To date, prior studies have provided empirical evidence of the efficacy of concept drift adaptation methods in effectively addressing data streams with dynamic distributions. It is worth noting that the majority of existing techniques have been tailored specifically for a single stream with delayed labels Yu et al. (2022b); Song et al. (2021b). However, it is common to encounter scenarios where multiple data streams are generated simultaneously in real-world intelligent systems. For example, data samples continuously stream from sensors in manufacturing systems. These data streams, despite being associated with the same task, often exhibit distinct distributions due to varying data sources Zhou et al. (2023b). In addition, while data collection is straightforward, the labeling process incurs high time and labor costs, leading to the hybrid multiple streams where massive labeled and unlabeled streams arrive simultaneously Yu et al. (2022a).

To tackle this scenario, multistream classification has been proposed, in which a model can be flexibly transferred from labeled source streams to the unlabeled target stream while employing online detection and adaptation working principles. This not only enables the model to adapt to new and unlabeled data streams but also mitigates the expenses and logistical challenges. The multistream classification problem features three major challenges that have to be tackled simultaneously: *1) Scarcity of labels:*

this arises from the absence of labels specifically for the target stream, while the source streams possess labeled data; *2) Covariate shift:* this implies that any two data streams exhibit distinct distributions, whether they are different source streams or a source stream and a target stream; and *3) Asynchronous drift:* the source and target streams are susceptible to independent concept drift, which occurs at varying time periods and results in unique effects on the model performance.

In recent years, several approaches have been proposed to address the multistream classification problem by using online domain adaptation and drift handling techniques Chandra et al. (2016); Haque et al. (2017); Pratama et al. (2019); Wang et al. (2021). However, many of these methods have primarily focused on the single-source stream, potentially impeding model performance due to limitations in the quality of the source data. Furthermore, such single-source-based approaches may be prone to overfitting issues. Accordingly, the multi-source configuration is introduced, which enables the acquisition of supplementary information from different source streams, thereby providing more valuable information to build a more accurate and robust model Wang et al. (2022g); Yang et al. (2021). However, leveraging the information from each individual source stream exposes a new challenge: *4) temporal dynamic correlations* between the source and target streams. In other words, any drift occurring within each stream has the potential to alter the correlation between the source and target streams. It is crucial for the predictive model to adapt promptly, extracting valuable insights from relevant source streams while avoiding the assimilation of irrelevant knowledge from other source streams.

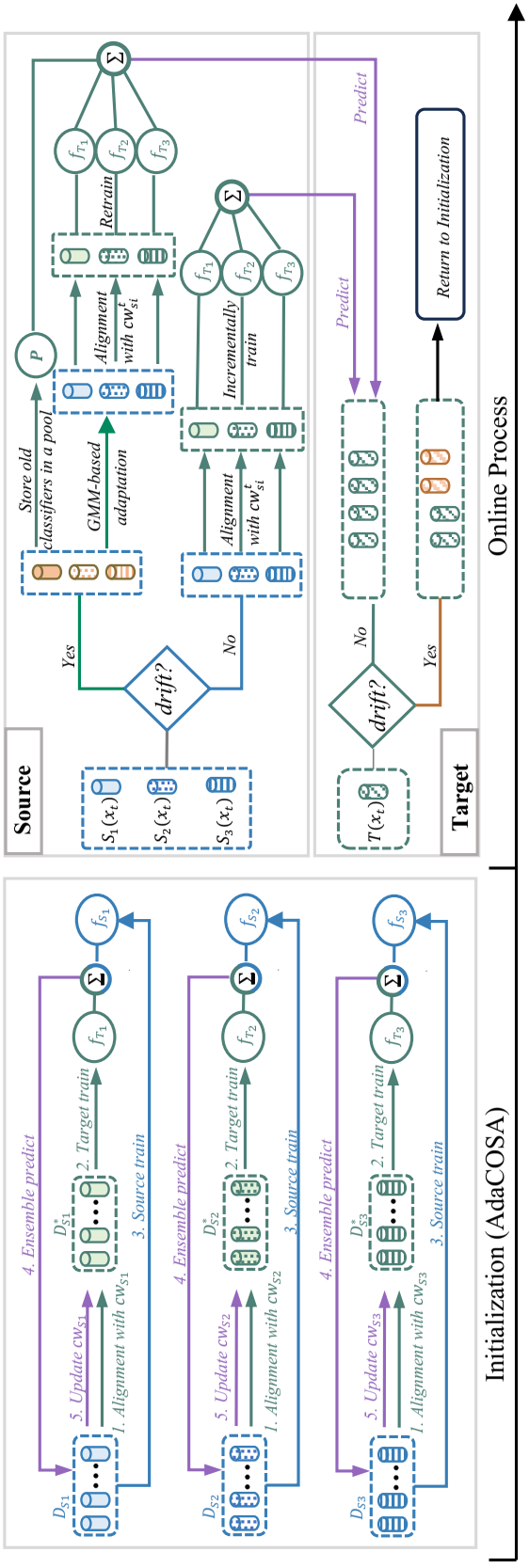


Figure 3.1: Framework of OBAL. The initialization stage is principally devoted to mitigating the problem of covariate shift, along with learning the intricate dynamic correlations that exist between various data streams. In the online phase, the core focus is on the detection and adaptation of asynchronous drift. This stage further integrates the covariate shift alignment and correlation matrices learned during the initial phase, facilitating a seamless ensemble prediction from the source to the target stream.

To address all issues in the multi-stream classification task, we propose the **O**nline **B**oosting **A**daptive **L**earning (OBAL) method. As shown in Figure 3.1, OBAL consists of two stages, the first of which is the initialization phase, where we propose the AdaCOSA algorithm. The fundamental principle of AdaCOSA involves an adaptive interaction between models learned in the original source space and those acquired in the target space, aiming to align the *temporal covariate shift* and explore the *dynamic relationships* between different data streams based on feedback from the target domain. This process reinforces positive knowledge transfer, leading to optimal model migration. The second stage involves the online processing phase, during which our primary aim is to detect and adapt to the *asynchronous drift* in each data stream in real time. To achieve this, we employ the Drift Detection Method (DDM) Gama et al. (2004) for labeled source streams, as it offers a stable and accurate detection approach. Simultaneously, we utilize the Gaussian Mixture Model (GMM) Oliveira et al. (2021) based weighting strategy for asynchronous drift adaptation in these streams. For the unlabeled target stream, we design two sliding windows and continuously monitor their distribution changes to effectively detect drift occurrences. Once a drift is detected in the target stream, it signifies that the dynamic relationships learned in the first stage are no longer applicable, necessitating a return to the first stage for reinitialization.

3.2 Proposed Method

3.2.1 Problem Definition

Multi-source-stream classification involves the presence of multiple labeled source streams and one unlabeled target stream as shown in Figure 3.2. These streams possess interconnected internal representations and share a common label space. The objective of this task is to predict the labels of the target stream by effectively transferring knowledge

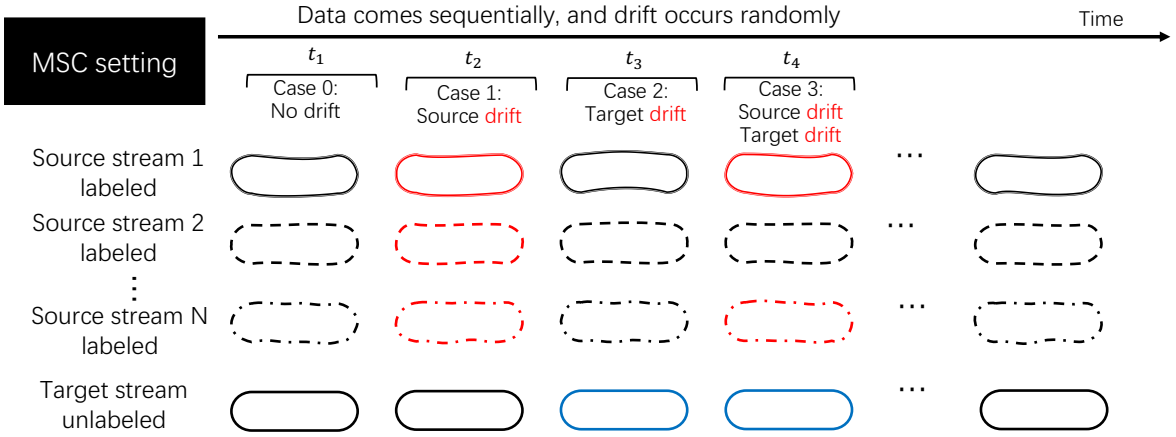


Figure 3.2: Illustration of multi-source-stream classification.

from the labeled source to the target stream, and it can be defined as follows.

Definition 3.1 ***Multi-source-stream Classification.*** *It involves N labeled source streams $S = \{S_1, S_2, \cdots, S_N\}$ and one unlabeled target stream T. Each arrived source sample at time t is represented by $S_i(\boldsymbol{x}_t, y_t)$, where $\boldsymbol{x}_t \in \mathscr{D}^d$ is the d features, and y_t is the true label of the instance which can only be observed in S_i, $i \in \{1, 2, \cdots, N\}$. It aims to build a classification model to predict the class label of a target sample $T(\boldsymbol{x}_t)$ at time t using the $S_i(\boldsymbol{x}_t, y_t)$ and $T(\boldsymbol{x}_t)$.*

As mentioned before, four main challenges must be addressed simultaneously in the multistream classification problem, i.e., *scarcity of labels*, *covariate shift*, *asynchronous drift* and *dynamic correlation*. These challenges are defined as follows,

Challenge 3.1 ***Scarcity of labels.*** *This is a major issue in the multistream classification problem. Labeled samples are provided only to the source streams $S_i(\boldsymbol{x}_t, y_t), i \in \{1, 2, \cdots, N\}$, leaving the target stream entirely unlabelled $T(\boldsymbol{x}_t)$. Consequently, the challenge lies in achieving accurate predictions in the target stream, where no labeled samples are available.*

Algorithm 1 Initialization (AdaCOSA)

Input: The archived data batches $D_{Si}, i \in \{1,2,\cdots N\}$ and D_T, Maximum iteration I_{max}.
Output: Target classifier f_{Ti}, weight vector $\mathbf{cw}_{Si}$.

```
Set up β_n and initialize cw_Si.
Get the mapped source data D*_Si according to Eq.(3.3).
for iter = 1 : I_max do
  for i = 1 : N do
    Create source classifiers f_Si(x) ← {D_Si, Y}.
    Create target classifiers f_Ti(x) ← {D*_Si, Y, cw_Si}
    Predict the instance from D_Si using F_est.
    Adjust the weight vector cw_Si according to Eq.(3.9).
  end for
end for
```

Challenge 3.2 ***Covariate shift.*** *Denoting P_{Si} and P_T as the distributions from $S_i, i \in 1,2,\cdots,N$ and T, all streams at the same time step are related but with covariate shift, i.e., $P_{Si}(y_t \mid \boldsymbol{x}_t) = P_{Sj}(y_t \mid \boldsymbol{x}_t) = P_T(y_t \mid \boldsymbol{x}_t)$ while $P_{Si}(\boldsymbol{x}_t) \neq P_{Sj}(\boldsymbol{x}_t) \neq P_T(\boldsymbol{x}_t)$*

Challenge 3.3 ***Asynchronous drift.*** *This refers to the observation of the effect of drift at different times on different independent non-stationary processes that continuously generate data from $S = \{S_1, S_2, \cdots, S_N\}$ and T.*

- ***Source Drift****: $\exists t$ if $P_{Si}(\boldsymbol{x}_t) \neq P_{Si}(\boldsymbol{x}_{t+1}), i \in 1,2,\cdots,N$ but $P_T(\boldsymbol{x}_t) = P_T(\boldsymbol{x}_{t+1})$, the drift only occurs in the source stream.*
- ***Target Drift****: $\exists t$ if $P_{Si}(\boldsymbol{x}_t) = P_{Si}(\boldsymbol{x}_{t+1}), i \in 1,2,\cdots,N$ but $P_T(\boldsymbol{x}_t) \neq P_T(\boldsymbol{x}_{t+1})$, the drift only occurs in the target stream.*
- ***Concurrent Drifts****: $\exists t$ if $P_{Si}(\boldsymbol{x}_t) \neq P_{Si}(\boldsymbol{x}_{t+1}), i \in 1,2,\cdots,N$ and $P_T(\boldsymbol{x}_t) \neq P_T(\boldsymbol{x}_{t+1})$, it means drift occurs in both source and target streams.*

Challenge 3.4 ***Temporal dynamic correlation.*** *The dynamic interplay between source and target streams leads to varying relevance, expressed as $C(S_i(\boldsymbol{x}_t), T(\boldsymbol{x}_t))$. At the time t, some source streams may possess complementary information $C(S_i(\boldsymbol{x}_t), T(\boldsymbol{x}_t)) = +$, while*

others may contain negative information $C(S_i(\boldsymbol{x}_t), T(\boldsymbol{x}_t)) = -$. *The complexity arises as* C *may change over time, such as* $C(S_i(\boldsymbol{x}_t), T(\boldsymbol{x}_t)) \neq C(S_i(\boldsymbol{x}_{t+\tau}), T(\boldsymbol{x}_{t+\tau}))$, *disrupting the inherent relationship between the streams.*

To address all challenges, we propose the OBAL method which comprises two stages: initialization (AdaCOSA) and online processing. Next, we will provide a detailed description of these two stages.

3.2.2 Adaptive Covariate Shift Adaptation (AdaCOSA)

To align the covariate shift $P_{Si}(\boldsymbol{x}_t) \neq P_{Sj}(\boldsymbol{x}_t) \neq P_T(\boldsymbol{x}_t)$ as well as to explore the temporal dynamic relationship $C(S_i(\boldsymbol{x}_t), T(\boldsymbol{x}_t))$ between source and target streams, we propose an AdaCOSA algorithm. Inspired by the CORrelation ALignment (CORAL) method Sun et al. (2016), the covariance between shifting domains can be aligned by minimizing the distance between the second-order statistics, which provides a stable and effective solution. However, the standard CORAL method is incapable of identifying source instances that are irrelevant to the target, thereby leading to negative transfer effects Wang et al. (2019); Yang et al. (2021). Furthermore, it fails to address the dynamic relationship between the data streams. As a solution, we propose an adaptive re-weighting strategy to dynamically and iteratively adjust the weights of the source data based on their relevance to the target domain.

Specifically, given any archived source data batch $D_{Si} = S_i(\boldsymbol{X}, Y), i \in \{1, 2, \cdots, N\}$ and the target data batch $D_T = T(\boldsymbol{X})$, we first assign a correlation weight vector $\mathbf{cw}_{Si} = [cw_{Si}^1, cw_{Si}^2, \cdots cw_{Si}^{L_n}], i \in \{1, 2, \cdots, N\}$ to each source stream, where L_n is the instance number of each archived data batch. Then we can align the shifting covariance by mapping each weighted source data to the target domain using a transformation matrix A_{Si}, and the objective can be formulated as,

$$\min_{A_{Si}} \left\| C_{\hat{S}_i} - C_T \right\|_F^2 = \min_{A_{Si}} \left\| A_{Si}^\top C_{Si} A_{Si} - C_T \right\|_F^2, \tag{3.1}$$

where $\|\cdot\|_F^2$ is the Frobenius norm. C_{Si} and C_T are the covariance matrices of $\mathbf{cw}_{Si}D_{Si}$ and D_T, respectively. $C_{\hat{S}i}$ is the covariance matrix of transformed source features $\mathbf{cw}_{Si}D_{Si}A$, and

$$\begin{aligned} C_{S_i} &= \mathrm{cov}(\mathbf{cw}_{Si}D_{Si}) + \mathrm{eye}(\mathrm{size}(\mathbf{cw}_{Si}D_{Si}, 2)), \\ C_T &= \mathrm{cov}(D_T) + \mathrm{eye}(\mathrm{size}(D_T, 2)). \end{aligned} \tag{3.2}$$

Then the aligned source data D_{Si}^* can be obtained by the classical whitening and re-coloring strategy Sun et al. (2016),

$$D_{Si}^* = \mathbf{cw}_{Si}D_{Si}C_S^{\frac{-1}{2}}C_T^{\frac{1}{2}}. \tag{3.3}$$

The solution for this general case is derived from the following lemma in detail.

Lemma 3.1 *Cai et al. (2010) Let Y be a real matrix of rank r_Y and X be a real matrix of rank at most r, where $r \le r_Y$. let $Y = U_Y\Sigma_Y V_Y$ be the SVD of Y, and $\Sigma_{Y[1:r]}, U_{Y[1:r]}, V_{Y[1:r]}$ be the largest r singular values and the corresponding left and right singular vectors of Y respectively. Then $X^* = U_{Y[1:r]}\Sigma_{Y[1;r]}V_{Y[1;r]}^\top$ is the optimal solution to the problem of $\min_X \|X - Y\|_F^2$.*

Theorem 3.1 *Sun et al. (2016) Let Σ^+ be the Moore-Penrose pseudoinverse of Σ and r_{C_T} denote the rank of C_{Si} and C_T respectively. Then, $A^* = U_{Si}\Sigma_{Si}^{+\frac{1}{2}}U_{Si}^\top U_{T[1:r]}\Sigma_{T[1:r]}^{\frac{1}{2}}U_{T[1:r]}^\top$ is the optimal solution of Eq. (3.1) with $r = min(r_{C_{Si}}, r_{C_T})$.*

Proof. Since A is a linear transformation, $A^\top C_{Si}A$ will not increase the rank of C_{Si}, i.e., $r_{C_{\hat{S}_i}} \le r_{C_S i}$. Conducting SVD on C_{Si} and C_T, we can get $C_{Si} = U_{Si}\Sigma_{Si}U_{Si}^\top$ and $C_T = U_T\Sigma_T U_T^\top$, respectively. In order to get the optimal value of $C_{\hat{S}i}$, we consider the following two cases:

- *case 1:* $r_{C_{Si}} > r_{C_T}$. The optimal solution is $C_{\hat{S}i} = C_T$. Thus, $C_{\hat{S}i} = U_T\Sigma_T U_T^\top = U_{T[1:r]}\Sigma_{T[1:r]}{U_{T[1:r]}}^\top$ is the optimal solution of Eq.(3.1) with $r = r_{C_T}$.

- *case 2:* $r_{C_{Si}} \leq r_{C_T}$. Then, according to Lemma 3.1, $C_{\hat{S}i} = U_T \Sigma_T U_T^\top = U_{T[1:r]} \Sigma_{T[1:r]} {U_{T[1:r]}}^\top$ is the optimal solution of Eq.(3.1) where $r = r_{C_{Si}}$.

Therefore, the optimal solution of Eq.(3.1) can be derived as $C_{\hat{S}i} = U_T \Sigma_T U_T^\top = U_{T[1:r]} \Sigma_{T[1:r]} {U_{T[1:r]}}^\top$ with $r = min(r_{C_{Si}}, r_{C_T})$. Then, to obtain A based on the above analysis, let $C_{\hat{S}i} = A^\top C_{Si} A$ and we can get:

$$A^\top C_{Si} A = U_{T[1:r]} \Sigma_{T[1:r]} U_{T[1:r]}^\top \tag{3.4}$$

Since $C_{Si} = U_{Si} \Sigma_{Si} U_{Si}^\top$, we have

$$A^\top U_{Si} \Sigma_{Si} U_{Si}^\top A = U_{T[1:r]} \Sigma_{T[1:r]} U_{T[1:r]}^\top \tag{3.5}$$

It can be re-written as

$$\left(U_{Si}^\top A\right)^\top \Sigma_{Si} \left(U_{Si}^\top A\right) = U_{T[1:r]} \Sigma_{T[1:r]} U_{T[1:r]}^\top \tag{3.6}$$

Assuming $E = \Sigma_{Si}^{+\frac{1}{2}} U_{Si}^\top U_{T[1:r]} \Sigma_{T[1:r]}^{\frac{1}{2}} U_{T[1:r]}^\top$, then the right side of the above equation can be simplified as $E^\top \Sigma_S i E$. This gives

$$\left(U_{Si}^\top A\right)^\top \Sigma_{Si} \left(U_S^\top A\right) = E^\top \Sigma_{Si} E \tag{3.7}$$

Therefore, we can get $U_{Si}^\top A$, and the optimal solution of A can be calculated by

$$A = U_{Si} E = \left(U_{Si} \Sigma_{Si}^{+\frac{1}{2}} U_{Si}^\top\right) \left(U_{T[1:r]} \Sigma_{T[1:r]}^{\frac{1}{2}} U_{T[1:r]}^\top\right). \tag{3.8}$$

Finally, as analyzed in Sun et al. (2016), the first part $U_{Si} \Sigma_{Si}^{+\frac{1}{2}} U_{Si}^\top$ whitens the source data while the second part $U_{T[1:r]} \Sigma_{T[1:r]}^{\frac{1}{2}} U_{T[1:r]}^\top$ re-colors it with the target covariance.

Next, we use a supervised method to train the source classifiers f_{Si} using raw source data $\{D_{Si}, Y\}$. In addition, the covariate-adopted target classifiers f_{Ti} can be learned by using the transformed $\{D_{Si}^*, Y\}$. Finally, we can employ an average ensemble F_{est} that combines models derived from each original source space f_{Si} with those learned in the target space f_{Ti} to re-evaluate the source data iteratively.

Once the predicted label $\hat{y}_i$ is obtained, it can be used to re-estimate the correlation weights of the source instances because it contains reliable responses from the target domain. In each iteration, if the source instance is predicted mistakenly, this instance may likely conflict with the target stream. Then the effect of this irrelevant data will be diminished in the next iteration by decreasing its training weight. In contrast, accurate predictions indicate a minimal distance or positive correlation between the source and target domains, resulting in increased training weights to enhance learning. Here, the weight can be updated by,

$$cw_{Si}^{t} = cw_{Si}^{t} \cdot e^{-\beta_n |\hat{y}_i - y_i|}, \tag{3.9}$$

where β_n is a hyper-parameter defined as $\beta_n = 0.5 \ln\left(1 + \sqrt{2 \ln \frac{L_n}{I_{\max}}}\right)$. L_n is the total number of samples of the archived data batch D_{S_i}, and $I_{\max}$ is the maximum iterations for adaptive re-weighting.

After several iterations, the instances that exhibit a positive correlation with the target stream will be assigned higher training weights, whereas the training instances that diverge from the target stream will receive lower weights. The detailed process is presented in Algorithm 1. After that, the weight cw_{Si} of each target base classifier can be assigned based on the learned correlation weight and it is calculated by $cw_{Si} = \frac{1}{L_n} \sum_{t=1}^{L_n} cw_{Si}^{t}$. Therefore, the final ensemble f_E for the target stream can be formulated as follows:

$$f_E(x) = \frac{cw_{Si}}{\sum_{i=1}^{N} cw_{Si}} f_{Ti}. \tag{3.10}$$

3.2.3 Online Detection and Adaptation

As stated in Challenge 3.3, asynchronous concept drifts may occur in either the source or target streams over time. Therefore, for any given stream, it is necessary to continuously

Algorithm 2 The learning process of OBAL

Input: Multiple labeled source streams $\{S_1, S_2, \cdots, S_N\}$, Unlabeled target stream T, classifier pool P, initial sample size L_n.

Output: Labels predicted on T data.

1: $D_{Si}, D_T \leftarrow$ Read first L_n instances from S_i and T.
2: $f_E(x), \mathbf{cw}_{Si} \leftarrow$ initialize according to Algorithm 1.
3: Create the source detectors and adaptors DDM_{Si}, GMM_{Si}
4: Create the target Gaussian Mixture Model GMM_T.
5: Create the detection and reference windows W_{det}, W_{ref}.
6: **while** there is incoming data **do**
7: **for** $i = 1 : N$ **do**
8: **if** DDM_{Si} = True **then**
9: GMM-based adaptation by Eq.(3.13).
10: Weighted alignment and retrain a new classifier.
11: **else**
12: Weighted alignment and incrementally update.
13: **end if**
14: **end for**
15: Move detention window and calculate $\mu_{\text{det}}, \mu_{\text{ref}}$.
16: **if** Eq. (3.16) = True **then**
17: Remove all base classifiers and return to line 2.
18: **else**
19: Predict the target sample.
20: **end if**
21: **end while**

monitor its drifting situation in real time and promptly perform drift adaptation to accommodate the new concept.

3.2.3.1 Source stream processing

For scenarios involving source drift, existing supervised drift detectors such as DDM can be employed, which offers more accurate drift detection because of the leveraging of labels. As a new source sample $S_i(\boldsymbol{x}_t)$ arrives, the source classifier predicts its label, and then the drift detector is updated based on the prediction error. If no drift is detected, we will incrementally train the target classifier using the weighted mapped $S_i^*(\boldsymbol{x^*}_t, y)$ with its corresponding weight cw_{si}^t. Since we have obtained the optimal weights

$\mathbf{cw}_{Si} = [cw^1_{Si}, cw^2_{Si}, \cdots cw^{L_n}_{Si}]$ for the archived data batch during the initialization stage, we can retrieve the most relevant data from the archived data batch and assign its weights to the new coming data by indexing the minimum L_2 distance between new coming and archived data instances.

However, once a drift is detected within each source stream, an adaptation module should be deployed to handle new concepts. Here, we utilize the GMM to evaluate the distributions of the old and new concepts. GMM assumes several mixture components can model all real-world data, and it is formulated as follows:

$$P(x) = \sum_{k=1}^{K} P(x \mid C_k) \cdot w_k, \tag{3.11}$$

where K represents the total number of Gaussians or mixture components, and x is the observed multivariate. w_k is a weight that is determined by the observations that constitute C_k, and $0 \leq w_k \leq 1, \sum_{k=1}^{K} w_k = 1$. $P(x \mid C_k)$ represents the likelihood of observation x being assigned to mixture component C_k. It can be calculated by using the mean and the covariance of each mixture component C_k as follows:

$$P(x \mid C_k) = \frac{1}{\left(2\pi^{d/2}\sqrt{|\Sigma_k|}\right)} \exp\left(-\frac{1}{2}\left(x-\mu_k\right)^T \Sigma_k^{-1}\left(x-\mu_k\right)\right). \tag{3.12}$$

According to the Expectation-Maximization (EM) algorithm, all the parameters of different mixture components are randomly initialized using the archived data batch D_{si}. Subsequently, it iteratively adjusts the mean and covariance of the mixture component to maximize the likelihood of each mixture component. For a newly incoming instance $S_i(\boldsymbol{x}_t)$, its importance weight $aw^t_{S_i}$ can be calculated by maximizing the conditional probability of GMM as follows:

$$aw^t_{S_i} = \max_{k \in \{1,2,\ldots,K\}} P(S_i(\boldsymbol{x}_t) \mid C_k). \tag{3.13}$$

Then, the new coming concept in any source stream can be adapted to the old concept by multiplying $aw^t_{S_i}$. Thus, its optimal correlation weight cw^t_{si} with the target stream

can also be obtained from the learned $\mathbf{cw}_{Si}$. Finally, a new target base classifier will be created and trained by using weighted mapped $S_i^*(\boldsymbol{x}^*{}_t, y)$ with its corresponding weight cw_{si}^t. Note that old base classifiers are no longer trained with new samples but are instead preserved within a base classifier pool denoted as P, allowing for their retention. Finally, the joint predictive probability can be ensembled as,

$$f^E(x) = \frac{w_{Si}}{\sum_{i=1}^{N} w_{Si} + \sum_{l=1}^{|P|} w_P} f_{T_i}(x) + \frac{w_P}{\sum_{i=1}^{N} w_{Si} + \sum_{l=1}^{|P|} w_P} f_P(x), \tag{3.14}$$

where w_P is the weight of l-th classifer in P, and $w_{Si} = \frac{1}{n}\sum_{t=1}^{n} aw_{S_i}^t * cw_{S_i}^t$.

3.2.3.2 Target stream processing:

In order to detect the drift in the target stream without utilizing labels, we use the archived target data batch D_T to initialize a GMM model and deploy two sliding windows to detect the changes over time. Specifically, we design two sliding windows, i.e., Reference Window $W_{ref} = \{T(x_1), \cdots, T(x_n)\}$ and Detect Window $W_{det} = \{T(x_{n+1}), \cdots, T(x_{2n})\}$, where n is the instance number within the window and it is set as $n = L_n$. Then, the average conditional probability of the reference window can be calculated by a point estimation of the mean for the normal distribution,

$$\mu_{\text{ref}} = \frac{1}{n}\sum_{t=1}^{n} \max_{k\in\{1,2,\ldots,K\}} P\left(T(\boldsymbol{x}_t) \mid C_k\right). \tag{3.15}$$

The confidence interval estimation of the μ_{ref} is known to be $\left[\mu_{\text{ref}} - z_\alpha(\sigma/\sqrt{n}), \mu_{\text{ref}} + z_\alpha(\sigma/\sqrt{n})\right]$, where σ is the standard deviation and z_α is the significance level which is set as 3 Kim and Park (2017). The decision is made that the change has occurred when the point estimation by the mean μ_{ref} in the detection window satisfies,

$$\mu_{\text{det}} \geq \mu_{\text{ref}} + z_\alpha \times \sigma/\sqrt{n}. \tag{3.16}$$

Table 3.1: Characteristics of all datasets including 3 sources and 1 target stream.

	Datasets	Drift types	Type	#Instances	#Features	#Classes
Synthetic	SEA	Sudden/recurring	Single	25K * 4	3	2
	Tree	Sudden/gradual	Single	5K * 4	20	2
	RBF	Incremental	Single	5K * 4	10	2
	Hyperplane	Incremental	Single	30K* 4	4	2
Real-world	Weather	Unknown	Single	4.5K* 4	8	2
	Kitti	Unknown	Single	6.25K * 4	55	8
	CNNIBN	Unknown	Multistream	30K * 4	124	2
	BBC	Unknown	Multistream	30K * 4	124	2

Otherwise, W_{ref} and W_{det} move step by step to receive new incoming data, i.e., $W_{ref} = \{T(x_i), \cdots, T(x_{n+i-1})\}$ and $W_{det} = \{T(x_{n+i}), \cdots, T(x_{2n+i-1})\}$. Once a change is detected, the historical base classifier becomes ineffective for classifying the target samples. Consequently, all base classifiers are eliminated from the base classifier pool, and the model undergoes re-initialization to adapt to the new concepts. The whole learning process is summarized in Algorithm 2.

3.3 Experiments

In the experiment, we first empirically demonstrated that OBAL consistently outperforms current methods in multi-stream classification, highlighting both robustness and superiority. Second, we validated the substantial impact of dynamic inter-stream relationships on prediction, emphasizing the effectiveness of the AdaCOSA by ablation study. Finally, we established the scalability of OBAL across diverse data streams, corroborating its stable predictive capabilities. In addition, we also analyzed parameter sensitivity and the time complexity and cost of the algorithm.

3.3.1 Benchmark Datasets

As shown in Table 3.1, we conduct the experiment on four synthetic datasets (i.e., SEA Street and Kim (2001), Tree Liu et al. (2020b), RBF Song et al. (2021a), and Hyperplane Bifet and Gavalda (2007)) and four popular real-world datasets (Weather Ditzler and Polikar (2012), Kitti Geiger et al. (2012), CNNIBN Vyas et al. (2014), and BBC Vyas et al. (2014)), and more detailed descriptions of each dataset and multistream scenario simulation are detailed as follows,

- SEA Street and Kim (2001) is a synthetic dataset with two classes consisting of abrupt and recurring drifts. There are three features and the feature's values range from 0 to 10. When $f_1 + f_2 \leq \theta$, the data belongs to class 1. Here, f_1 and f_2 represent the first and second features, respectively. And θ denotes the threshold for binary classification, which changes from $4 \rightarrow 7 \rightarrow 4 \rightarrow 7$.

- Tree Liu et al. (2020b) is generated based on a tree structure where features are randomly split, and labels are assigned to the tree leaves. Each attribute is assigned a random value from a uniform distribution to create a new sample, while new concepts are generated by constructing new trees.

- RBF Song et al. (2021a) generator generates data instances using a radial basis function. Centroids are created randomly and assigned a standard deviation value, a weight, and a class label. Incremental drifts are simulated by continuously moving the centroids.

- Hyperplane Bifet and Gavalda (2007) is also a synthetic dataset based on a rotating hyperplane explained in Hulten et al. (2001). Positive labels are assigned to examples where $\sum_{j=1}^{d} \omega_j x_j > \omega_0$, while negative labels are assigned to examples where $\sum_{j=1}^{d} \omega_j x_j < \omega_0$. Concept drifts can be simulated by adjusting the relative weights.

- Weather Ditzler and Polikar (2012) is a real-world dataset, which pertains to the task of one-step-ahead prediction for determining the occurrence of rainfall. It encompasses weather data spanning a period of 50 years, capturing both the annual seasonal variations and the long-term climate changes.

- Kitti Geiger et al. (2012) presents a real-world computer vision challenge that stems from the autonomous driving scenario. The primary objective is to accomplish 3D object detection, employing two high-resolution video cameras,Äîone capturing color images and the other grayscale images,Äîto capture the objects of interest.

- TV News Channel Commercial Detection Dataset[1] Vyas et al. (2014) is a real-world multistream dataset. It comprises of prominent audio-visual features collected from 150 hours of television news broadcasts, including 30 hours each from five news channels (i.e., BBC, CNNIB, CNN, NDTV, and TIMESNOW). All the video shots are recorded in a sequential way and used for commercial or non-commercial detection. In this chapter, we designate CNNIBN and BCC as the target streams, while treating the remaining streams as source streams to simulate a multistream classification task. Each individual data stream comprises 30,000 samples, thus providing two substantial benchmarks (CNNIBN and BBC) for analysis and evaluation. Specifically, the original dataset is multimodal and contains 5 sets of video features (i.e., video shot length, screen text distribution, motion distribution, frame difference distribution, and edge change ratio) and 7 sets of audio features (i.e., short-term energy, zero crossing rate, spectral centroid, spectral flux, spectral roll-off frequency, fundamental frequency and bag of audio words), totally for 4125 dimensions. In this experiment, we remove the bag of audio words feature and just use the other 11 sets of features. In addition, to retain as much of the original data as possible, we re-sampled all data streams to 30,000 samples.

[1] https://archive.ics.uci.edu/dataset/326/tv+news+channel+commercial+detection+dataset

To simulate the multistream classification scenario, we first sort all samples in descending order according to the probability of each sample $P(x) = \exp \frac{(x-\bar{x})^2}{2\sigma^2}$ in a Gaussian distribution, which induces the problem of covariate shift. And then the construction of source streams follows a sequential order, with the first source stream being built upon the top N_i samples, followed by the second source stream, the third source stream, and so on up to $(N-1)$-th source stream. The remaining data samples are then assigned to the target stream. All samples selected in each stream will be recovered to the original chronological order to maintain the raw temporal relationship (i.e., Asynchronous drift). Only source streams exclusively consist of labels, whereas the target stream lacks labels, resulting in the scarcity of labels problem.

3.3.2 Baselines and Experiment Settings

To demonstrate the superiority of our proposed method, we conducted experiments comparing it with five state-of-the-art methods. Among them, the FUSION Haque et al. (2017) and ATL Pratama et al. (2019) algorithms are based on single-source streams, while the Melanie Du et al. (2019), AOMSDA Renchunzi and Pratama (2022), and MCMO Jiao et al. (2022b) are specifically designed for the multi-source classification scenario. For FUSION and ATL, we pair each source stream with the target stream, resulting in three distinct groups denoted as {FUSIONs1, FUSIONs2, and FUSIONs3} and {ATLs1, ATLs2, ATLs3}, respectively. Among all compared methods, FUSION[2], Melanie[3] and MOMO[4] are ensemble learning-based methods, which are all implemented by Python. To ensure an equitable comparison, we adopt the Hoffeding Tree as the base classifier for all these three methods. On the other hand, ATL[5] and AOMSDA[6] employ

[2]https://github.com/ahhaque/FUSION
[3]https://github.com/nino2222/Melanie
[4]https://github.com/Jesen-BT/MCMO
[5]https://github.com/w870792/ATL
[6]https://github.com/Renchunzi-Xie/AOMSDA

the neural network paradigm and are executed in Matlab. In this study, we adhere to the configurations present in their respective original public source codes.

In this study, we implemented the framework using the scikit-multiflow learning library Montiel et al. (2018) in Python. All experimental evaluations were conducted on a server equipped with 187GB of memory and powered by an Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz.

3.3.3 Overall Performance.

Table 3.2 compares the classification accuracy of OBAL against all baselines on four synthetic and four real-world datasets. Overall, OBAL outperforms all other unsupervised multistream classification methods on both synthetic and real-world datasets, while it performs better than the supervised method (Melanie) on six out of eight datasets. First, compared to single-source-based methods (Fusion and ATL), all multi-source-based methods demonstrate significant improvement. This proves that multiple labeled source streams can provide more discriminative and complementary information, resulting in more accurate and robust predictions. Compared with Melanie, OBAL performs remarkably close to or even surpasses without considering the target labels. This is because we not only mitigate the covariate shift but also adaptively adjust sample weights based on the feedback from the target domain. This effectively avoids negative transfer from irrelevant data, thereby ensuring better prediction accuracy. Although AOMSDA and MCMO also consider exploiting the complementary information among multiple source data streams, they ignore the underlying correlation between various streams. In contrast, OBAL employs an adaptive re-weighting approach to iteratively decrease the weights of negative transfer samples and strengthen the weights of positive transfer samples based on the predictive feedback from the target domain. As a result, OBAL achieves the best predictive performance.

Table 3.2: Classification accuracy (%) with the variance of various methods on all benchmarks.

	FUSIONs1	FUSIONs2	FUSIONs3	ATLs1	ATLs2	ATLs3	Melanie	AOMSDA	MCMO	OBAL (ours)
SEA	$85.04_{\pm 0.84}$	$85.78_{\pm 0.92}$	$84.31_{\pm 1.13}$	$88.42_{\pm 1.70}$	$88.74_{\pm 1.75}$	$87.62_{\pm 1.01}$	$89.18_{\pm 0.77}$	$90.23_{\pm 1.42}$	$87.46_{\pm 2.12}$	$\mathbf{90.98}_{\pm 0.87}$
Tree	$76.98_{\pm 1.11}$	$76.74_{\pm 1.00}$	$75.21_{\pm 1.07}$	$76.43_{\pm 2.17}$	$76.71_{\pm 1.86}$	$76.07_{\pm 2.42}$	$\mathbf{78.93}_{\pm 0.61}$	$76.87_{\pm 3.47}$	$77.64_{\pm 1.47}$	$78.45_{\pm 1.01}$
RBF	$82.03_{\pm 1.41}$	$83.46_{\pm 1.20}$	$81.03_{\pm 1.73}$	$84.53_{\pm 2.01}$	$85.21_{\pm 1.85}$	$83.16_{\pm 2.13}$	$86.04_{\pm 0.39}$	$85.26_{\pm 2.89}$	$86.26_{\pm 0.77}$	$\mathbf{86.78}_{\pm 0.91}$
Hyperplane	$83.29_{\pm 0.67}$	$84.05_{\pm 0.52}$	$82.17_{\pm 0.57}$	$86.17_{\pm 1.04}$	$87.07_{\pm 1.21}$	$86.01_{\pm 1.49}$	$86.38_{\pm 0.57}$	$87.66_{\pm 1.74}$	$84.04_{\pm 1.42}$	$\mathbf{88.01}_{\pm 1.17}$
Weather	$71.04_{\pm 1.50}$	$70.65_{\pm 1.32}$	$72.17_{\pm 1.17}$	$74.57_{\pm 1.94}$	$75.03_{\pm 2.01}$	$74.62_{\pm 1.77}$	$77.74_{\pm 0.89}$	$76.55_{\pm 1.41}$	$76.02_{\pm 3.43}$	$\mathbf{79.22}_{\pm 2.07}$
Kitti	$54.21_{\pm 2.61}$	$52.36_{\pm 2.72}$	$50.38_{\pm 2.43}$	$52.78_{\pm 3.78}$	$54.01_{\pm 3.09}$	$53.26_{\pm 3.21}$	$50.29_{\pm 1.34}$	$67.79_{\pm 3.16}$	$64.82_{\pm 4.17}$	$\mathbf{70.29}_{\pm 3.42}$
CNNIBN	$66.76_{\pm 0.74}$	$67.54_{\pm 1.11}$	$65.34_{\pm 0.92}$	$62.78_{\pm 1.44}$	$65.74_{\pm 1.76}$	$62.65_{\pm 1.38}$	$68.79_{\pm 0.31}$	$69.07_{\pm 1.40}$	$68.83_{\pm 0.89}$	$\mathbf{70.71}_{\pm 0.77}$
BBC	$61.76_{\pm 0.09}$	$61.26_{\pm 0.43}$	$59.86_{\pm 0.19}$	$62.78_{\pm 1.16}$	$62.34_{\pm 0.83}$	$60.76_{\pm 0.77}$	$\mathbf{68.04}_{\pm 0.01}$	$63.36_{\pm 1.07}$	$60.12_{\pm 1.51}$	$66.43_{\pm 1.42}$

Table 3.3: Classification accuracy (%) of OBAL variants.

	OBAL_{v1}	OBAL_{v2}	OBAL_{v3}	OBAL
SEA	79.54	82.48	88.76	**90.98**
Tree	72.42	74.74	77.01	**78.45**
RBF	79.78	81.41	84.23	**86.78**
Hyperplane	81.42	82.35	86.34	**88.01**
Weather	72.25	74.18	77.43	**79.22**
Kitti	62.14	64.09	68.24	**70.29**
CNNIBN	63.12	67.44	69.01	**70.71**
BBC	58.02	62.77	64.12	**66.43**

3.3.4 Ablation Study

To validate the rationality of each component and its impact on the overall classification results, we designed three variants of OBAL. As shown in Table 3.3, OBAL_{v1} as a baseline design does not consider the synchronous drift and covariate shift adaptations. In this situation, each stream is assigned a base classifier and it is updated incrementally. Thus, the performance of OBAL_{v1} is the worst, and significantly lower than that of OBAL on all datasets. This phenomenon highlights the crucial role of concept drift adaptation in dynamic environment learning. OBAL_{v2} considers the synchronous drift in each stream while ignoring the covariate shift alignment. OBAL_{v3} further employs the traditional CORAL strategy to align the covariate shift which does not explore the dynamic correlation. By comparing OBAL_{v2} and OBAL_{v3}, it can be seen aligning the covariate shift can effectively enhance the performance of the target prediction. Furthermore, the final OBAL highlights the significance of appropriate weights in mitigating the influence of irrelevant source samples and effectively addressing the problem of covariance shift.

Table 3.4: Parameter settings on different datasets.

	L_n	I_{max}	$\|P\|$
SEA	200	3	5
Tree	200	3	5
RBF	300	4	10
Hyperplane	400	3	5
Weather	100	4	5
Kitti	100	5	2
CNNIBN	200	3	5
BBC	300	3	10

3.3.5 Influence of Source Numbers

In this section, we examine the impact of the number of source streams. To ensure a fair comparison with a fixed target stream, we initially sample seven streams from all datasets and vary the number of source streams. Specifically, we evaluate the performance of OBAL using 1, 3, 5, and 7 source streams, respectively.

This experiment first investigates whether using multi-source streams improves predictive capability compared to a single-source stream. From Figure 3.3, we can observe that the performance of multi-source streams outperforms single-stream performance on all datasets. This indicates that multi-source streams can provide additional complementary information to enhance predictive performance.

However, as the number of source streams increases, there may be a decline in performance. For example, the performance with five source streams is better than that with seven source streams on the Tree dataset. This may be because as the number of source streams increases, the complexity of the model also increases, which affects its performance. Overall, the performance of OBAL is stable across various sources, which demonstrates that our proposed method can easily adapt to different numbers of data

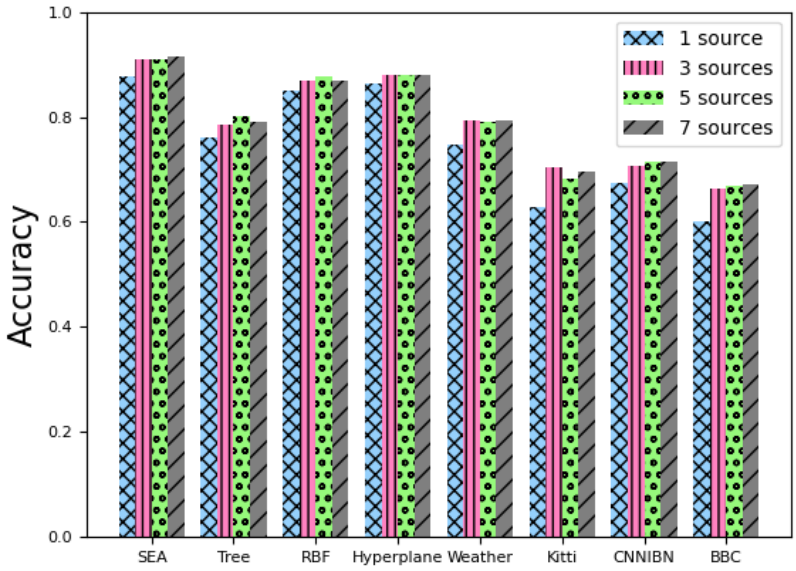


Figure 3.3: The influence of the different number of sources.

streams.

3.3.6 Parameter Sensitivity

In the proposed OBAL, there are three main parameters affecting the classification performance, including the window size of the initialization stage L_n, the re-weighting steps I_{max}, and the maximum classifier pool size $|P|$. To analyze their impact on the overall performance, we carry out experiments under various values of all parameters on all datasets. Here, we set $L_n \in \{100, 200, 300, 400, 500\}$, $I_{max} \in \{1, 3, 5, 7, 10\}$ and $|P| \in \{1, 5, 10, 15, 20\}$. During the experiment, each parameter is tuned while others are kept fixed, and the various predictive performances are shown in Figure 3.4.

Different datasets display varying optimal window sizes due to their unique drift frequencies and periods. For those with frequent drifts, a larger window might encompass multiple concepts, complicating accurate covariate adaptation. Hence, matching

the window size to the dataset's drift characteristics is crucial for effective prediction. In the re-weighting phase, the optimal number of iterations for most datasets is three. This is because the algorithm tends to overfit during the initialization phase with an increasing number of iterations. Additionally, as the classifier pool size grows, predictive performance generally improves across datasets, underscoring the importance of retaining historical data. However, after a certain threshold, this performance enhancement plateaus. Detailed parameter settings are shown in Table 3.4.

3.3.7 Time Complexity and Execution Time

In our method, we employed the Hoeffding Tree as the base classifier with time complexity of $O(L_n log(L_n))$. Thus, the time complexities of AdaCOSA is $O(L_n log(L_n))O(I_{max}N)$, where L_n, I_{max} and N are the sample number in the initialization stage, re-weighting interaction and source numbers. During the online process, there are three main modules, GMM, DDM, and Hoeffding Tree classifier. We use the EM algorithm to estimate the GMM parameters and the complexity of EM can be regarded as linear $O(L_n)$ in this method. Time complexities of DDM and Hoeffding Tree are $O(L_n)$ and $O(N)O(L_n log(L_n))$. Therefore, the overall complexity of OBAL is $O(L_n log(L_n))O(I_{max}N)+O(L_n)+O(N)(L_n log(L_n))$. In fact, N and I_{max} are quite small, so the complexity of OBAL depends on the size of L_n. Therefore, we can tune L_n to execute OBAL within the available resource.

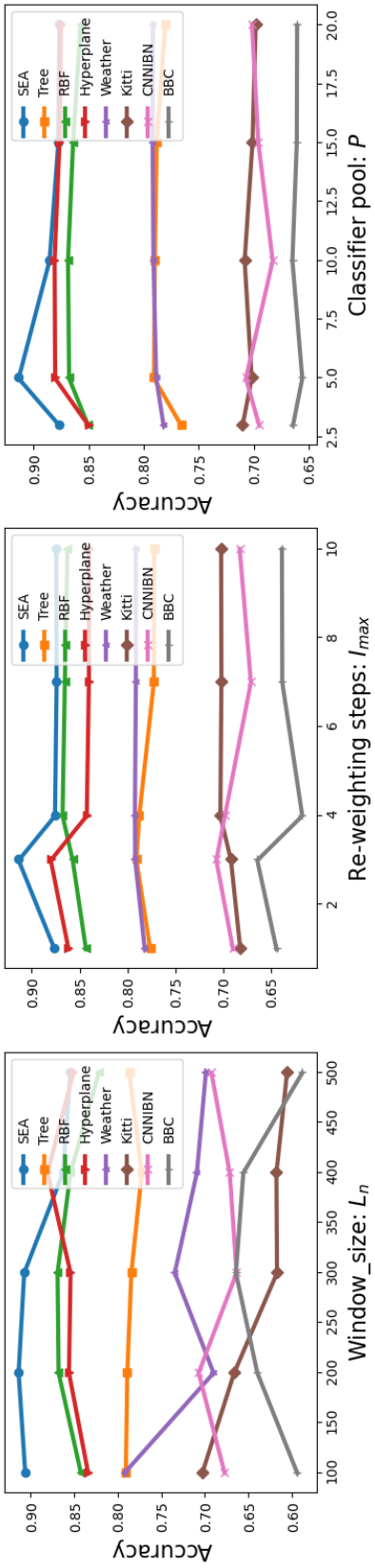


Figure 3.4: The effect of different parameters on classification accuracy.

Table 3.5: Execution time (s) of various methods on all benchmarks.

	FUSIONs1	FUSIONs2	FUSIONs3	ATLs1	ATLs2	ATLs3	Melanie	AOMSDA	MCMO	OBAL (ours)
SEA	50.07	52.45	49.83	52.31	51.47	50.07	3.01	51.30	43.79	16.67
Tree	37.76	36.13	35.87	40.42	39.65	39.43	4.17	37.79	41.04	14.93
RBF	32.57	33.14	33.76	43.01	42.35	42.50	3.04	43.10	35.04	16.87
Hyperplane	57.41	55.73	56.20	52.38	53.09	53.42	4.10	54.75	57.17	19.37
Weather	60.41	59.14	57.23	52.01	51.49	52.37	4.97	51.70	89.43	10.71
Kitti	71.23	71.75	70.93	94.87	96.21	95.43	10.43	48.42	77.85	45.11
CNNIBN	409.58	412.37	408.60	367.54	359.40	355.90	53.15	354.56	532.94	233.25
BBC	424.20	407.18	431.47	419.87	417.76	420.39	53.01	424.15	541.02	287.93

Moreover, Table 3.5 shows the comparison of execution time, which showcases a competitive runtime. Melanie achieved the swiftest execution, followed by our method. OBAL's occasional inefficiency arises from its need for model re-initialization upon target drift detection. When benchmarked against neural network approaches like AOMSDA and ATL, OBAL stands out in computational efficiency. Compared to ensemble strategies like FUSION and MCMO, OBAL still showcases a competitive runtime due to its closed-form solution for distribution alignment.

3.4 Summary

In this chapter, we have addressed a significant gap in multistream classification, where the dynamic relationships between streams have largely been overlooked. This oversight can often result in the issue of negative transfer stemming from irrelevant data. To overcome this challenge, we introduced the Online Boosting Adaptive Learning (OBAL) method, coupled with the proposed AdaCOSA algorithm, effectively exploring the dynamic correlation among various streams. The experiments performed on several synthetic and real-world data streams have shown that our method effectively navigates the dynamic correlations between streams, mitigates covariate shifts, and adeptly handles asynchronous drift using a GMM-based weighting mechanism. The insights gained from this study not only advance the field of multistream classification but also provide a promising direction for future research in adaptive learning across various dynamic data environments.

CHAPTER 4

FUZZY SHARED REPRESENTATION LEARNING FOR MULTISTREAM CLASSIFICATION

The advancement of multistream classification is challenged by 1) guaranteed joint representations of different streams, 2) uncertainty and interpretability during knowledge transfer, and 3) tracking and adapting the asynchronous drifts in each stream. To solve RQ2 and achieve RO2, this chapter proposes an interpretable Fuzzy Shared Representation Learning (FSRL) method based on the Takagi-Sugeno-Kang (TSK) fuzzy system. Following that, window-based and GMM-based online adaptation strategies are designed to address the asynchronous drifts over time. The former can directly demonstrate the effectiveness of FSRL in knowledge transfer across multiple streams, while the GMM-based method offers an informed way to overcome the asynchronous drift problem by integrating drift detection and adaptation. In this chapter, the motivations and backgrounds are introduced in Section 4.1. Definitions, preliminaries, and proposed FSRL are introduced in Section 4.2. The proposed method is validated in Section 4.3. Section 4.4 summarizes this chapter.

4.1 Introduction

In machine learning, there is a common assumption that optimal model performance is contingent upon the training and test datasets adhering to identical distributions, thereby enabling the model to generalize effectively. However, in real-world applications, various data are always generated continuously and sequentially with unpredictable changes in their underlying distribution. The phenomenon is referred to as concept drift Lu et al. (2018a); Zhou et al. (2023c); Wang et al. (2022d); Yu et al. (2020b), which results in a decline in the performance of models trained on historical data once drift occurs. Consequently, researchers have shown significant interest in devising effective learning techniques to analyze streaming data in non-stationary environments. The objective of this research is to address the challenges posed by concept drift and enhance the model's adaptability to the continuously evolving data distributions in dynamic real-world environments.

Existing studies have provided empirical evidence regarding the efficacy of concept drift adaptation methods for handling dynamic distributions in data streams Jiao et al. (2022a); Liu et al. (2020a); Miyaguchi and Kajino (2019); Hinder et al. (2023); Bai et al. (2023). Research in this domain predominantly falls into two categories: informed and blind. Informed methods typically leverage supervised or unsupervised drift detection strategies to dynamically monitor data streams. Upon detecting a drift, an adaptation mechanism will be triggered, enabling the model to quickly adapt to the new concept Abadifard et al. (2023); Gama and Castillo (2006). In contrast, blind methods continuously update the model with new incoming data without explicitly employing drift detection mechanisms Yang et al. (2021); Renchunzi and Pratama (2022). It's important to note, however, that a considerable number of these approaches are specifically designed for single-stream scenarios with delayed labels.

In the realm of advanced intelligent systems, it is quite typical to encounter the

concurrent generation of multiple data streams simultaneously. For example, in a healthcare monitoring system, multiple data streams like patient vital signs, electronic health records, and wearable device data vary over time but interact to facilitate critical medical decisions. These various data streams enable personalized patient care and real-time health assessments, despite exhibiting distinct distributions due to varying data sources Wang et al. (2023a); Song et al. (2021b). In addition, the labeling of this multifaceted data, essential for facilitating critical medical decisions and enabling personalized patient care, presents significant challenges in terms of time and labor costs. This has led to the emergence of hybrid data environments, where voluminous streams of both labeled and unlabeled data are processed concurrently Yu et al. (2022b); Jiao et al. (2022b).

To address this situation, multistream classification has been introduced, involving both labeled and unlabeled data streams with concept drifts Chandra et al. (2016); Dong et al. (2019); Tao et al. (2019). This task aims to predict the labels of the target stream by transferring knowledge from one or multiple labeled source streams, while also handling the concept drift problem. Generally, there are three common challenges in this task Haque et al. (2017): *1) Label scarcity*, which refers to the lack of labels for the target data stream, unlike the source streams that have labeled data; *2) Covariate shift*, which indicates that each data stream exhibits a unique distribution, differing from others and *3) Asynchronous concept drift*, which occurs when source and target streams independently experience concept drift at different times, uniquely impacting the model's performance. In recent years, some methods have been proposed to address these challenges by integrating online domain adaptation and drift-handling techniques. Most of them predominantly concentrate on single-source streams, which can hamper model performance due to the constraints in source data quality. Additionally, these approaches, being reliant on a single source, often face the risk of overfitting. To address these issues, a multi-source configuration has been introduced, which taps into diverse

source streams to gather supplementary data, thereby enriching the model's accuracy and robustness Wang et al. (2022g); Yang et al. (2021).

However, transferring knowledge among multiple non-stationary data streams presents additional challenges that are not considered in current works. Firstly, most current methodologies construct classifiers based on information from the original feature space, overlooking the impact of redundant or low-quality features, which can detrimentally affect the final decision-making process Jiao et al. (2022b). Therefore, a method **to exploit guaranteed joint representations of different streams** becomes particularly crucial during knowledge transfer. Secondly, various data streams with asynchronous drifts inevitably contain inherent uncertainties and dynamic relationships. Consequently, a method **to address these uncertainties and provide a reasonable interpretability analysis** is also a crucial focus of this task.

Since fuzzy systems have shown robust learning capabilities and transparent interpretability for knowledge transfer Lu et al. (2019); Li et al. (2023a); Xu et al. (2019), we propose a novel Fuzzy Shared Representation Learning (FSRL) method to address the newly exposed challenges. TSK fuzzy system is a data-driven system comprised of multiple IF-THEN fuzzy rules, offering high interpretability. It can learn model parameters in a data-driven manner similar to other machine learning methods. The output of a TSK fuzzy system is based on linear functions of the inputs, providing a more flexible way to learn shared representations. This flexibility allows for better adaptation to various tasks and data types. Many methods based on TSK fuzzy systems have been proposed to enhance interpretability and address uncertainty in transfer learning, which also provides a guaranteed foundation for the design of my model Yang et al. (2015); Xu et al. (2019); Zhang et al. (2023).

Therefore, FSRL employs a multioutput TSK fuzzy system as a transformation method to learn a shared fuzzy space. Specifically, as shown in Figure 4.1, we collect a

data batch from each data stream and construct a fuzzy mapping with the antecedent part of the TSK fuzzy system individually to realize nonlinear transformation. It preserves discriminative information for each original stream in an interpretable way. Then, an advanced multistream joint distribution adaptation method is proposed to learn the consequent parameters of the TSK fuzzy system by considering all data streams simultaneously. Following that, blind window-based (BFSRL) and informed GMM-based (IFSRL) adaptation strategies are designed to address the various drifting situations over time. Specifically, BFSRL utilizes a fixed-size sliding window to aggregate incoming data, implementing FSRL at each step. This method not only effectively demonstrates FSRL's capability in facilitating knowledge transfer across multiple streams but also provides a direct solution to the challenges in multistream classification. However, due to the continuous variation in the frequency and types of data drifting in different datasets, it is sensitive to window size.

To enhance the robustness and generalization, we further introduce the IFSRL, which integrates drift detection and adaptation mechanisms, providing a more nuanced response to the concept drift problem. As shown in Figure 4.2, we initially train a Gaussian Mixture Model (GMM) Oliveira et al. (2021) using collected data, which is employed to estimate the conditional distributional relationship between new data and current data. Additionally, we employ the Drift Detection Method (DDM) Gama et al. (2004) to detect the drift in each labeled source stream, as it offers a stable and accurate detection approach. Simultaneously, we utilize the weighted ensemble strategy for asynchronous drift adaptation leveraging the conditional distributions derived from the GMM. For the unlabeled target stream, we design two sliding windows and continuously monitor their distribution changes to effectively detect drift occurrences based on the mean conditional distribution. When drift is detected in the target stream, it indicates that the previously established fuzzy shared space, learned through FSRL, is no longer

valid. This necessitates the re-collection of data and the learning of a new fuzzy shared space, ensuring our model's adaptability and relevance in the face of evolving data characteristics.

4.2 Proposed Method

In this section, we first define the multisource stream classification problem and analyze the challenges inherent in this task, as well as its objectives. Then, we provide an overview of our proposed Fuzzy Shared Representation Learning (FSRL) method. Subsequently, we delve into the detailed description of the FSRL algorithm and the optimization strategy. Furthermore, we discuss in depth two online adaptation strategies designed to handle different drifting scenarios over time, i.e., blind window-based and informed GMM-based approaches.

4.2.1 Problem Definition and Overall Framework

Multisource stream classification is an online process involving V distinct labeled source streams $S_v, v \in \{1,2,\ldots,V\}$ and a single target stream T with *scarcity of labels*. Denoting P_{S_v} and P_T as the distributions from S_v and T. Each arrived data sample from source streams at time i is represented by $\mathbf{x}_{S_{vi}} \in \mathscr{D}^d$ with true label $y_{S_{vi}}$, while only d-dimensional features $\mathbf{x}_{T_i}$ of target stream can be obtained. All streams at the same time step i are related but with *joint domain shift*, i.e., 1) marginal shift $P_{Sv}(\mathbf{x}_{S_{vi}}) \neq P_{Sv^*}(\mathbf{x}_{S_{v^*i}}) \neq P_T(\mathbf{x}_{T_i})$ and 2) conditional shift $Q_{Sv}(y_{S_{vi}} \mid \mathbf{x}_{S_{vi}}) \neq Q_{Sv^*}(y_{S_{v^*i}} \mid \mathbf{x}_{S_{v^*i}}) \neq Q_T(y_{T_i} \mid \mathbf{x}_{T_i})$ and . In addition, another challenging issue of this setup is the *asynchronous concept drift* over time, which can manifest in three primary scenarios: source drift, target drift and concurrent drift, as detailed in Section 3.2.1.

Our primary goal is to precisely predict the labels of the target stream by adeptly harnessing the insights gleaned from the labeled source streams. This entails not only

utilizing the available knowledge from the source streams but also adeptly navigating and adapting to various drifting scenarios. Furthermore, our attention is centered on developing methods to secure more robust and dependable joint representations in the midst of transferring knowledge across diverse streams. Simultaneously, we aim to identify and implement strategies to circumvent potential uncertainties, thereby enhancing the overall efficacy and reliability of our approach. This focus is integral to advancing the precision and effectiveness of our methods within the context of dynamic stream environments.

Therefore, we propose the FSRL method based on the multioutput TSK fuzzy system. It learns a common feature subspace for various data streams, in which both the unique discriminative information of each original stream and the distribution-adapted common features are simultaneously preserved and emphasized. Furthermore, FSRL delves into deciphering nonlinear interactions among multiple data streams via the rule-based TSK fuzzy system. It facilitates the achievement of promising performance in terms of robustness and interpretability. Following that, blind window-based (BFSRL) and informed GMM-based (IFSRL) online adaptation strategies are designed to address the various drifting situations over time based on the learned common features.

4.2.2 Takagi-Sugeno-Kang Fuzzy System

The TSK fuzzy system Takagi and Sugeno (1985), an advanced model within the realm of fuzzy logic, stands out for its data-driven methodology and application of IF-THEN fuzzy rules Zhang et al. (2023). This approach facilitates the construction of models that are not only robust in learning from complex data patterns but also excel in providing transparent and interpretable results. The versatility of the TSK fuzzy system is evidenced by its successful deployment across a wide range of fields, from control systems to pattern recognition and beyond. Specifically, it can be can be formulated based on

‚ÄúIF-THEN‚Äù rules as follows:

$$\begin{aligned} &\text{IF}: x_1 \,\text{is}\, A_1^k \wedge x_2 \,\text{is}\, A_2^k \wedge \cdots \wedge x_d \,\text{is}\, A_d^k \\ &THEN: f^k(\mathbf{x}) = p_0^k + p_1^k x_1 + \cdots + p_d^k x_d \end{aligned} \tag{4.1}$$

where $k \in \{1,2,\ldots,K\}$ denotes the index of the k-th rule. Let $\mathbf{x} \in \mathbb{R}^{d\times 1}$ represent the sample vector, where d is the dimensionality of the samples. The function $f^k(\mathbf{x})$ signifies the output of the k-th rule. Additionally, A_i^k denotes the fuzzy set associated with the i-th feature under the k-th rule, and $\wedge$ symbolizes a fuzzy conjunction operator.

In contrast to crisp sets where membership values are strictly confined to either 0 or 1, fuzzy sets allow for membership degrees ranging continuously from 0 to 1. A frequently employed representation of fuzzy membership is through the Gaussian function, which is formulated as,

$$\mu_{A_i^k}(x_i) = \exp\left(-(x_i - c_i^k)/2\delta_i^k\right)^2, \tag{4.2}$$

where parameters c_i^k and δ_i^k represent the center and width of the Gaussian function and they are also the antecedent parameters of the TSK fuzzy system. The estimation of these parameters can be effectively achieved through various methodologies, such as the Fuzzy C-Means Clustering (FCM) Yang et al. (2015) and Var-Part Su and Dy (2007).

Once the antecedent parameters are established, it becomes feasible to compute the membership value for each feature in the specific fuzzy set A_i^k, as delineated in Eq.(4.2). When multiplication serves as the conjunction operator, the firing level of each sample's k-th rule can be determined via Eq.(4.3a). Eq.(4.3b) presents the normalized version of this calculation. Further, the output of the TSK fuzzy system is represented as the

weighted mean of $f_k(\mathbf{x})$, as elucidated in Eq.(4.3c).

$$\mu^k(\mathbf{x}) = \prod_{i=1}^{d} \mu_{A_i^k}(x_i) \tag{4.3a}$$

$$\tilde{\mu}^k(\mathbf{x}) = \mu^k(\mathbf{x}) \Big/ \sum_{k'=1}^{K} \mu^{k'}(\mathbf{x}) \tag{4.3b}$$

$$f(\mathbf{x}) = \sum_{k=1}^{K} \tilde{\mu}^k(\mathbf{x}) f^k(\mathbf{x}) \tag{4.3c}$$

Upon acquiring the antecedent parameters, the expression of the TSK fuzzy system's output, as indicated in Eq.(4.3c), can be represented as linear regression, as delineated in Eq.(4.4).

$$y = f(\mathbf{x}) = \mathbf{p}_g^{\mathrm{T}} \mathbf{x}_g \tag{4.4}$$

where

$$\mathbf{x}_e = [1, \mathbf{x}^{\mathrm{T}}]^{\mathrm{T}} \in R^{(d+1)\times 1} \tag{4.5a}$$

$$\tilde{\mathbf{x}}^k = \tilde{\mu}^k(\mathbf{x}) \mathbf{x}_e \in R^{(d+1)\times 1} \tag{4.5b}$$

$$\mathbf{x}_g = [(\tilde{\mathbf{x}}^1)^{\mathrm{T}}, (\tilde{\mathbf{x}}^2)^{\mathrm{T}}, \ldots, (\tilde{\mathbf{x}}^K)^{\mathrm{T}}]^{\mathrm{T}} \in R^{K(d+1)\times 1} \tag{4.5c}$$

$$\mathbf{p}^k = [p_0^k, p_1^k, \cdots, p_d^k]^{\mathrm{T}} \in R^{(d+1)\times 1} \tag{4.5d}$$

and

$$\mathbf{p}_g = [(\mathbf{p}^1)^{\mathrm{T}}, (\mathbf{p}^2)^{\mathrm{T}}, \ldots, (\mathbf{p}^K)^{\mathrm{T}}]^{\mathrm{T}} \in R^{K(d+1)\times 1}. \tag{4.6}$$

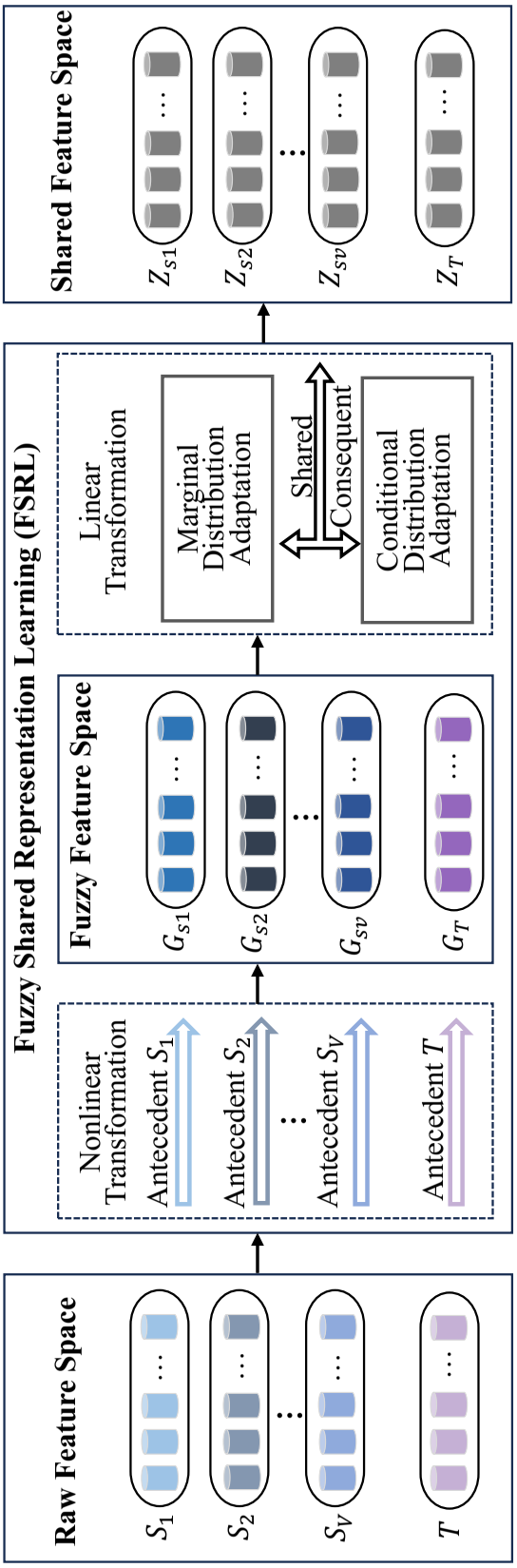


Figure 4.1: The high-level illustration of FSRL. It first collects data batches from all data streams and constructs a fuzzy mapping with the antecedent part of the TSK fuzzy system individually to realize nonlinear transformation. Then, the consequent components are optimized by aligning the conditional and marginal distributions simultaneously.

4.2.3 Fuzzy Shared Representation Learning

As previously discussed, prevalent approaches in multistream classification predominantly rely on the original feature space for constructing classifiers. This often leads to the inadvertent inclusion of redundant or low-quality features, potentially compromising the efficacy of the decision-making process Jiao et al. (2022b). To address this issue, our methodology involves the creation of a fuzzy shared space, which encompasses a two-step process for deriving fuzzy shared representations. This process comprises an initial nonlinear transformation, followed by a linear dimensionality reduction.

In our proposed method, the nonlinear transformation within the multioutput TSK fuzzy system is facilitated by its antecedent parameters. For this purpose, the Gaussian function is employed as the membership function, where the antecedent parameters are specifically designed to represent the center and the width of the Gaussian curve. Following this, the Var-Part clustering method Su and Dy (2007) is applied to estimate the clusters $\mathbf{C_{Sv}}$ and $\mathbf{C_T}$, corresponding to each source and target stream, respectively. Subsequently, the kernel width matrices $\mathbf{D_{Sv}}$ and $\mathbf{D_T}$ are derived, utilizing a procedure analogous to that used in the FCM Yang et al. (2015) approach as follows:

$$(\mathbf{D}_{Sv})_p^k = \sum_{i=1}^{n_{Sv}} \left(x_{S_{vip}} - (\mathbf{C}_{Sv})_p^k\right)^2, v = 1,2,\cdots,V \tag{4.7a}$$

$$(\mathbf{D}_T)_p^k = \sum_{i=1}^{n_T} \left(x_{T_{ip}} - (\mathbf{C}_T)_p^k\right)^2, \tag{4.7b}$$

where $k = 1,2,\ldots,K$ represents the total number of rules, while $p = 1,2,\ldots,d$ denotes the dimension of samples. n_{Sv} and n_T signify the respective numbers of samples within each source and target stream. Notably, the estimation of clusters and kernel widths is conducted individually across different streams. This approach ensures that the discriminative information pertinent to each original stream is not only preserved but also rendered in an interpretable manner.

Once all the antecedent parameters of the TSK fuzzy system are established, any

sample ($\mathbf{x}_{S_{vi}}$ or $\mathbf{x}_{Ti}$) from either the source stream or the target stream is first mapped into a stream-specific fuzzy space via Eq. (4.5a)–(4.5c). The new representations of the source and target samples in the fuzzy space can be formulated as follows:

$$\mathbf{g}_{S_{vi}} = [(\tilde{\mathbf{x}}^1_{S_{vi}})^{\mathrm{T}}, (\tilde{\mathbf{x}}^2_{S_{vi}})^{\mathrm{T}}, \ldots, (\tilde{\mathbf{x}}^K_{S_{vi}})^{\mathrm{T}}]^{\mathrm{T}} \in R^{K(d+1)\times 1}, \quad v = 1, 2, \cdots, V \tag{4.8a}$$

$$\mathbf{g}_{T_i} = [(\tilde{\mathbf{x}}^1_{T_i})^{\mathrm{T}}, (\tilde{\mathbf{x}}^2_{T_i})^{\mathrm{T}}, \ldots, (\tilde{\mathbf{x}}^K_{T_i})^{\mathrm{T}}]^{\mathrm{T}} \in R^{K(d+1)\times 1} \tag{4.8b}$$

Subsequently, the consequent components of the TSK fuzzy system are employed as the transformation function $\phi(*)$. This function is instrumental in facilitating linear dimensionality reduction within the resultant shared fuzzy feature space, which is formulated as follows:

$$\phi(\mathbf{x}_{S_{vi}}) = \mathbf{z}_{S_{v_i}} = \mathbf{P}^T \mathbf{g}_{S_{vi}}, v = 1, 2, \ldots V \tag{4.9a}$$

$$\phi(\mathbf{x}_{T_i}) = \mathbf{z}_{T_i} = \mathbf{P}^T \mathbf{g}_{T_i} \tag{4.9b}$$

$$\mathbf{P} = [\mathbf{p}^1_g, \mathbf{p}^2_g, \ldots, \mathbf{p}^m_g] \in R^{K(d+1)\times m} \tag{4.9c}$$

In this study, distinct from the design for antecedent parameters, we postulate that the consequent parameters across different streams are shared. This assumption aids in streamlining the feature set, thereby enhancing computational efficiency and improving the overall performance of the model. Additionally, it facilitates the discovery of common representations among various data streams.

Due to the presence of joint domain shifts among different data streams, it becomes imperative to consider this factor while learning the consequent parameters. This consideration aims to minimize the distribution differences: i.e., 1) marginal distribution $P_{Sv}(\mathbf{x}_{S_{vi}})$ and $P_T(\mathbf{x}_{T_i})$, and 2) conditional distribution $Q_{Sv}(y_{S_{vi}} \mid \mathbf{x}_{S_{vi}})$ and $Q_T(y_{T_i} \mid \mathbf{x}_{T_i})$, in the final shared fuzzy space. Therefore, we further propose a multistream joint distribution adaptation method in our study. Given the transformation function $\phi(*)$, it aims

at matching the joint expectations between each source stream and target stream,

$$
\begin{aligned}
&\min_{\phi} \sum_{v=1}^{V} \left(\left\| \mathbb{E}_P[\phi(\mathbf{x}_{S_{vi}}), y_{Svi}] \; - \mathbb{E}_P\left[\phi(\mathbf{x}_{T_i}), y_{Ti}\right] \right\|^2 \right) \\
&\approx \sum_{v=1}^{V} \left(\left\| \mathbb{E}_{P_{Svi}}\left[\phi(\mathbf{x}_{S_{vi}})\right] \; - \mathbb{E}_{P_T}\left[\phi(\mathbf{x}_{T_i})\right] \right\|^2 \right. \\
&\left. + \left\| \mathbb{E}_{Q_{Si}}\left[y_{Svi} \mid \phi(\mathbf{x}_{S_{vi}})\right] \; - \mathbb{E}_{Q_T}\left[y_{Ti} \mid \phi(\mathbf{x}_{T_i})\right] \right\|^2 \right)
\end{aligned}
\tag{4.10}
$$

4.2.3.1 Marginal Distribution Adaptation

To align marginal distributions between source streams and target stream, we regard the Maximum Mean Discrepancy (MMD) Long et al. (2017) as the distance:

$$
\begin{aligned}
&\sum_{v=1}^{V} \left(\left\| \frac{1}{n_{S_v}} \sum_{i=1}^{n_{S_v}} \mathbf{P}^{\mathrm{T}} \mathbf{g}_{S_{vi}} - \frac{1}{n_T} \sum_{j=1}^{n_T} \mathbf{P}^{\mathrm{T}} \mathbf{g}_{T_j} \right\|^2 \right) \\
&= \mathrm{tr}\left(\mathbf{P}^{\mathrm{T}} \mathbf{G}_X \mathbf{M}^0 \mathbf{G}_X^{\mathrm{T}} \mathbf{P} \right)
\end{aligned}
\tag{4.11}
$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix. $\mathbf{G}_X = [\mathbf{G}_{S_1}, \ldots, \mathbf{G}_{S_V}, \mathbf{G}_T] \in R^{K(d+1)\times(V\times n_{S_v}+n_T)}$ and $\mathbf{G}_{S_v} = [\mathbf{g}_{S_{v_1}}, \mathbf{g}_{S_{v_2}}, \ldots, \mathbf{g}_{S_{v_{n_{S_v}}}}] \in R^{K(d+1)\times n_{S_v}}$ and $\mathbf{G}_T = [\mathbf{g}_{T_1}, \mathbf{g}_{T_2}, \ldots, \mathbf{g}_{T_{n_T}}] \in R^{K(d+1)\times n_T}$. $\mathbf{M}^0 \in R^{(V\times n_{S_v}+n_T)^2}$ is the MMD matrix, which is formulated as:

$$
\left(\mathbf{M}^0\right)_{ij} = \begin{cases} \frac{1}{n_S n_S}, & i,j \le n_S \\ \frac{1}{n_T n_T}, & i,j > n_T \\ \frac{-1}{n_S n_T}, & \text{otherwise} \end{cases}
\tag{4.12}
$$

where $n_S = V \times n_{S_v}$ is the total number of V source streams.

4.2.3.2 Conditional Distribution Adaptation

Although minimizing the discrepancies in the marginal distributions across domains can be beneficial, this alone doesn't ensure the alignment of the conditional distributions $Q_{S_v}(y_{S_{vi}} \mid \mathbf{x}_{S_{vi}})$ and $Q_T(y_{T_i} \mid \mathbf{x}_{T_i})$. It is imperative to focus on reducing the differences between these conditional distributions for effective and robust distribution adaptation in the context of domain adaptation Wang et al. (2022b). However, due to the scarcity

of labels in the target stream, it is impractical to directly estimate the conditional distribution $Q_T(y_{T_i} \mid \mathbf{x}_{T_i})$ of the target stream. Consequently, we propose an assumption: $Q_{S_v}(y_{S_{vi}} \mid \mathbf{x}_{S_{vi}}) \approx Q_T(y_{T_i} \mid \mathbf{x}_{T_i})$. This allows us to apply the classifiers trained on source streams to predict pseudo labels $\hat{y}_T$ for the target stream Long et al. (2017). To enhance accuracy, we adopt an iterative approach to update the classifier and the feature transformation $\phi(*)$. Since the true labels for source streams and the pseudo labels for target stream have been obtained, the MMD distance between class-conditional distributions $Q_{S_v}(\mathbf{x}_{S_{vi}} \mid y_{S_{vi}} = c)$ and $Q_T(\mathbf{x}_{T_i} \mid y_{T_i} = c)$ can be measured by:

$$\begin{aligned} &\sum_{v=1}^{V}\sum_{c=1}^{C}\left\| \frac{1}{n_{S_v}^{(c)}} \sum_{y_{S_{vi}}=c} \mathbf{P}^{\mathrm{T}}\mathbf{g}(\mathbf{x}_{S_{vi}}) - \frac{1}{n_T^{(c)}} \sum_{\hat{y}_{T_i}=c} \mathbf{P}^{\mathrm{T}}\mathbf{g}(\mathbf{x}_{T_i}) \right\|^2 \\ &= \mathrm{tr}\left(\mathbf{P}^{\mathrm{T}}\mathbf{G}_X \sum_{c=1}^{C} \mathbf{M}^c \mathbf{G}_X^{\mathrm{T}} \mathbf{P} \right) \end{aligned} \tag{4.13}$$

where $c = 1, 2, \ldots C$ is the class numbers. $n_S^{(c)}$ and $n_T^{(c)}$ represent the number of examples belonging to the c-th class in all source streams and the target stream, respectively. $\mathbf{M}^c$ is defined as:

$$\left(\mathbf{M}^c\right)_{ij} = \begin{cases} \frac{1}{n_S^{(c)} n_S^{(c)}}, & i,j \le n_S \wedge y_{S_i}, y_{S_j} = c \\ \frac{1}{n_T^{(c)} n_T^{(c)}}, & i,j > n_S \wedge \hat{y}_{T_{(i-n_S)}}, \hat{y}_{T_{(j-n_S)}} = c \\ \frac{-1}{n_S^{(c)} n_T^{(c)}}, & \begin{cases} i \le n_S, j > n_S \wedge \hat{y}_{S_i}, \hat{y}_{T_{(j-n_S)}} = c \\ i > n_S, j \le n_s \wedge \hat{y}_{T_{(i-n_S)}}, \hat{y}_{S_j} = c \end{cases} \\ 0, & \text{otherwise} \end{cases} \tag{4.14}$$

4.2.3.3 Overall objective function and optimization

The subsequent parameters in P can be derived using the conventional Principal Component Analysis (PCA) methodology by incorporating both Eq.(4.11) and Eq.(4.13). There-

fore, the overall objective function can be formulated as:

$$\begin{aligned} &\min_{P} \sum_{c=0}^{C} \operatorname{tr}\left(\mathbf{P}^{\mathrm{T}}\mathbf{G}_X\mathbf{M}^c\mathbf{G}_X^{\mathrm{T}}\mathbf{P}\right) + \lambda\|\mathbf{P}\|_F^2 \\ &s.t. \qquad \mathbf{P}^{\mathrm{T}}\mathbf{G}_X\mathbf{H}\mathbf{G}_X^{\mathrm{T}}\mathbf{P} = \mathbf{I} \end{aligned} \tag{4.15}$$

where $\|\mathbf{P}\|_F^2$ is a constraint term to avoid overfitting and λ is the corresponding regularization parameter and it is set as 1 in this method. $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}$ represents the centering matrix and $n = V \times n_{S_v} + n_T$. $\mathbf{I} \in R^{K(d+1)\times K(d+1)}$ is a identity matrix.

According to the constrained optimization theory, we denote $\mathbf{\Phi} = \operatorname{diag}(\phi_1, \ldots, \phi_m) \in \mathbb{R}^{m\times m}$ as the Lagrange multiplier. Consequently, the Lagrangian function related to Eq.(4.15) can be derived as:

$$\begin{aligned} L = &\operatorname{tr}\left(\mathbf{P}^{\mathrm{T}}\left(\mathbf{G}_X\sum_{c=0}^{C}\mathbf{M}^c\mathbf{G}_X^{\mathrm{T}} + \lambda\mathbf{I}\right)\mathbf{P}\right) \\ &+ \operatorname{tr}\left(\left(\mathbf{I} - \mathbf{P}^{\mathrm{T}}\mathbf{G}_X\mathbf{H}\mathbf{G}_X^{\mathrm{T}}\mathbf{P}\right)\Phi\right) \end{aligned} \tag{4.16}$$

By setting $\frac{\partial L}{\partial \mathbf{P}} = 0$, we obtain generalized eigendecomposition form:

$$\left(\mathbf{G}_X\sum_{c=0}^{C}\mathbf{M}^c\mathbf{G}_X^{\mathrm{T}} + \lambda\mathbf{I}\right)\mathbf{P} = \mathbf{G}_X\mathbf{H}\mathbf{G}_X^{\mathrm{T}}\mathbf{P}\mathbf{\Phi} \tag{4.17}$$

Ultimately, the task of identifying the optimal transformation matrix $\mathbf{P}$ simplifies to resolving Eq.(4.17), specifically targeting the m smallest eigenvectors. In this context, $\phi_1, \ldots, \phi_m$ represent the minimal eigenvalues, with the corresponding eigenvectors denoted as $\mathbf{P} = [\mathbf{p}_g^1, \mathbf{p}_g^2, \ldots, \mathbf{p}_g^m]$. Once the consequent parameters $\mathbf{P}$ are acquired, it becomes feasible to derive new representations for each source stream and the target stream in the final fuzzy shared representation space. The detailed learning process is summarized in Algorithm 3.

In Algorithm 3, it is assumed that each stream comprises n samples, resulting in a total of $N = (V+1)n$ samples. The computational complexity of step 1 and step 2 is denoted as $O(2dNK)$ and $O((1+d)NK)$, respectively. Subsequently, assuming I iterations for multistream joint distribution adaptation and a final data dimensionality of $\hat{d}$ in the

fuzzy shared representation space, the computational complexities in steps 5, 6, and 7 are denoted as $\hat{d}m^2d^2$, $N\hat{d}d$, and CN^2, respectively. Therefore, the overall complexity of the proposed FSRL is expressed as $O(NK(3d+1)+I(\hat{d}m^2d^2+N\hat{d}d+CN^2))$.

4.2.4 Blind Window-based Adaptation (BFSRL)

The proposed BFSRL is a straightforward yet highly effective approach. In this strategy, a set of sliding windows W_{S_v} and W_T with size n are crafted to collect the incoming data from source and target streams, respectively. This framework facilitates the joint representation learning of the collected data via FSRL, paving the way for subsequent label prediction tasks for the target stream. Specifically, within any given sliding window, the source data $W_{S_v} = [\mathbf{x}_{S_{v_1}}, \mathbf{x}_{S_{v_2}}, \ldots, \mathbf{x}_{S_{v_n}}], v \in \{1,2,\ldots,V\}$ and target data $W_T = [\mathbf{x}_{T_1}, \mathbf{x}_{T_2}, \ldots, \mathbf{x}_{T_n}]$ are procured. Subsequently, stream-specific fuzzy representations are derived using Eq.(4.3a) and (4.3b) as follows:

$$\mathbf{G}_{W_{S_v}} = [\mathbf{g}_{S_{v_1}}, \mathbf{g}_{S_{v_2}}, \ldots, \mathbf{g}_{S_{v_n}}], v = 1,2,\ldots,V \tag{4.18a}$$

$$\mathbf{G}_{W_T} = [\mathbf{g}_{T_1}, \mathbf{g}_{T_2}, \ldots, \mathbf{g}_{T_n}] \tag{4.18b}$$

Subsequently, the consequent parameters P_W are optimized by employing the optimization criteria defined in Equation (4.15). This optimization facilitates the extraction of the final fuzzy shared representations for each window, delineated as follows:

$$\mathbf{Z}_{W_{S_v}} = \mathbf{P}^T \mathbf{G}_{W_{S_v}}, v = 1,2,\ldots,V \tag{4.19a}$$

$$\mathbf{Z}_{W_T} = \mathbf{P}^T \mathbf{G}_{W_T} \tag{4.19b}$$

Ultimately, classifiers f_{S_v} are trained on the entirety of the fuzzy shared representations derived from the source streams. The aggregate predictive probability is then obtained by an ensemble, where each classifier's contribution is weighted and averaged, formalized as:

$$f^E(\mathbf{Z}_{W_T}) = \frac{1}{V}\sum_{v=1}^{V} f_{S_v}(\mathbf{Z}_{W_T}) \tag{4.20}$$

Algorithm 3 The learning process of FSRL

Input: Input data $\{(\mathbf{x}_{S_{vi}}, y_{S_{vi}})\}_{i=1}^{n}$, $\{\mathbf{x}_{T_i}\}_{i=1}^{n}$; the number of fuzzy rules K; hyperparameters λ

Output: Antecedent parameters of the TSK fuzzy systems; consequent transformation matrix $\mathbf{P}$.

1: Estimate the antecedent parameters of the TSK fuzzy systems by using the Var-Part clustering algorithm for each stream.
2: Construct the new fuzzy representations $\{\mathbf{g}_{S_{v_i}}\}_{i=1}^{n}$ and $\{\mathbf{g}_{T_i}\}_{i=1}^{n}$ generated by fuzzy rules for each stream by Eq. (4.5a)-(4.5c).
3: Construct MMD matrix $\mathbf{M}^0$ by Eq.(4.12)
4: **repeat**
5: Solve the generalized eigendecomposition problem in Eq.(4.17) and select the m smallest eigenvectors to construct $\mathbf{P}$.
6: Train a classifier f using new fuzzy representations $\{(\mathbf{z}_{S_{v_i}}, y_{S_{vi}})\}_{i=1}^{N}$ to update pseudo target labels.
7: Construct MMD matrices $\mathbf{M}^c$ by Eq.(4.14).
8: **until** Convergence

This approach not only efficaciously showcases the potential of the proposed FSRL in enabling knowledge transfer across various streams but also offers a direct resolution to the complexities inherent in multistream classification.

4.2.5 Informed GMM-based Adaptation (IFSRL)

Although BFSRL offers a straightforward and effective approach, it is highly sensitive to the window size due to the varying frequencies and types of concept drift in different datasets. This variability can lead to challenges in consistently applying this method across diverse drifting conditions. To enhance the generalization and robustness of the proposed method, we introduce the IFSRL, which integrates drift detection and fuzzy shared representation learning, providing a more nuanced response to the concept drift problem.

In this process, similar to window-based approaches, sliding windows W_{S_v} and W_T with size n are crafted to collect the incoming data from source and target streams, respectively. Subsequently, we obtain an initial set of antecedent parameters and the

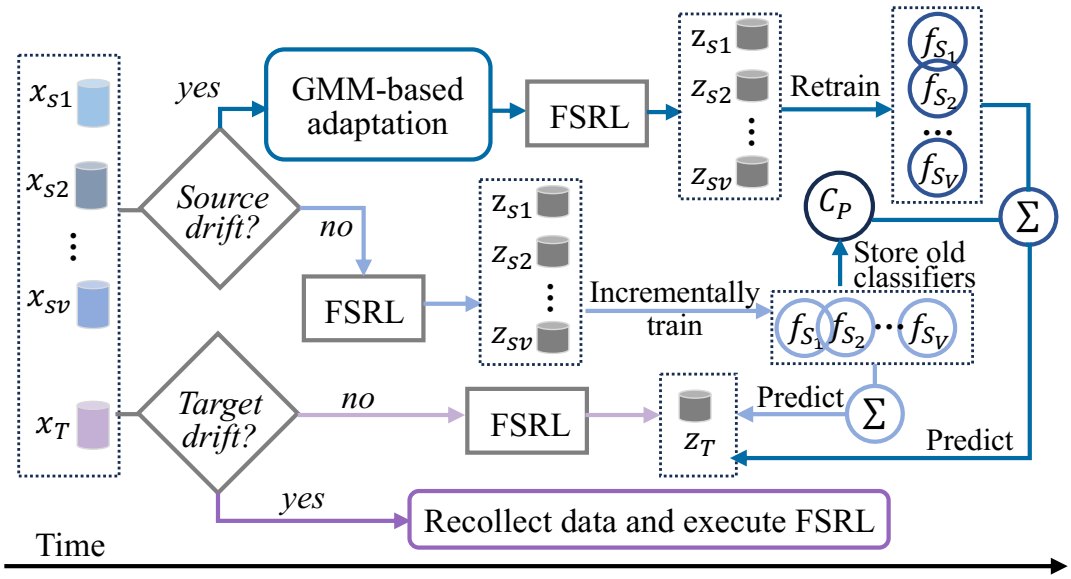


Figure 4.2: The IFSRL framework involves using DDM for drift detection in source streams. New samples are mapped to the learned fuzzy shared space for incremental base classifier training when there is no drift. If drift occurs, the GMM-based adaptation module adjusts the data and retrains a new base classifier, while preserving old base classifiers in a pool for retaining previous knowledge. Drift in the target stream is monitored using two sliding windows based on the mean conditional distribution, triggering data re-collection and new fuzzy shared space learning once target drift is detected.

shared consequent parameters $\mathbf{P}$ as well as base classifiers f_{S_v} via FSRL. Then, we conduct online detection and adaptation based on different asynchronous drift scenarios.

4.2.5.1 Drift Detection

The selection of appropriate drift detection methods hinges on the availability of labels. For labeled source streams, we use the DDM Gama et al. (2004) to detect the drift due to its accuracy and stability. When a new source sample $\mathbf{x}_{S_{vi}}$ arrives, its predicted label updates the drift detector based on prediction error. Drift is detected if the error rate exceeds a set warning threshold. This provides a reliable and efficient mechanism for monitoring drift in dynamic data streams.

However, due to the absence of labels in the target data, we are constrained to employing unsupervised methods for drift detection. Our strategy for identifying drift in

the target stream involves vigilantly tracking alterations in its probability distribution. Recognizing the efficacy of GMM in accurately representing data probability distributions, we leverage a batch of archived target data to initialize a GMM model based on the Expectation-Maximization algorithm. The underlying premise of GMM is its capacity to approximate real-world data through a finite number of mixture components, and it is formulated as follows:

$$P(x) = \sum_{k=1}^{K} P(x \mid C_i) \cdot w_k, \tag{4.21}$$

where K signifies the total number of Gaussians or mixture components which is set the same as the number of fuzzy rules, and x is the observed multivariate. Each mixture component C_k is associated with a weight w_k, which is determined based on the observations encompassing C_k, and $0 \leq w_k \leq 1, \sum_{k=1}^{K} w_k = 1$. $P(x \mid C_k)$ defines the likelihood of the observation x being allocated to the mixture component C_k. It can be represented by using the mean μ_k and the covariance Σ_k of each mixture component C_k as follows:

$$P(x \mid C_k) = \frac{1}{\left(2\pi^{d/2}\sqrt{|\Sigma_k|}\right)} \exp\left(-\frac{1}{2}\left(x-\mu_k\right)^T \Sigma_k^{-1}\left(x-\mu_k\right)\right). \tag{4.22}$$

Then the correlation between target samples can be assessed by the conditional probability $P(\mathbf{x}_{T_i}|C_k)$. If there is a temporal variation in the conditional probability of target samples, it may indicate the emergence of a new concept. However, detection based on a single sample is prone to sensitivity towards outliers. To mitigate this, we employ two sliding windows: the Reference Window $W_{ref} = [\mathbf{x}_{T_1}, \mathbf{x}_{T_2}, \ldots, \mathbf{x}_{T_n}]$ and the Detect Window $W_{det} = [\mathbf{x}_{T_{(n+1)}}, \mathbf{x}_{T_{(n+2)}}, \ldots, \mathbf{x}_{T_{(2n)}}]$, where n represents the number of instances in each window. The average conditional probability for the reference window is then determined using point estimation of the mean within the normal distribution by:

$$\mu_{\text{ref}} = \frac{1}{n} \sum_{k=1}^{n} \max_{k \in \{1,2,\ldots,K\}} P\left(\mathbf{x}_{T_i} \mid C_k\right). \tag{4.23}$$

The confidence interval estimation of the μ_{ref} is known to be $\left[\mu_{\text{ref}} - z_\alpha(\sigma/\sqrt{n}), \mu_{\text{ref}} + z_\alpha(\sigma/\sqrt{n})\right]$, where σ is the standard deviation and z_α is the significance level which is set as 3 Kim

and Park (2017). When the point estimation by the mean μ_{ref} in the detection window satisfies $\mu_{\text{det}} \geq \mu_{\text{ref}} + z_\alpha \times \sigma/\sqrt{n}$, the decision is made that the change has occurred. Otherwise, W_{ref} and W_{det} move step by step to receive new incoming data. If μ_{det} in the detection window meets or exceeds $\mu_{\text{ref}} + z_\alpha \times \sigma/\sqrt{n}$, it is inferred that a change has taken place. Otherwise, both W_{ref} and W_{det} are incrementally shifted to incorporate new incoming data.

4.2.5.2 Drift Adaptation

As discussed before, there are various drifting scenarios in multistream classification. Here, we systematically categorize them into three distinct scenarios for comprehensive analysis.

- No drift: when no drift is detected in either the source or target streams, the process advances with the incremental training of the base classifiers. During the initialization phase, the implementation of FSRL equips us with optimal antecedent parameters and shared consequent parameters $\mathbf{P}$. This allows for the newly arriving data $\mathbf{x}_{S_{vi}}$ to be effectively mapped into the fuzzy shared space. Subsequently, this mapped data $\mathbf{z}_{S_{v_i}}$ is employed to refine and update the base classifiers f_{S_v}.

- Source-only drifting: once a drift is detected within any source stream, an adaptation module should be deployed to handle new concepts. Similarly, we utilize the GMM to evaluate the distributions of the old and new concepts. For a newly incoming instance $\mathbf{x}_{S_{vi}}$, its importance weight $w_{S_{vi}}$ can be calculated by maximizing the conditional probability of GMM as follows:

$$w_{S_{vi}} = \max_{k \in \{1,2,\ldots,K\}} P\left(\mathbf{x}_{S_{vi}} \mid G_i\right). \tag{4.24}$$

When a new concept emerges in any source stream, adaptation is achieved by applying a multiplicative factor $w_{S_{vi}}$ to the data. This weighted data is then mapped into the fuzzy shared space. Subsequently, a new base classifier is created and trained using this mapped data $\mathbf{z}_{S_{v_i}}$. It is important to note that old base classifiers are not updated with new samples. Instead, they are preserved within a classifier pool, denoted as C_p, to maintain the old knowledge. Finally, the ensemble of joint predictive probabilities is formulated as follows:

$$f^E(\mathbf{z}_{T_i}) = \frac{w_{Si}}{\sum_{n=1}^{N} w_{Si} + \sum_{l=1}^{|C_p|} w_P} f_{S_v}(\mathbf{z}_{T_i}) + \frac{w_P}{\sum_{n=1}^{N} w_{Si} + \sum_{l=1}^{|C_p|} w_P} f_P(\mathbf{z}_{T_i}), \tag{4.25}$$

where w_P is the weight of l-th classifer in C_p, and $w_{Si} = \frac{1}{N}\sum_{i=1}^{N} w_{S_{vi}}$.

- Target-inclusive drifting: upon the detection of a target drift, the antecedent parameters and the shared consequent parameters $\mathbf{P}$, as well as the base classifiers f_{S_v} that were established in the initial phase, become inadequate for the classification of target samples. As a result, all base classifiers are removed from the classifier pool. This necessitates a re-initialization of the model, enabling it to adapt efficiently to the newly identified concepts.

The online learning process of IFSRL is detailed in Algorithm 4. During the online process, there are four main modules: FSRL, GMM, DDM, and the Hoeffding Tree classifier. The overall complexity of FSRL is given by $O(NK(3d+1)+\hat{d}m^2d^2+N\hat{d}d+CN^2)$. In this method, we employ the EM algorithm to estimate the GMM parameters, and its complexity can be regarded as linear, i.e., $O(n)$, where n is the window size. DDM also exhibits linear complexity, $O(n)$. The time complexity of each Hoeffding Tree classifier is $O(n\log(n))$. Assuming there are V source streams and 1 target stream, the time complexities of GMM, DDM, and Hoeffding Tree are $O((V+1)n)$, $O(Vn)$, and $O(Vn\log(n))$, respectively. Consequently, the overall complexity of IFSRL is $O(NK(3d+1)+\hat{d}m^2d^2+$

$N\hat{d}d + CN^2) + O((2V + 1 + V\log(n))n)$. In fact, since $N = (V + 1)n$, and V and K are quite small compared to n, the complexity of IFSRL depends on the window size n. Hence, we can adjust n to strike a balance between the performance of IFSRL and available resources.

Algorithm 4 The online learning process of IFSRL

Input: Input data $\{(\mathbf{x}_{S_{vi}}, y_{S_{vi}})\}_{i=1}^{N}$, $\{\mathbf{x}_{T_i}\}_{i=1}^{N}$; Window size n.
Output: Predicted target labels.

1: Read first n instances from each stream.
2: Obtain the Antecedent parameters of the TSK fuzzy systems and consequent transformation matrix P via Algorithm 3.
3: Create the source drift detectors DDM_{S_v} and GMM models for each source and target stream.
4: **while** there is incoming data **do**
5: **for** $v = 1 : V$ **do**
6: **if** Source drift is detected **then**
7: GMM-based adaptation by Eq.(4.24).
8: Construct the fuzzy shared representation $\mathbf{z}_{S_{v_i}}$ by Eq. (4.8a) and Eq.(4.9a).
9: Move the current classifier to the classifier pool C_p and retrain a new classifier f_{S_v} using $(\mathbf{z}_{S_{v_i}}, y_{S_{vi}})$.
10: **else**
11: Construct the fuzzy shared representation $\mathbf{z}_{S_{v_i}}$ by Eq. (4.8a) and Eq.(4.9a).
12: Incrementally training using $(\mathbf{z}_{S_{v_i}}, y_{S_{vi}})$.
13: **end if**
14: **end for**
15: Move detection window and calculate $\mu_{\text{det}}, \mu_{\text{ref}}$
16: **if** Target drift is detected **then**
17: Remove all base classifiers and return to line 1.
18: **else**
19: Construct fuzzy shared representation $\mathbf{z}_{T_i}$ by Eq. (4.8b) and Eq.(4.9b).
20: Predict the target label.
21: **end if**
22: **end while**

4.3 Experiments

In the experiment, we first demonstrated that BFSRL and IFSRL consistently outperform current methods in multistream classification, highlighting both robustness and

superiority. Second, because IFSRL performs a more comprehensive perspective, we validated the effectiveness of each proposed component when facing different challenges by ablation study. Finally, we established the scalability of IFSRL across diverse data streams, corroborating its stable predictive capabilities. In addition, we also analyzed parameter sensitivity.

Table 4.1: Characteristics of all datasets.

	Dataset	Type	Features	Class	Instance	Drift type
Sythetic	SEA	Single stream	3	2	100K	Abrupt/Recurring
	RBF	Single stream	10	2	20K	Incremental
	Tree	Single stream	20	2	20K	Abrupt/Gradual
	Hyperplane	Single stream	4	2	120K	Incremental
Real-world	Weather	Single stream	8	2	18K	Unknown
	Electricity	Single stream	8	2	45K	Unknown
	Kitti	Single stream	55	8	25K	Unknown
	CNNIBN	Multistream	124	2	120K	Unknown
	BBC	Multistream	124	2	120K	Unknown

4.3.1 Benchmarks

In the experiment, we use four synthetic: SEA Street and Kim (2001), RBF Song et al. (2021a), Tree Liu et al. (2020b), Hyperplane Bifet and Gavalda (2007), and four real-world datasets: Weather Ditzler and Polikar (2012), Electricity Liu et al. (2020b), Kitti Geiger et al. (2012), BBC Vyas et al. (2014) and CNNIBN Vyas et al. (2014) to simulate the multistream classification task and test our proposed method. The detailed characteristics of all datasets are elaborated in Table 4.1. And We adopt the same data sampling approach introduced in Section 3.3.1 to simulate the multistream classification scenario Yu et al. (2024a).

4.3.2 Baselines and Experiment Settings

To validate the effectiveness of our proposed approach, we conducted experiments comparing it with five state-of-the-art methods. Specifically, FUSION Haque et al. (2017) and ATL Pratama et al. (2019) are single-source stream-based algorithms, whereas Melanie Du et al. (2019), AOMSDA Renchunzi and Pratama (2022), and MCMO Jiao et al. (2022b) are designed for multi-source classification scenarios. For FUSION and ATL, we formed three distinct groups by pairing each source stream with the target stream: FUSIONs1, FUSIONs2, and FUSIONs3 and ATLs1, ATLs2, and ATLs3, respectively.

In this study, we implemented the framework using the scikit-multiflow learning library Montiel et al. (2018) in Python. All experimental evaluations were conducted on a server equipped with 187GB of memory and powered by an Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz.

4.3.3 Overall Performance

Our main experiments used a 3-source and 1-target setup, and Table 4.2 compares the classification accuracy of our proposed methods, i.e., BFSRL and IFSRL, against all baselines on four synthetic and five real-world datasets.

Table 4.2: Classification accuracy (%) with the variance of various methods on all benchmarks. The best results are highlighted in bold, while the second-best results are marked with an underline.

		Synthetic Datasets				Real-World Datasets				
Methods		**SEA**	**RBF**	**Tree**	**Hyperplane**	**Weather**	**Electricity**	**Kitti**	**CNNIBN**	**BBC**
Fusion	**S1**	$85.04_{\pm 0.84}$	$82.03_{\pm 1.41}$	$76.98_{\pm 1.11}$	$83.29_{\pm 0.67}$	$71.04_{\pm 1.50}$	$73.82_{\pm 2.56}$	$54.21_{\pm 2.61}$	$66.76_{\pm 0.74}$	$61.76_{\pm 0.09}$
	S2	$85.78_{\pm 0.92}$	$83.46_{\pm 1.20}$	$76.74_{\pm 1.00}$	$84.05_{\pm 0.52}$	$70.65_{\pm 1.32}$	$73.07_{\pm 3.01}$	$52.36_{\pm 2.72}$	$67.54_{\pm 1.11}$	$61.26_{\pm 0.43}$
	S3	$84.31_{\pm 1.13}$	$81.03_{\pm 1.73}$	$75.21_{\pm 1.07}$	$82.17_{\pm 0.57}$	$72.17_{\pm 1.17}$	$74.31_{\pm 2.73}$	$50.38_{\pm 2.43}$	$65.34_{\pm 0.92}$	$59.86_{\pm 0.19}$
ATL	**S1**	$88.42_{\pm 1.70}$	$84.53_{\pm 2.01}$	$76.43_{\pm 2.17}$	$86.17_{\pm 1.04}$	$74.57_{\pm 1.94}$	$75.07_{\pm 1.81}$	$52.78_{\pm 3.78}$	$62.78_{\pm 1.44}$	$62.78_{\pm 1.16}$
	S2	$88.74_{\pm 1.75}$	$85.21_{\pm 1.85}$	$76.71_{\pm 1.86}$	$87.07_{\pm 1.21}$	$75.03_{\pm 2.01}$	$75.83_{\pm 1.77}$	$54.01_{\pm 3.09}$	$65.74_{\pm 1.76}$	$62.34_{\pm 0.83}$
	S3	$87.62_{\pm 1.01}$	$83.16_{\pm 2.13}$	$76.07_{\pm 2.42}$	$86.01_{\pm 1.49}$	$74.62_{\pm 1.77}$	$73.96_{\pm 2.13}$	$53.26_{\pm 3.21}$	$62.65_{\pm 1.38}$	$60.76_{\pm 0.77}$
Melanie		$89.18_{\pm 0.77}$	$86.04_{\pm 0.39}$	$78.93_{\pm 0.61}$	$86.38_{\pm 0.57}$	$77.74_{\pm 0.89}$	$77.45_{\pm 2.95}$	$50.29_{\pm 1.34}$	$68.79_{\pm 0.31}$	$\mathbf{68.04}_{\pm 0.01}$
AOMSDA		$\mathbf{90.23}_{\pm 1.42}$	$85.26_{\pm 2.89}$	$76.87_{\pm 3.47}$	$87.66_{\pm 1.74}$	$76.55_{\pm 1.41}$	$78.02_{\pm 3.32}$	$\underline{67.79}_{\pm 3.16}$	$69.07_{\pm 1.40}$	$63.36_{\pm 1.07}$
MCMO		$87.46_{\pm 2.12}$	$86.26_{\pm 0.77}$	$77.64_{\pm 1.47}$	$84.04_{\pm 1.42}$	$76.02_{\pm 3.43}$	$78.79_{\pm 2.17}$	$64.82_{\pm 4.17}$	$68.83_{\pm 0.89}$	$60.12_{\pm 1.51}$
BFSRL		$87.62_{\pm 0.01}$	$\underline{86.32}_{\pm 0.42}$	$\underline{79.42}_{\pm 0.32}$	$\mathbf{87.78}_{\pm 0.14}$	$\underline{78.93}_{\pm 0.11}$	$\mathbf{80.27}_{\pm 0.58}$	$62.32_{\pm 1.21}$	$\underline{69.13}_{\pm 1.74}$	$60.15_{\pm 0.79}$
IFSRL		$\underline{89.53}_{\pm 2.01}$	$\mathbf{87.94}_{\pm 1.39}$	$\mathbf{79.89}_{\pm 1.06}$	$\underline{87.75}_{\pm 1.71}$	$\mathbf{80.04}_{\pm 2.12}$	$\underline{79.46}_{\pm 1.62}$	$\mathbf{69.06}_{\pm 1.14}$	$\mathbf{73.4}_{\pm 1.45}$	$\underline{63.48}_{\pm 2.07}$

First, in comparison to single-source-based techniques, such as Fusion and ATL, the multi-source-based methods consistently exhibit substantial advancements. This observation underscores the efficacy of leveraging multiple labeled source streams, as they contribute more discriminative and complementary information. Consequently, it emphasizes the significance of incorporating diverse data sources for achieving superior performance.

Secondly, compared to other multi-source-based methods, our approach consistently achieves top-tier performance on most datasets. Specifically, BFSRL outperforms the MCMO algorithm across all datasets, and even when compared to the supervised method (Melanie), it demonstrates superior results on six out of seven datasets. This directly attests to the effectiveness of our proposed FSRL algorithm. However, when contrasted with another blind adaptation method, AOMSDA, BFSRL exhibits weaker performance on the SEA and Kitti datasets. This discrepancy can be attributed to the fact that different methods respond differently to various types and frequencies of drift. Despite this, BFSRL outperforms AOMSDA significantly on the other five datasets. This highlights the robust performance of BFSRL across diverse datasets, showcasing its effectiveness in addressing knowledge transfer.

In addition, our proposed informed adaptation method, IFSRL, consistently exhibits superior performance on almost all datasets. This is attributed to the integration of drift detection, allowing for a more nuanced response to different types and frequencies of drift. Moreover, the method employs a GMM-based ensemble weighting prediction strategy, enabling the retention of prior knowledge for replay during predictions of new concepts. This approach proves to be more effective compared to previous methods, such as direct retraining or fine-tuning, in mitigating catastrophic forgetting. The incorporation of drift detection and ensemble weighting enhances the adaptability and knowledge preservation capabilities of IFSRL, contributing to its superiority across diverse datasets.

Table 4.3: Classification accuracy (%) of variants.

	$\mathbf{V}_1$	$\mathbf{V}_2$	$\mathbf{V}_3$	$\mathbf{V}_4$	**IFSRL**
SEA	85.76	87.78	87.04	88.42	**89.53**
RBF	85.39	85.91	85.03	86.92	**87.94**
Tree	76.57	77.49	77.91	78.72	**79.89**
Hyperplane	86.31	87.21	87.04	86.64	**87.75**
Weather	76.35	78.89	78.04	78.71	**80.04**
Electricity	75.87	76.72	77.21	78.33	**79.46**
Kitti	61.76	65.85	67.01	67.78	**69.06**
CNNIBN	69.47	71.71	70.42	71.84	**73.40**
BBC	60.41	62.05	62.57	63.04	**63.48**

4.3.4 Ablation Study

To validate the rationality of each component and its impact on the overall classification results, we designed four variants of IFSRL. As shown in Table 4.3, the baseline (V_1) directly removes the FSRL component, providing a reference point for comparison. Subsequent variants explore the effects of individual modifications. V_2 disregards the shared consequent parameters and independently considers each stream. V_3 omits the use of GMM to address asynchronous concept drift in source streams. V_4 eliminates the classifier pool, i.e., it employs only one specific classifier for each stream without considering prior knowledge.

The classification accuracy results across multiple datasets provide several insights. Concretely, IFSRL consistently outperforms all variants across all datasets, demonstrating the effectiveness of the integrated components, especially the significance of fuzzy shared representation during the knowledge transfer. Specifically, V_2 and V_3 exhibit competitive accuracy compared to the baseline V_1, suggesting that shared consequent parameters and GMM-based handling of asynchronous concept drift individually con-

tribute to improved performance. V_4 demonstrates the importance of the classifier pool, showcasing that considering multiple classifiers for each stream and preserving prior knowledge through the ensemble approach in IFSRL significantly enhances adaptability and performance.

These findings underscore the synergistic impact of the proposed modifications in IFSRL, affirming its robustness in addressing various challenges in the multistream classification. The superior performance across all ablation variants reaffirms the efficacy of our proposed method in handling complex dynamic correlations.

4.3.5 Influence of Source Numbers

In this section, we explore the influence of the number of source streams on the performance of IFSRL. Specifically, we assess IFSRL's performance using 1, 2, 3, and 5 source streams, respectively. For configurations with more than 3 sources, we evenly distribute the source samples into V streams and re-sample them to ensure an equivalent number of samples as in the target stream. Our primary objective is to determine if leveraging multiple source streams enhances predictive capability compared to a single-source stream. As depicted in Figure 4.3, the results indicate a consistent improvement in performance with multi-source streams across all datasets, surpassing the performance of a single-stream setup. This observation underscores the ability of multi-source streams to provide supplementary and complementary information, thereby enhancing overall predictive performance.

However, it is noteworthy that as the number of source streams increases, there might be a decline in performance. For instance, on the Weather and Kitti datasets, the performance with 3 source streams surpasses that with 5 source streams. This trend could be attributed to the increased complexity of the model as the number of source streams grows, potentially affecting its overall performance. Despite these fluctuations,

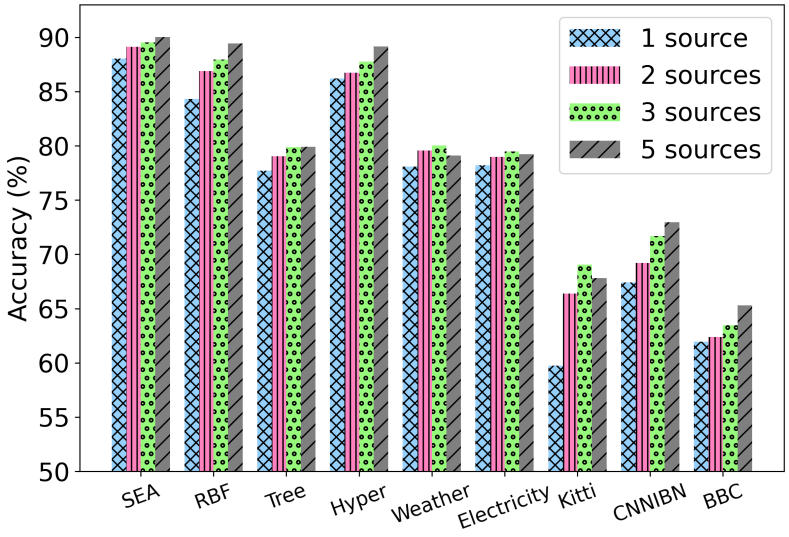


Figure 4.3: The influence of the different number of sources.

the performance of IFSRL remains stable across various source configurations. This stability suggests that our proposed method can effectively adapt to different numbers of data streams, showcasing its versatility and robustness in handling diverse source stream scenarios.

4.3.6 Parameters Analysis

In the proposed IFSRL method, three key parameters significantly influence the classification results: the number of fuzzy rules K, the window size n, and the dimensions after feature mapping. To analyze their respective impacts on the prediction performance, experiments were conducted on all datasets using different parameter values. Specifically, we set the fuzzy rules number to be {2, 3, 5, 7}, the window size to be {100, 200, 300, 500}, and the dimensions after reduction to be {100%, 70%, 50%, 30%, 10%} of the original dimensions, respectively. During the experiments, each parameter was tuned independently while keeping the others fixed, and the diverse performances are depicted in Figure 4.4.

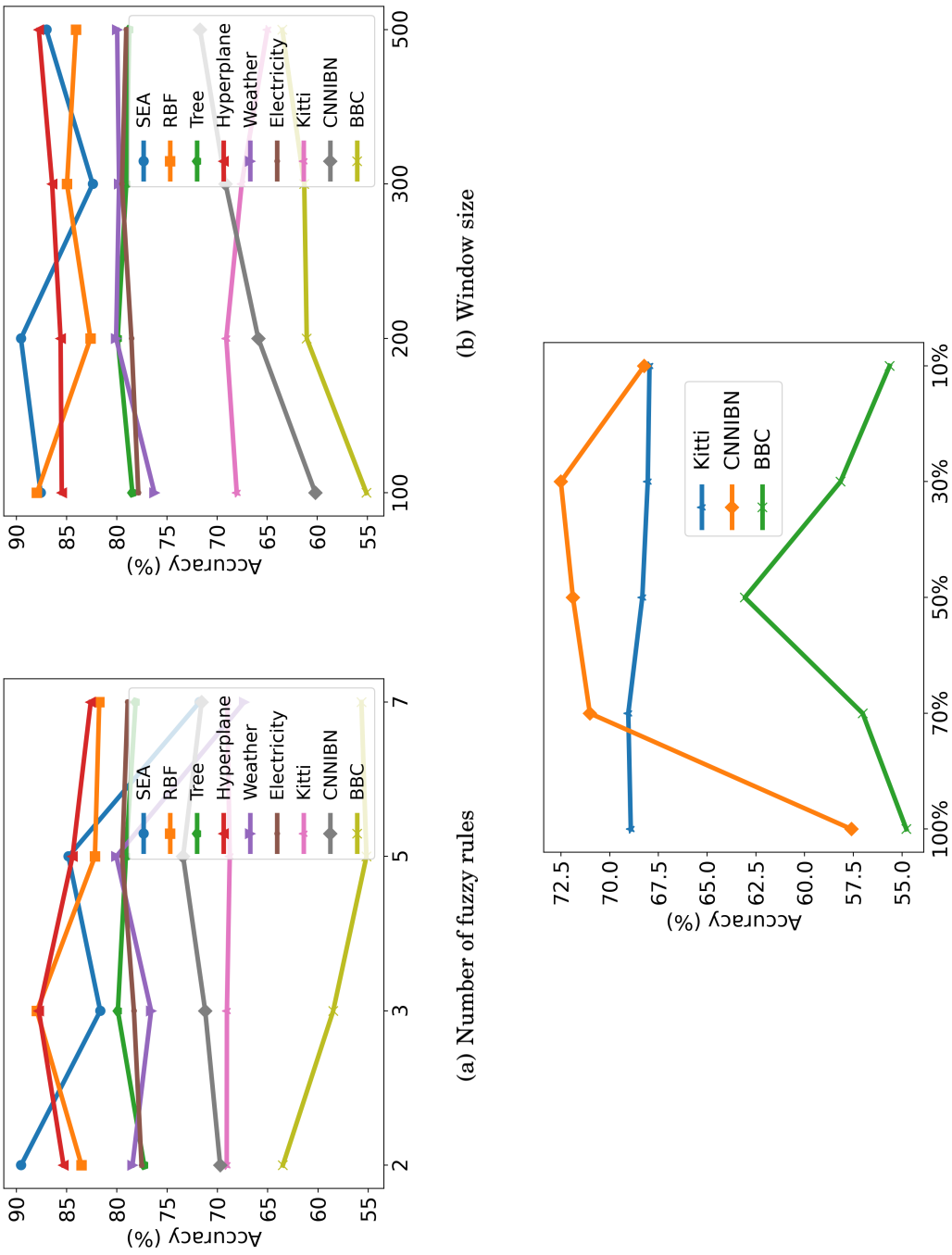


Figure 4.4: The effect of main parameters on classification accuracy.

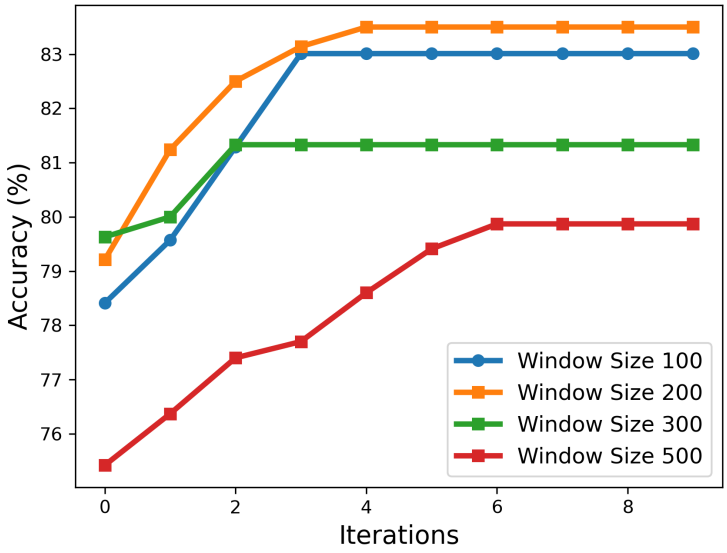
Different datasets exhibit varying optimal fuzzy rules primarily due to the diverse fuzzy relationships among multiple data streams. Additionally, each dataset possesses unique drift frequencies and periods, and the choice of window size significantly impacts the results. For datasets with higher frequency drifts, a more flexible window size may be necessary to adapt to changes in data distribution. Finally, the impact of data dimensionality on results is paramount, primarily due to the substantial influence of redundant features. This is why our approach considers drift adaptation in a low-dimensional fuzzy shared space, effectively mitigating the influence of redundant features and thereby enhancing the algorithm's robustness. The optimal parameters configured in our experiments are also listed in Table 4.4.

4.3.7 Convergence Analysis

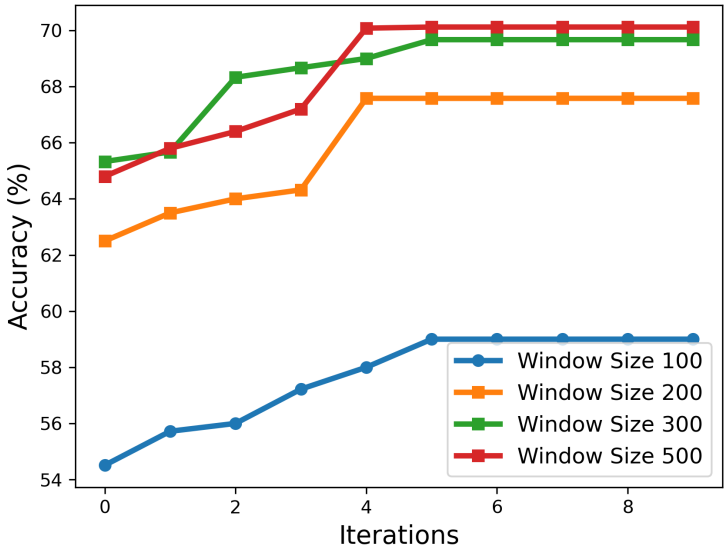
To further validate the convergence of this algorithm, we conducted experiments on both a synthetic (SEA) and a real-world (Weather) dataset. We tested the trend of accuracy with increasing iterations under different window sizes (i.e., window size $\in \{100, 200, 300, 500\}$). As depicted in Figure 4.5, we observed a increase in accuracy with the number of iterations, stabilizing within approximately five iterations for both datasets. These experimental results demonstrate the guaranteed convergence of our method.

4.4 Summary

This chapter introduces Fuzzy Shared Representation Learning (FSRL) as an innovative method for multistream classification, specifically tailored to address challenges associated with non-stationary data streams that exhibit concept drift. FSRL integrates a TSK fuzzy system for non-linear transformation and introduces a joint distribution adaptation for inter-stream shift alignment and feature reduction. It enhances the model's



(a) SEA dataset



(b) Weather Dataset

Figure 4.5: Convergence analysis on SEA and Weather datasets.

Table 4.4: Parameter settings on different datasets.

	Fuzzy rules	**Window size**	**Feature dimension**
SEA	2	200	3
RBF	3	100	10
Tree	3	200	20
Hyperplane	3	500	4
Weather	5	200	8
Electricity	5	300	8
Kitti	2	200	30
CNNIBN	5	400	50
BBC	2	500	50

interpretability, adaptability, and promotes robust knowledge transfer across multiple streams. Moreover, our proposed method features two pioneering adaptation strategies: 1) BFSRL empirically demonstrates the effectiveness of FSRL; 2) IFSRL incorporates a drift detection module and GMM-based distribution adaptation to handle asynchronous drifts. These contributions present practical solutions for managing the dynamic nature of real-world data streams, underscoring the significance of FSRL in fortifying the robustness and interpretability of multistream classification.

CHAPTER 5

LEARN-TO-ADAPT FOR MULTISTREAM CLASSIFICATION

Current drift adaptation methods often rely on hand-crafted features, which are typically low-dimensional and may not be suitable for handling complex data (RQ3). To solve this problem and achieve RO3, this chapter proposes a meta-learning approach and introduces a learn-to-adapt (L2A) framework. L2A adapts the feature extractor and classifier in a feedback process, which is advanced in dealing with more complex and high-dimensional data streams. In this chapter, the detailed analysis of motivations and research backgrounds is introduced in Section 5.1. Preliminaries, settings, and proposed L2A are discussed in Section 5.2. Section 5.3 conducts the experiment and Section 5.4 concludes this chapter.

5.1 Introduction

Social media and autonomous systems generate enormous volumes of data continually over time, i.e., data streams, such as the fields of automatic driving Liu et al. (2020c), wearable devices, weather forecasts, and e-commerce Gomes et al. (2017a). Developing

efficient data analytics and learning techniques for data streams to facilitate precise decision-making in real-world applications is in high demand McKay et al. (2019). One of the most challenging problems in the prediction of data streams is concept drift, where the data distribution irregularly changes over time Lu et al. (2018a). When concept drift occurs, the prediction ability of an offline-trained model will be destroyed Song et al. (2021a) since a well-trained model with previous data cannot obtain accurate prediction results when it is applied to newly arriving data Haque et al. (2018); Wang et al. (2022c).

Concept drift adaptation (CDA) methods aim to tackle the concept drift problem in data streams by continuously retraining outdated models Halstead et al. (2021), namely by discarding previous classifiers and training a new classifier with the newly arriving data when concept drift occurs. In this way, the classifier can always fit the newest data distribution. CDA methods have been extensively studied and developed in recent years by introducing many state-of-the-art ideas, such as automated machine learning (AutoML) based methods Celik and Vanschoren (2021); Yan et al. (2021a). These methods aim to design automatic operations to search for the optimal pipelines and adjust the pipelines continually when the data distribution changes over time Pratama et al. (2019). Online incremental learning-based methods are proposed to accumulate knowledge for the evolving adaptation problem or data streaming regression Bitarafan et al. (2016); Yu et al. (2020a). In addition, lifelong learning is applied to develop models that can expand their knowledge to make it suitable for new concepts Lao et al. (2020). For instance, the streaming decision tree Korycki and Krawczyk (2021) deals with the concept drift problem by using a proposed streaming class-conditional attribute estimation to avoid knowledge forgetting. These existing studies have shown that CDA methods can effectively deal with data streams with changing distributions Dong et al. (2021); Song et al. (2021c). So far, most existing CDA methods are designed for a single-labeled stream, where the label arrives after obtaining the features for each

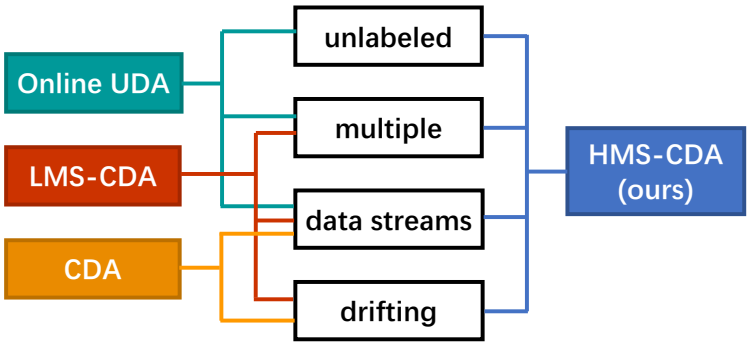


Figure 5.1: Comparison of different settings related to our HMS-CDA. Online UDA: online unsupervised domain adaptation; CDA: concept drift adaptation; LMS-CDA: labeled multi-stream concept drift adaptation; HMS-CDA (ours): hybrid multi-stream concept drift adaptation.

data instance.

However, real-world applications often involve hybrid multiple streams where massive labeled and unlabeled streams arrive simultaneously Chandra et al. (2016). For example, social media platforms like TikTok always handle *multiple data streams* (e.g., images or videos) from different users Yue et al. (2019); Ji et al. (2016), with each user generating one data stream. Real-time sentiment prediction in social media requires users' true sentiment to train the model. However, some users are not willing to provide the ground truth. Therefore, the prediction method needs to handle labeled and unlabeled streams at the same time. When using existing single-stream aimed CDA methods to deal with this kind of multi-stream tasks, they cannot predict the unlabeled streams because they can only handle each stream with delayed labels separately Song et al. (2020).

We propose a more challenging and general setting to handle such hybrid multiple streams, named concept drift adaptation for hybrid multiple streams (HMS-CDA) which contains unlabeled drifting streams. As shown in Figure 5.1, online unsupervised domain adaptation (UDA) refers to the continual adaption from labeled source domains (without drifts) to the target domain Hoffman et al. (2014); Bitarafan et al. (2016). Existing

concept drift adaptation (CDA) mainly focuses on a single drifting stream with delayed labels Song et al. (2021a). Labeled multiple stream adaptation (LMS-CDA) deals with labeled and drifting data streams Zhou et al. (2021). Compared with these existing settings, HMS-CDA places fewer limitations on data streams, where streams can be stationary (non-drifting) or non-stationary (drifting), and streams can be labeled or unlabeled. Since HMS-CDA does not require fully labeled streams, this can effectively save the cost of annotation and protect the privacy of users' information.

In this proposed setting, we assume that one drifting stream is always labeled (denoted as the labeled source stream) and other drifting streams are always unlabeled (denoted as unlabeled target streams). As shown in Figure 5.2, the distribution of the labeled source stream (stream A) is biased towards any target stream (stream I or stream M), and concept drifts may occur synchronously or asynchronously between any two streams. Details of the proposed setting will be explained with our proposed framework in Section 5.4. Compared with the existing CDA studies, there are three main problems that need to be solved in HMS-CDA: a) unlabeled adaptation and prediction, b) distribution bias between streams, c) Asynchronous drift adaptation. In other words, the goal of HMS-CDA is how to predict the unlabeled and drifting streams, or how to efficiently use the source stream with the label information to predict target streams and address the concurrent and non-concurrent drifts between two streams at the same time. A possible solution is to use unsupervised domain adaptation (UDA) methods to classify unlabeled data Zhong et al. (2021). However, UDA assumes a fixed relationship between the labeled data and unlabeled data. The relationship means there is only domain bias between the source and target domains while the distribution of each domain is fixed instead of evolving over time. Therefore, the UDA method is not designed to react to concept drift and track the evolving data, leading to prediction failure when applied in drifting streams Yang et al. (2021).

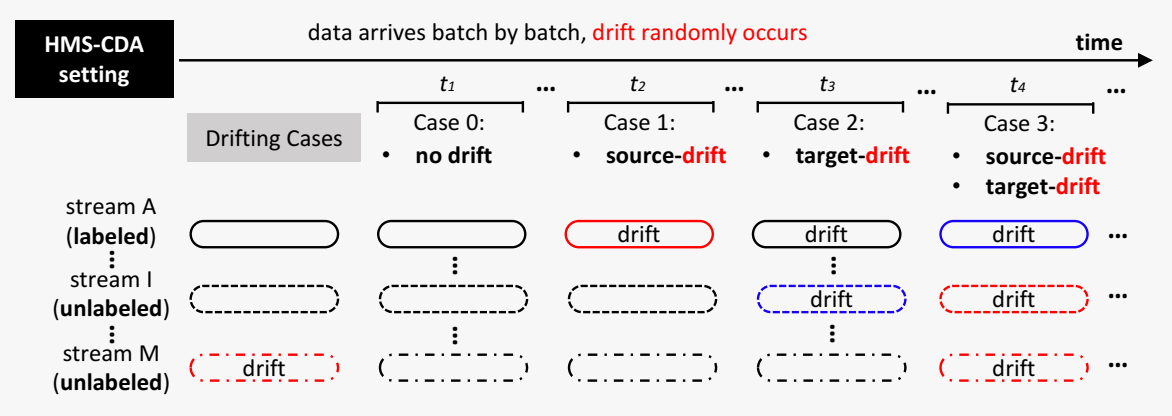


Figure 5.2: Schematic of HMS-CDA setting. In this figure, the color change denotes the drift occurring in this batch, where its distribution is different from the previous batch. In HMS-CDA there are four different cases: no drift, source drift, target drift and concurrent drift.

Inspired by Finn et al. (2017), the learning-to-learn (meta-learning) methods have made considerable advances in the few-shot learning domains, demonstrating a promising performance for fast adaptation. Thus, we cast the *HMS-CDA* problem into a **L**earn-to-**A**dapt framework (L2A). Specifically, the proposed framework comprises two stages as shown in Figure 5.3: (1) the meta-training stage aims at building the ability of invariant representations for adapting different streams and accumulating the existing knowledge to accelerate future learning; (2) the online adaptation stage updates the old model by exploiting and adapting the knowledge of downstream concepts, preventing old knowledge from being overwritten. Compared with the existing CDA methods, L2A not only adapts the classifiers, but also integrates the feature extractor and classifiers into a feedback process.

5.2 Proposed Method

In this section, we formalize the *HMS-CDA* setting and present the proposed **L**earn-to-**A**dapt (L2A) framework in detail. Our study assumes that the source stream and

multiple target streams have some internal related representations and share the same label space. Therefore, the intuition behind this approach is to predict the target labels by transferring the knowledge from the labeled source data to the target streams independently. The source labels can be manually or automatically annotated in practical applications. For a better understanding of knowledge transfer from the source to any target stream, we introduce the proposed approach in the two-stream scenario in this section. Also, it can be easily extended to a multi-stream scenario due to the independent transfer process from the source stream to each target stream.

In addition, we focus only on the CDA problem and hence consider the streaming process as a sequence of stationary concepts. More concretely, it is assumed that the data come continually in batches, each of which is termed as one concept with the same distribution. On the contrary, the distribution between each concept is indeterminate. The biggest challenge in this task is to determine how to efficiently use the source stream with label information to predict the target stream while addressing the concurrent and non-concurrent drifts between two streams at the same time.

5.2.1 Preliminaries and the *HMS-CDA* Setting

Given the samples X and the corresponding labels Y, $S_i = (X_i^s, Y_i^s)$ denotes a batch of labeled source data arriving at the i-th time period (shorten as at i) with its probability density function (pdf) $P_{S_i}(X_i^s, Y_i^s)$. Similarly, $T_i = (X_i^t)$ denotes a batch of unlabeled target data arriving at i with a marginal pdf of $P_{T_i}(X_i^t)$, where $P_{S_i}(X_i^s) \neq P_{T_i}(X_i^t)$. Specifically, we assume X_i^s arrives at the beginning of the i-th period, and Y_i^s is obtained at the end of this period. Therefore, the source stream is denoted as $S = \{S_i | i = 1, 2, \ldots\}$ and the target stream as $T = \{T_i | i = 1, 2, \ldots\}$.

Our task is to predict the labels for the source and target streams. At each i, L2A first estimates Y_i^s by outputting $\hat{Y}_i^s$, and we can obtain the accuracy of S_i. Then L2A estimates

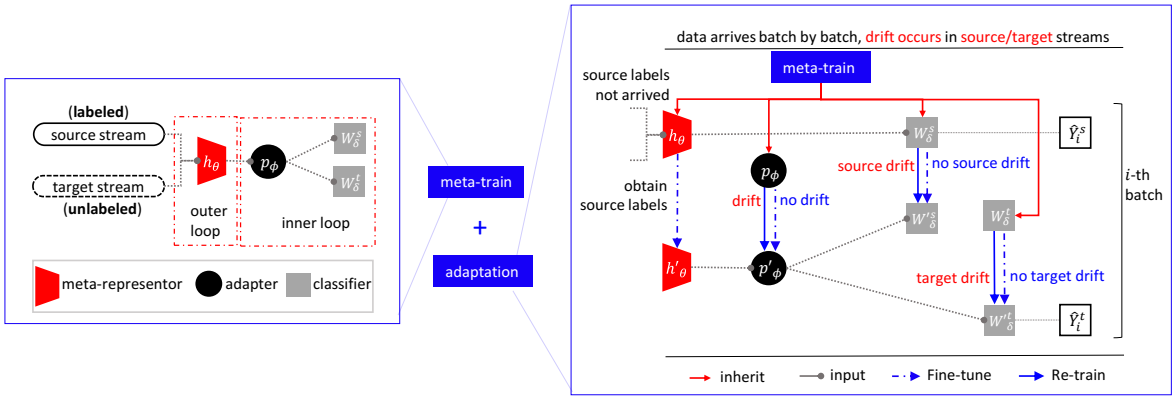


Figure 5.3: The L2A framework. 1) Meta-training stage: A meta-representor and an adapter are designed to learn the invariant representations between streams. The learned invariant representations are fed into classifiers. The adapter and classifier are optimized in the inner loop while the meta-representor is optimized in the outer loop. (The meta-training stage incorporates the MAML training strategy.) The meta-training stage offers an initialized model built with historical source and target data. 2) Online adaptation stage: We assume the samples arrive at the beginning of the period, and the labels of the source stream arrive at the end of the period. In this stage, we first predict the source stream and obtain the prediction error for the source stream. After obtaining the true labels, the meta-representor, adapter and classifiers will be updated and then used to predict the target stream.

the label of T_i by outputting $\hat{y}_i^t$, with known Y_i^s. It should be noted in the online setting that S_i and T_i are *only* available at i and will *not be stored* for future use.

One challenge in the *HMS-CDA* setting is that both S and T have the concept drift problem, summarized into the following four cases below (also shown in Fig. 5.2):

- Case 0 (*No Drifts*): $\exists i$ if $P_{S_i} = P_{S_{i-1}}$ and $P_{T_i} = P_{T_{i-1}}$, denoting no drift occurs in any streams.

- Case 1 (*Source Drift*): $\exists t$ if $P_{S_i} \neq P_{S_{i-1}}$ but $P_{T_i} = P_{T_{i-1}}$, the drift only occurs in the *source* stream.

- Case 2 (*Target Drift*): $\exists t$ if $P_{S_i} = P_{S_{i-1}}$ but $P_{T_i} \neq P_{T_{i-1}}$, the drift only occurs in the *target* stream.

- Case 3 (*Concurrent Drifts*): $P_{S_i} \neq P_{S_{i-1}}$ and $P_{T_i} \neq P_{T_{i-1}}$, *concurrent drift* occurs meaning both source and target streams have drift.

Therefore, our goal is to effectively correlate the distribution bias between different streams, whilst continuously addressing the arbitrary concept drift in each stream.

5.2.2 Learn-to-Adapt

L2A has two stages (As shown in Figure 5.3): 1) a meta-training stage and 2) an online adaptation stage. We will explain these two stages separately.

5.2.2.1 Stage 1: Meta-training

In the meta-training stage, a meta-representor (h_θ) with an adapter (p_ϕ) is designed to learn the invariant representations (i.e., a feature extractor) between drifting streams, and the learned invariant representations will be fed into classifiers (W_δ^s, and W_δ^t).

The meta-representor is inspired by the meta-leaning idea presented in model-agnostic meta-learning (MAML) Finn et al. (2017), which can quickly adapt from existing tasks to a new task. More specifically, MAML aims to learn the meta parameters that are sensitive to changes, which means minimal changes to the parameters can cause a large improvement toward to the direction of the gradient of the new loss. The inspiration behind the design is that some internal representations are broadly applicable to all tasks and therefore they can be expanded to our hybrid setting. In the MAML protocol, a meta-training module separates the data samples of the existing tasks into support sets and query sets. The parameters in the meta-training module are optimized by minimizing errors on the support set in the inner loop and minimizing errors on the query set in the outer loop.

To fit our problem, we regard each batch of data as one task, namely there are two tasks, $S_i = (X_i^s, Y_i^s)$ and $T_i = (X_i^t)$ at each i. Before L2A reaches to the online adaptation

stage, we collect n batches of historical source and target data, $S(n) = \{S_i | i = 1, \ldots, n\}$ and $T(n) = \{T_i | i = 1, \ldots, n\}$, to pre-train the model in the meta-training stage. Hence, the overall meta-objective function can be formulated as,

$$\min_{\theta,\phi,\delta} = \sum_{i=1\ldots n} \left[L(f_{\theta,\phi,\delta}(X_i^s), Y_i^s) + d(S_i, T_i) \right], \tag{5.1}$$

where $f_{\theta,\phi,\delta} = h_\theta \circ p_\phi \circ W_\delta$ represents the composite function of the model and L is the cross-entropy loss. The domain discrepancy d is implemented as the Maximum Mean Discrepancy (MMD) following the approach by Long et al. (2017). Concretely, given the source domain data S_i containing N_{S_i} samples and the target domain data T_i containing N_{T_i} samples,

$$\begin{aligned} d(S_i, T_i^{spt}) = & \frac{1}{N_{S_i}^2} \sum_{u,v} k\left(f_\theta(X_u^s), f_\theta(X_v^s)\right) \\ & + \frac{1}{N_{T_i^{spt}}^2} \sum_{u,v} k\left(f_\theta(X_u^{t:spt}), f_\theta(X_v^{t:spt})\right) \\ & - \frac{2}{N_{S_i} N_{T_i^{spt}}} \sum_{u,v} k\left(f_\theta(X_u^s), f_\theta(X_v^{t:spt})\right), \end{aligned} \tag{5.2}$$

where $X_u^{t:spt}$ is one data sample in T_i^{spt}, and $k(\cdot,\cdot)$ is the multivariate Gaussian kernel.

More concretely, we separate $T(n)$ into the *support sets* $T^{spt}(n) = \left\{ T_i^{spt} | i = 1, \ldots, n \right\}$ and *query sets* $T^{qry}(n) = \left\{ T_i^{qry} | i = 1, \ldots, n \right\}$. As $S(n)$ is involved during the optimization of all the parameters (θ, ϕ, and δ), we do not separate $S(n)$, regarding $S(n)$ as support sets in the inner loop as well as query sets in the outer loop.

Given the historical source data $S(n)$, support sets of the historical target data $T^{spt}(n)$, and query sets of the historical target data $T^{qry}(n)$, the adapter and classifiers are trained in the inner loop with SGD Denevi et al. (2019):

$$(\phi, \delta)_{i+1} \leftarrow (\phi, \delta)_i - \eta_{in} \nabla_{(\phi,\delta)} \left[L(f_{\theta,\phi_i,\delta_i}(X_i^s), Y_i^s) + d(F_i, F_i^{spt}) \right], \tag{5.3}$$

where η_{in} is the inner loop learning rate.

In the outer loop, the meta-representor h_θ is optimized by:

$$\theta \leftarrow \theta - \eta_{\text{out}} \nabla_\theta \left[L(f_{\theta,\phi,\delta}(X^s), Y^s) + \frac{1}{n} \sum_{i=1}^{n} d(S_i, T_i^{qry}) \right], \tag{5.4}$$

where η_{out} is the outer loop learning rate and n is the number of concepts within one batch.

Algorithm 5 The learning algorithm of L2A

Stage 1: Meta Training

Input: n historical source and target data batches: $S(n) = \{S_i | i = 1, \ldots, n\}$ and $T(n) = \{T_i | i = 1, \ldots, n\}$

Output: Pre-trained meta-representor h_θ, adapter p_ϕ and classifier W_δ.

for t = 0 : MaxIter **do**
  Sample source stream $S = \{X_i^s, Y_i^s\}$,
  Sample target stream T_i^{spt} and T_i^{qry}.
  Initialize θ, ϕ and δ
  for i=0:innerIter **do**
    Update the adapter p_ϕ and classifier W_δ according to Eq.(5.3).
  end for
  Update the meta-representor h_θ according to Eq.(5.4)
end for

Stage 2: Online Adaptation

Input: New coming source and target data batches: $S = \{(X_1^s, Y_1^s), ..., (X_i^s, Y_i^s)\}$ and $T = \{(X_1^t,), (X_2^t,), ..., (X_i^t,)\}$; Pre-trained meta-representor h_θ, adapter p_ϕ and classifier W_δ.

Output: Adapted meta-representor h_θ, adapter p_ϕ and classifier W_δ.

Initialize source classifier W_{δ^s}, target classifier W_{δ^t}.
while online learning **do**
  if Case 0 occurs **then**
    W_δ and h_θ are updated by Eq.(5.5) and Eq.(5.6)
  else if Case 1 occurs **then**
    h_θ, p_ϕ and W_{δ^t} are updated by Eq.(5.6) and Eq.(5.7)
  else if Case 2 occurs **then**
    h_θ, p_ϕ and W_{δ^t} are updated by Eq.(5.6) and Eq.(5.8)
  else if Case 3 occurs **then**
    h_θ, p_ϕ, W_{δ^s} and W_{δ^t} are updated by Eq.(5.6), Eq.(5.7) and Eq.(5.8)
  end if
end while

5.2.2.2 Stage 2: Online Adaptation

After the meta-training, we have an initialized model with the ability of invariant representation and adaptation. At the adaptation stage, we design an online update strategy to update the model by exploiting and adapting the knowledge of new data so that the model can always handle newly arriving data. Although we use W_δ^s and W_δ^t for meta-training, they are actually identical since there is no drift at this stage. W_δ^s and W_δ^p will be adapted into different classifiers at the adaptation stage to make more accurate predictions in the drifting environment.

At each time period, we need to predict T_i as well as using T_i to update the meta-representor. Therefore, T_i will be regarded as a support set in the inner loop for prediction and also a query set in the outer loop for updates, i.e., T_i^{spt}=T_i and T_i^{sqy}=T_i. More concretely, when a new concept pair arrives, we regard the newly arrived S_i and T_i as two support sets to update the adapter and stream-specific classifiers in the inner loop. The same S_i and T_i are also used as the query sets to update the meta-representor. Next, we explain the details of the adaptation strategies case by case.

(1) **Case 0** (*No Drifts*): In this situation, there are no distribution changes either in the source or target streams. Therefore, we only need to accumulate the newly arriving knowledge by fine-tuning the meta-representor, adapter and classifiers as follows:

$$\delta_{i+1} \leftarrow \delta_i - \eta_{cin} \nabla_\delta \left(f_{\theta,\phi_i,\delta_i} \left(X_i^s \right), Y_i^s \right), \tag{5.5}$$

$$\theta \leftarrow \theta - \eta_{\text{cout}} \nabla_\theta \left[L \left(f_{\theta,\phi_i,\delta_i} \left(X_i^s \right), Y_i^s \right) + d(S_i, T_i) \right], \tag{5.6}$$

where η_{cin} is the inner learning rate at the online stage.

(2) **Case 1** (*Source Drift*): Since a new pattern occurs in the source stream, the source classifier and adapter become outdated and the ϕ, δ^s should be *initialized* and then updated by,

$$(\phi, \delta^s)_{i+1} \leftarrow (\phi, \delta^s)_i - \eta_{cin} \nabla_{(\phi,\delta^s)} \left[L \left(f_{\theta,\phi_i,\delta_i^s} \left(X_i^s \right), Y_i^s \right) + d(S_i, T_i) \right] \tag{5.7}$$

In addition, to keep the ability of invariant representation, the meta-representor should be updated using Eq.(5.6) simultaneously.

(3) **Case 2** (*Target Drift*): A new concept occurs only in the target stream, so the target classifier and the adapter should be *initialized* first and updated by,

$$(\phi,\delta^t)_{i+1} \leftarrow (\phi,\delta^t)_i - \eta_{cin}\nabla_{(\phi,\delta^t)}\left[L\left(f_{\theta,\phi_i,\delta_i^t}\left(X_i^s\right),Y_i^s\right) \quad + d\left(S_i,T_i\right)\right] \tag{5.8}$$

Similarly, the meta-representor should keep accumulating the newly arriving knowledge from the labeled source streams and enhance the ability of inter-stream adaptation. Therefore, it should be further fine-tuned using Eq.(5.6).

(4) **Case 3** (*Concurrent Drift*): New concepts occur in both streams, so all the components (i.e., feature extractor, classifiers, and adapter) in the model should be updated to adapt to the new concepts via Eq.(5.6), Eq.(5.7), and Eq.(5.8). All the processes of *L2A* algorithm are summarized in Algorithm 5.

5.3 Experiment

To evaluate our proposed method (L2A), we conduct comprehensive experiments to show the performance of L2A when dealing with the HMS-CDA problem. Firstly, we introduce the benchmarks and give a description of how to simulate the proposed setting in Section 5.3.1. Then, the detailed implementation of the framework is introduced in Section 5.3.2 while the baseline and ablation design are described in Section 5.3.3. In Section 5.3.4, we conduct the average classification performance of all benchmarks and discuss the strengths and weaknesses of the proposed method. In order to show the online adaptation capacity, we also show the online classification accuracy in Section 5.3.5. In addition, we further analyze and discuss the influence of some experimental settings on the adaptation performance, such as convergence, online time, sensitivity to the batch size, and the effect of adaptation steps.

Table 5.1: The detailed implementation of the feature-extractor.

Convolution layer	**Activation Function**	**Batch Normalization**
conv1, [64, 1, ,3, 3, 2, 0]	ReLU	✓
conv2, [64, 64, ,3, 3, 2, 0]	ReLU	✓
conv3, [64, 64, ,3, 3, 2, 0]	ReLU	✓
conv4, [64, 64, ,3, 3, 2, 0]	ReLU	✓

Table 5.2: Classification accuracy on Rotated MNIST dataset.

Method	**Source Stream**					**Target Stream**				
	0°	25°	50°	75°	Avg.	90°	115°	140°	165°	Avg.
EAML	–	–	–	–	–	0.16	0.17	0.23	**0.33**	0.22
L2A-α	0.78	0.77	0.74	0.67	0.76	0.20	0.16	0.14	0.13	0.16
L2A-β	**0.94**	**0.90**	**0.83**	**0.78**	**0.86**	0.19	0.14	0.18	0.24	0.19
L2A-γ	0.84	0.78	0.69	0.63	0.74	0.15	0.13	0.13	0.15	0.14
L2A	0.93	0.89	0.82	0.75	0.85	**0.25**	**0.24**	**0.30**	0.31	**0.28**

5.3.1 Benchmarks & Setting Simulation

In the experiment, we simulate three HMS scenarios using two public datasets (i.e., MNIST and FashionMNIST), named **Rotated MNIST**, **Rotated FashionMNIST** and **MNIST / FashionMNIST**, to simulate three HMS scenarios via the rotation operation.

Firstly, we process these three benchmarks as the study case of two streams (1-source, 1-target). Specifically, for **Rotated MNIST** and **Rotated FashionMNIST**, the training images with a rotation of 0° are regarded as the source stream while the images with a rotation of 90° are the target stream at the meta-train stage. In the online adaptation stage, we construct the source and target streams by rotating the testing images from different angles. Specifically, the source stream $S = \{X_1^s, X_2^s, ..., X_i^s\}, i \in [0°, 25°, 50°, 75°]$

Table 5.3: Classification accuracy on Rotated FashionMNIST dataset.

Method	**Source Stream**					**Target Stream**				
	0°	25°	50°	75°	Avg.	90°	115°	140°	165°	Avg.
EAML	–	–	–	–	–	0.05	0.07	0.12	**0.20**	0.11
8**L2A-**α	0.67	0.54	0.54	0.48	0.55	0.16	0.11	0.12	0.16	0.13
L2A-β	0.78	0.66	0.62	0.54	0.65	0.13	0.06	0.11	0.17	0.11
L2A-γ	0.72	0.64	0.60	0.53	0.62	0.16	0.11	0.10	0.15	0.13
L2A	**0.79**	**0.68**	**0.63**	**0.54**	**0.66**	**0.23**	**0.12**	**0.13**	0.18	**0.16**

and the target stream $T = \{X_1^t, X_2^t, ..., X_j^t\}, j \in [90°, 115°, 140°, 165°]$ are sampled randomly to formulate the *HMS-CDA* problem.

For **MNIST / FashionMNIST:**. In the meta-training stage, we regard the training MNIST images with a rotation of 0° as the source stream while the FashionMNIST with a rotation of 0° are the target stream. In the online stage, the rotated MNIST and the FashionMNIST with $\{0°, 25°, 50°, 75°\}$ are randomly sampled as the source stream and target stream.

To further demonstrate its ability to deal with multiple streams, we also extend the setting into three-stream case (1-source, 2-target). In detail, we add one more target stream $T = \{X_1^t, X_2^t, ..., X_j^t\}, j \in [90°, 180°, 205°, 230°]$ for **Rotated MNIST** and **Rotated FashionMNIST**.

5.3.2 Implementation

For all benchmarks, each task within a distribution-consistent batch during the meta-training is sampled as a 10-way, 10-shot classification problem. We sample 100 tasks in the meta-training stage and construct the drifting streams with 1000 batches in the online adaptation stage. For all experiments, the feature extractor is constructed by

Table 5.4: Classification accuracy on MNIST / FashionMNIST dataset.

Method	Source Stream					Target Stream				
	0°	25°	50°	75°	Avg.	0°	25°	50°	75°	Avg.
EAML	–	–	–	–	–	0.11	0.12	0.12	0.09	0.11
L2A-α	0.74	0.70	0.62	0.54	0.65	0.11	0.12	0.11	0.10	0.11
L2A-β	**0.91**	**0.84**	0.70	0.59	**0.76**	**0.13**	0.08	0.10	0.11	0.10
L2A-γ	0.76	0.60	0.56	0.54	0.61	0.11	0.12	0.11	0.10	0.11
L2A	0.77	0.76	**0.71**	**0.67**	0.73	0.11	**0.14**	**0.13**	**0.14**	**0.13**

using a four-layer convolutional neural network, as shown in Table 5.3, and the adapter consists of two fully connected layers with ReLU activation Lu et al. (2018b); Glorot et al. (2011).

For the learning rates, we set $\eta_{in} = \eta_{cin} = 0.01$ and $\eta_{out} = \eta_{cout} = 0.001$ on the Rotated MNIST dataset. On the Rotated FashionMNIST dataset, we use the same parameters as MNIST. On the MNIST / FashionMNITS dataset, the η_{in} and η_{out} at the meta-training stage are set as 0.0001 and 0.001, respectively. And the η_{cin} and η_{cout} at the adaptation stage are all set as 0.0001. Furthermore, we train the meta stage with 500 epochs and run the online adaptation stage 5 times to get the average performance. All the experiments are carried out on a 12GB GPU using PyTorch.

5.3.3 Baseline and Ablation Design

EAML Liu et al. (2020c) is the work that is the most related work to the proposed *HMS-CDA* setting. EAML assumes that the target domains evolve incrementally (from mild to severe) over time instead of evolving arbitrarily. Therefore, we cannot compare it with our proposed method directly. To ensure a fair comparison, we change the distribution arbitrarily and test the performance.

Table 5.5: Multi-stream classification accuracy on the Rotated MNIST dataset.

Rotated MNIST					
Source		**Target 1**		**Target 2**	
Rotation	Acc	Rotation	Acc	Rotation	Acc
0°	0.94 ± 0.03	90°	0.23 ± 0.01	90°	0.23 ± 0.02
25°	0.90 ± 0.04	115°	0.24 ± 0.01	180°	0.28 ± 0.01
50°	0.83 ± 0.02	140°	0.30 ± 0.01	205°	0.26 ± 0.01
75°	0.77 ± 0.01	165°	0.29 ± 0.02	230°	0.26 ± 0.02
avg	0.86	avg	0.27	avg	0.26

In addition, we design three ablation forms to evaluate the impact of each component in the framework. Specifically, L2A-α is built without considering the meta-training, L2A-β removes the adapter and L2A-γ is a variant by fixing the meta-representor during online adaptation.

5.3.4 Average Performance

5.3.4.1 The results of the two-stream case

For all benchmarks, we first report the Average Classification Accuracy of the baseline, ablations and the proposed L2A to validate the overall performance of the L2A. Table 5.2 compares the accuracy on the Rotated MNIST dataset and the best result in each column is shown in boldface. Intuitively, L2A-β achieves the best performance for source stream prediction while the whole *L2A* framework performs best for the target stream prediction. L2A-β only accumulates knowledge from labeled source data without adapting knowledge to the target stream, which demonstrates the effectiveness of the designed adapter compared with the full L2A.

Compared with EAML, L2A has a more general prediction ability for both drifting source and target streams while EAML only focuses on the evolving domain adaptation

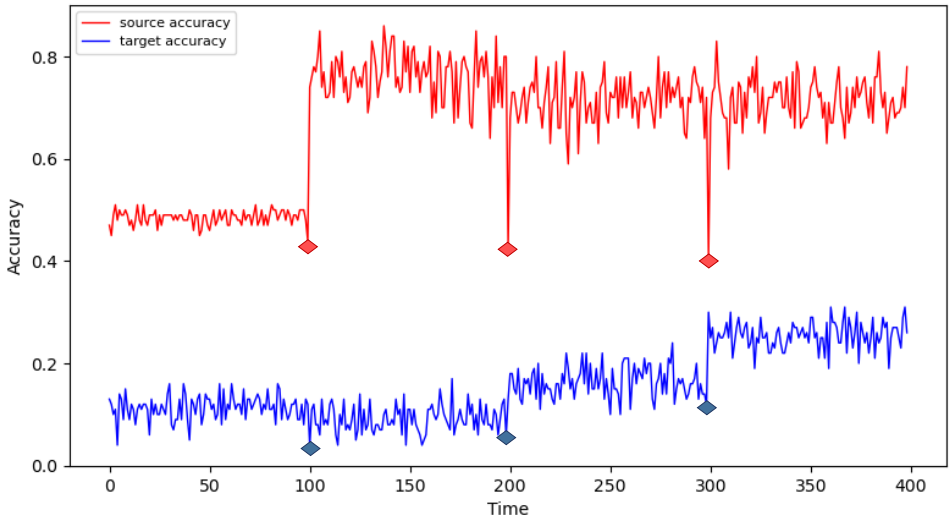
Table 5.6: Multi-stream classification accuracy on the Rotated FashionMNIST dataset.

Rotated FashionMNIST					
Source		**Target 1**		**Target 2**	
Rotation	Acc	Rotation	Acc	Rotation	Acc
0°	0.79 ± 0.05	90°	0.23 ± 0.02	90°	0.21 ± 0.01
25°	0.68 ± 0.03	115°	0.13 ± 0.02	180°	0.15 ± 0.03
50°	0.62 ± 0.03	140°	0.14 ± 0.01	205°	0.12 ± 0.02
75°	0.56 ± 0.03	165°	0.18 ± 0.02	230°	0.19 ± 0.02
avg	0.66	avg	0.17	avg	0.17

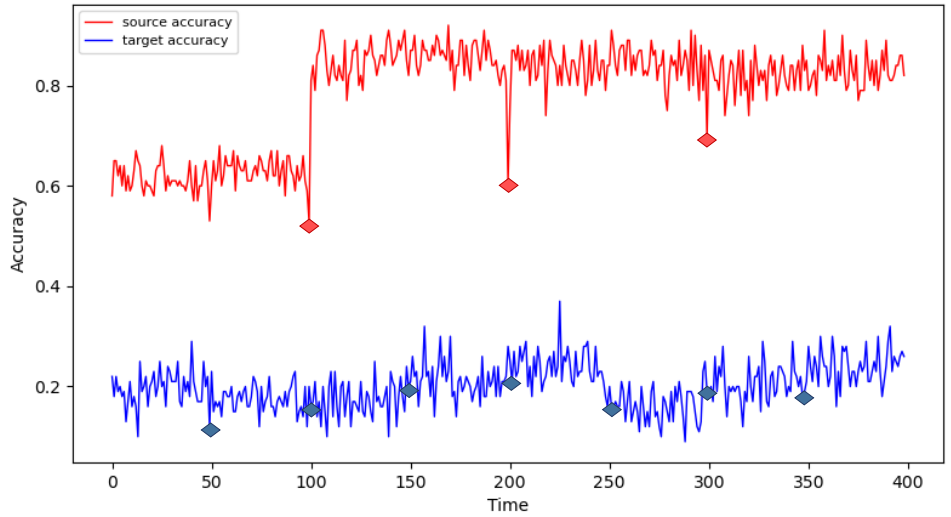
from the source to the target stream. Regarding the target performance, although EAML achieves increasing accuracy and reaches 33% on the 165° rotated MNIST, it is still limited for irregular drifts and thus it is not competitive with L2A in terms of the overall performance (22% v.s. 28%).

Furthermore, the poor performance of L2A-α without meta-training indicates that the meta-representor with prior knowledge is necessary since training is difficult in the non-i.i.d. environment. As previously mentioned, traditional CDA methods are based on hand-craft features and only update the predictors, which limits the feedback benefits between feature extraction and predictor training. Therefore, we fix the meta-representor in L2A-γ and evaluate the performance. As a result, this also proves the superiority of our method. However, because the knowledge is transferred from the source to the target stream continually, it impairs the representation ability for source streams. Therefore, the *L2A* still does not perform better in terms of adapting the drifting source stream.

Table 5.3 details the Average Classification Accuracy on the Rotated FashionMNIST dataset. It shows a similar superior performance to it on Rotated MNIST, which further indicates the effectiveness of the proposed method. In contrast, *L2A* achieves the best performance for source stream prediction. Table 5.4 details the performance of a more



(a) Concurrent drift (Drift occurs per 100-batches in both the source and target streams)



(b) Non-concurrent drift (Drift occurs per 100 batches in the source stream while 50 batches in the target stream)

Figure 5.4: Source and target online accuracy on Rotated MNIST dataset and the locations of concept drift.

complex simulation environment, which exposes a larger gap between source and target distributions. The same superior performance further demonstrates the universality of the proposed framework for the *HMS-CDA* problem.

5.3.4.2 The results of the multi-stream case

As previously discussed, the proposed L2A can easily be extended to a multi-stream situation due to the generalization of the meta-learner obtained at the meta-training stage. Here, we extend the setting into a three-stream case (1-source, 2-target). To ensure the meta-learner can adapt the model correctly to classify the out-of-distribution data, there must be overlapping concepts during meta-training and online learning stages. Thus, the added target stream 2 always includes the same concept that is used in the meta-training stage (i.e., MNIST and FashionMNIS with 90° rotation). As shown in Tables 5.5 and 5.6, L2A achieves almost the same classification accuracy on both target streams, which further demonstrates that L2A can deal with the proposed hybrid multiple streams scenario successfully.

5.3.5 Online Performance

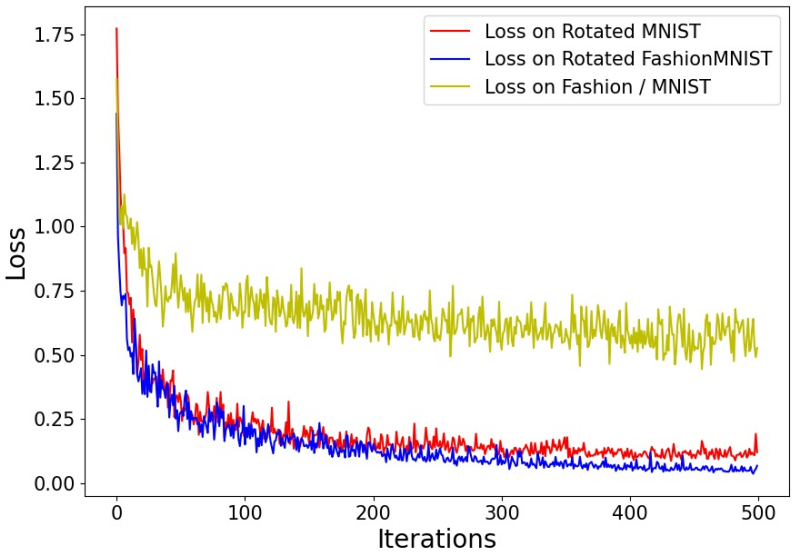


Figure 5.5: The loss curves on three benchmarks.

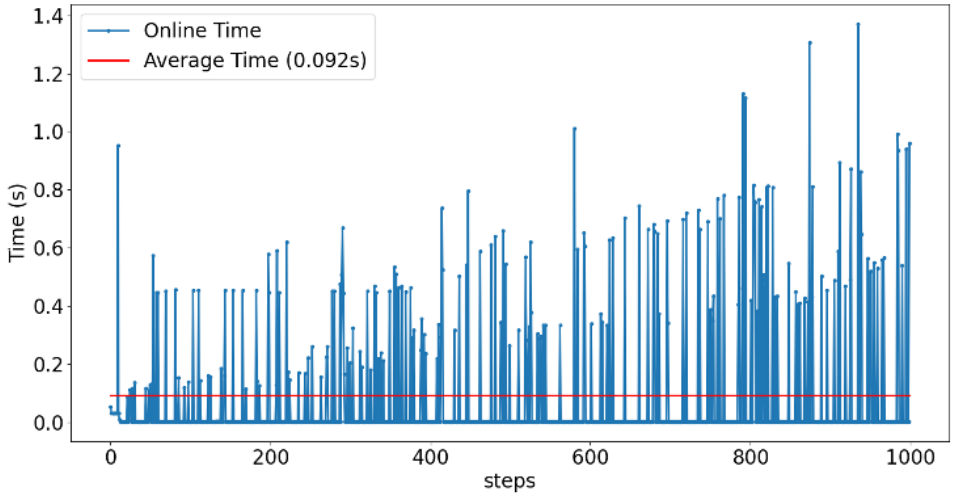


Figure 5.6: Online time on the Rotated MNIST dataset.

We also show the online classification accuracy on the Rotated MNIST dataset in Figure 5.4. Specifically, we first simulate the concurrent drifting streams, in which we set four rotations $\{0°, 25°, 50°, 75°\}$ as the source stream while the FashionMNIST with $\{90°, 115°, 140°, 165°\}$ are regarded as the target stream, and each concept lasts for 100 batches. From Figure 5.4 (a), we can see that when drift occurs in the source stream, the classification accuracy decreases dramatically, and then returns to normal quickly after adaptation. This illustrates that the designed L2A learns a good meta-model and can converge quickly on new concepts with very few training data. From another aspect, its performance on the target stream continues to improve. This is because the meta-learning-based strategy can adapt different streams and accumulate the existing knowledge to accelerate future learning.

In addition, we also simulate the non-concurrent drifting situation and the results are shown in Figure 5.4 (b). In this case, source drifts occur every 100 batches while target drifts occur every 50 batches. This is a similar phenomenon compared to the concurrent drift case when drift occurs in the source stream. However, the accuracy of the target

stream decreases at 250-th, because there is a large distribution shift (from 165° to 90°). However, the performance keeps increasing after the decline and which further shows that our algorithm can still adapt quickly when dealing with large distribution shift.

This shows that when drift occurs (e.g., the 2nd to 3rd time in the target stream), classification accuracy decreases and then it increases quickly on the 4th time due to the adaptation update. Similarly, if there is no drift (e.g., the 0th to 4th time in the source stream), the accuracy will keep increasing by accumulating the incoming knowledge.

5.3.6 Additional Analysis

Convergence: L2A involves two learning procedures, meta-training and online adaptation. Since the performance of online adaptation depends on the meta-training, we test the convergence performance on the three datasets, Rotated MNIST, Rotated FashionMNIST, and Fashion / MNIST, during meta-training. Figure 5.5 shows that the performance loss monotonically decreases with an increase in the iterations and tends to be stable. It indicates that the meta-model converges at the optimal status for invariant representations and ensures good performance for future learning.

Online adaptation time: Time cost is also an important metric to show the superiority of the proposed method. Therefore, we record the online adaptation time on the Rotated MNIST dataset as shown in Figure 5.6. All experiments were conducted on a 12G GPU using PyTorch. Specifically, the time cost can be ignored when there is no drift because the adaptation is not triggered. When drift occurs, the model will be updated to fit the new concept, and the time cost will vary according to the drift severity. From Figure 5.6, we find that most time costs are less than one second and the overall average time for each batch is 0.092 seconds, which shows our model exhibits high performance for drift adaptation.

Sensitivity to batch-size: In this chapter, we sample the 10-way 10-shot concepts

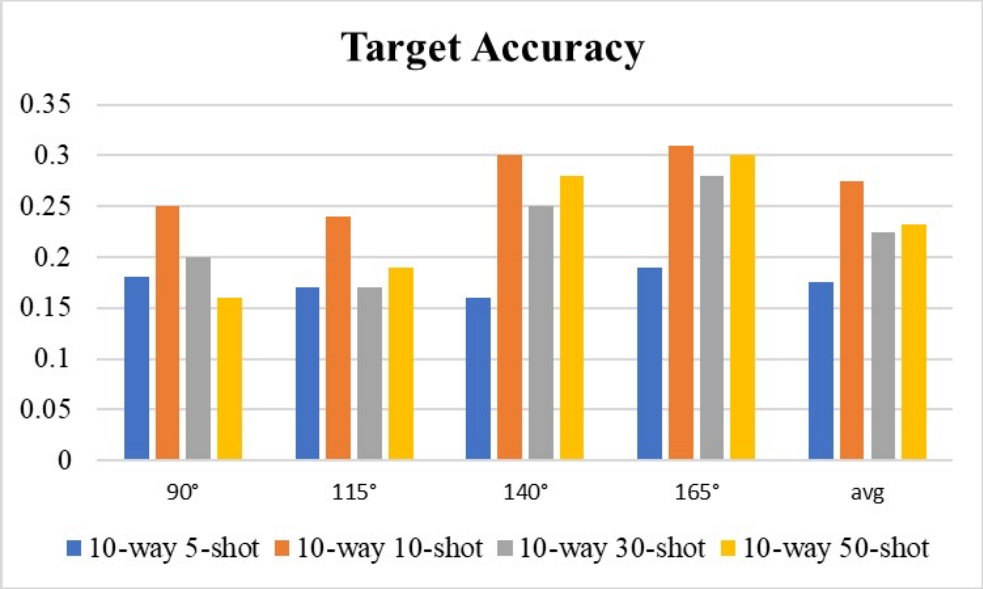


Figure 5.7: Target accuracy under different sampling situations.

and detail the results. To further explore the influence of different sample sizes on classification accuracy, we test different sample sizes (10-way 5-shot, 10-way 10-shot, 10-way 30-shot and 10-way 50-shot) during the online adaptation stage. As shown in Figure 5.7, it is easy to see that the performance on each concept decreases dramatically when reducing the training samples. However, even with an increase in sample size, the overall performance still declines, which comes from two reasons. Firstly, the meta-model must be sensitive to newly arriving data so that it can lead to large improvements for new concepts with very light tuning. Too much data only leads to a performance improvement in the meta-training phase, but severely affects the online adaptation results. Secondly, online fast adaptation requires that the learner can integrate or discard the prior knowledge flexibly when facing newly arriving data instead of overwriting the old knowledge when facing gradual or re-occurring drifts.

Effect of adaptation steps: When taking the target accuracy into consideration, the adaptation performance benefits from a proper learning epoch in the online stage. In Figure 5.8, we plot the target accuracy on four learning epochs (i.e., epoch={1, 3, 5, 10}).

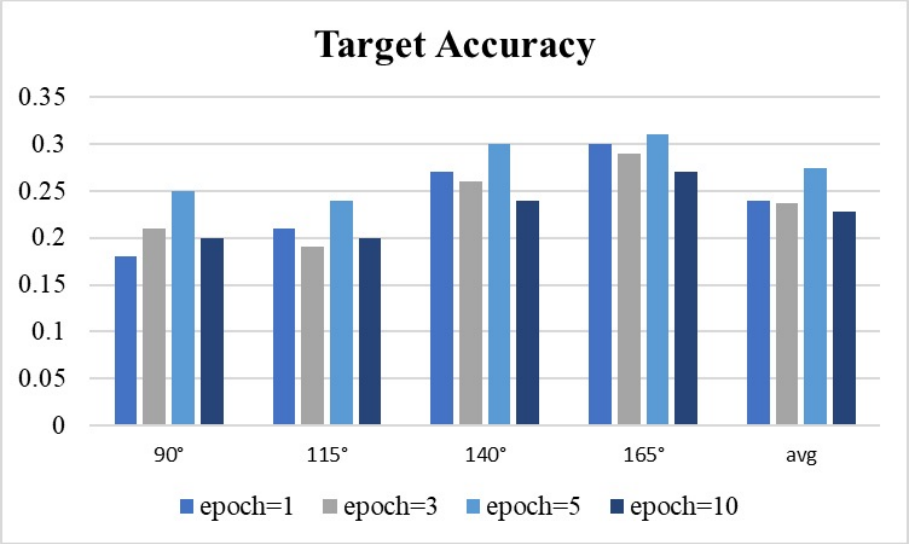


Figure 5.8: Target accuracy with different adaptation steps.

As shown, when the adaptation steps are reduced, it cannot achieve a good convergence on the new concept. It also has a deteriorating accuracy when increasing the adaptation steps. This is because it may cause over-fitting on the current concepts, and it also inevitably causes a time overload.

5.4 Summary

To address the inherent problems of hybrid multiple streams, we propose a new setting, Concept Drift Adaptation for Hybrid Multiple Streams (*HMS-CDA*), which is more aligned to real-world streaming data mining applications. Furthermore, we introduce a **L**earn-to-**A**dapt framework (L2A) to make adaptations for drifting stream prediction. This is the first time the Concept Drift Adaptation (CDA) problem has been addressed from the perspective of deep meta-learning. As a result, extensive experiments and analyses validate the efficacy of our method on the designed *HMS-CDA* benchmarks. Although our *L2A* framework improved the performance of drifting stream prediction, there are still many challenges to be further explored. In this study, we assume that when

knowledge is transferred from the source stream to target streams, it is an independent process which ignores the complex correlations between target streams. Therefore, in future works, we will explore the relationship among dependent streams. In addition, we will integrate the drift detection technique into the CDA method to develop a more general framework for drifting stream mining.

CHAPTER 6

GENERALIZED INCREMENTAL LEARNING UNDER CONCEPT DRIFT

As RQ4 mentioned, while existing research has focused on various aspects of concept drift adaptation, there remains a gap in addressing the continuous learning of both new classes and distribution changes over time. To solve this problem and achieve RO4, this chapter introduces a new setting called Generalized Incremental Learning under Concept Drift (GILCD) where both classes and distributions evolve over time. And a novel framework, Calibrated Source-Free Adaptation (CSFA), is proposed to solve the GILCD problem. It employs training-free calibrated prototypes to incrementally learn novel classes with limited labeled source samples. Additionally, it proposes a source-free adaptation method to address distribution shifts in target streams. In this chapter, the detailed analysis of backgrounds and motivations is introduced in Section 6.1. Definitions, setting details, and the proposed method are listed and explained in Section 6.2. Section 6.3 conducts experiments to evaluate the performance of the proposed method. Section 6.4 summarizes this chapter.

6.1 Introduction

In machine learning, the conventional training process typically relies on pre-collected datasets. It assumes that both training and test data ideally adhere to the same distribution, facilitating the effective generalization of trained models to test data Simon et al. (2022). However, real-world data are often continuously and sequentially generated over time, which is referred to as data streams or streaming data Wang et al. (2021, 2022d). These data streams are susceptible to changes in their underlying distribution, a phenomenon known as concept drift Lu et al. (2018a); Liu et al. (2020a). For instance, autonomous vehicles must navigate dynamically changing environments, which may include natural variations or corruptions such as unforeseen weather conditions or sensor degradation. In addition, these streaming data cannot be stored for extended periods due to storage constraints or privacy concerns. Consequently, the model must be updated incrementally with limited newly arriving data while also making fast adaptations for the concept drift problem.

Previous research has demonstrated the effectiveness of concept drift adaptation techniques in managing dynamic data stream distributions Jiao et al. (2022a). However, the majority of these methods are designed for individual streams with delayed labels, limiting the generalization in more complex drifting scenarios. For example, in autonomous driving systems, multiple sensors or cameras may capture multiple real-time data streams to assist in final decision-making simultaneously, while these data streams exhibit diverse distributions due to the various sources from which data are collected. Furthermore, although data collection is relatively straightforward, the labeling process involves significant time and labor costs, resulting in a hybrid scenario where numerous labeled and unlabeled streams arrive concurrently Yu et al. (2022a). To address this situation, multistream classification has been introduced, involving both labeled and unlabeled data streams with concept drifts Chandra et al. (2016). This task aims to

predict the labels of the target stream by transferring knowledge from one or multiple labeled source streams, while also handling the concept drift problem. However, these studies have overlooked a key issue in real-world applications: learning occurs continuously on incoming data streams, which may contain both data from new classes and new observations of old classes He et al. (2020a). For example, in autonomous vehicles, the decision system needs to continually learn new objects while also retaining knowledge of those it has encountered previously. Fortunately, this problem can be solved by using Class-Incremental Learning (CIL).

CIL methods continually acquire new knowledge while mitigating catastrophic forgetting issues from different perspectives, including knowledge distillation Rebuffi et al. (2017), parameter regularization Ritter et al. (2018), and example replay Gu et al. (2022). However, these methods typically require abundant labeled data for supervised learning. This challenge becomes more pronounced in real-world data streams due to the limited runtime and labeled data available for model updates. Hence, the Few-Shot Class-Incremental Learning (FSCIL) paradigm is introduced to tackle the class-incremental learning problem with limited labeled data Ren et al. (2023); Wang et al. (2024a). FSCIL is prone to overfitting due to the model's propensity to excessively fit the limited labeled data associated with new tasks as well as the catastrophic forgetting problem. However, these works do not consider that the data distribution of newly arriving data may undergo unforeseen distribution changes, i.e., concept drift. This renders the model unable to predict subsequent out-of-distribution samples accurately, leading to a decrease in performance.

To achieve human-level intelligence, learning systems must possess the capability of continual or incremental learning, allowing them to accumulate knowledge over time in response to evolving environmental dynamics, including both class incremental and concept drift problems Eddine Marouf et al. (2023). To fill this research gap, we introduce

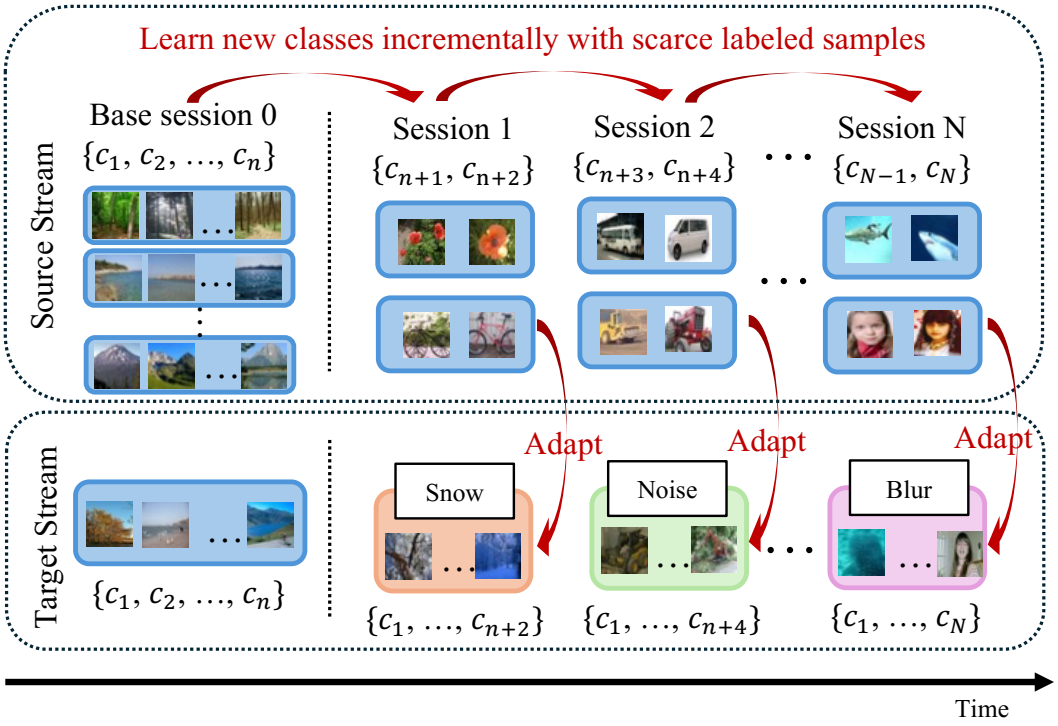


Figure 6.1: Illustration of our proposed Generalized Incremental Learning under Concept Drift (GILCD) setting.

a novel setting called Generalized Incremental Learning under Concept Drift (GILCD). Illustrated in Figure 6.1, this setting assumes the existence of two streams presented session by session in a sequential manner, where both the class and distribution of incoming sessions continuously evolve across sequential learning sessions. Notably, only the incremental sessions of the source stream (i.e., sessions occurring after the base session) provide limited labeled samples of newly emerging classes. The objective of GILCD is to incrementally learn novel classes without forgetting old classes from the source streams while adapting to the distribution changes in the streaming target sessions.

To address the GILCD problem, we propose a **C**alibrated **S**ource-**F**ree **A**daptation (CSFA) framework. Specifically, following previous FSCIL learning methods, we first pre-train a based model using the abundant labeled data in the base session and freeze

it as a feature extractor when dealing with new classes, which can alleviate the catastrophic forgetting problem. To overcome the limited data problem during incremental learning, we adopt a simple yet effective calibrated prototype-based strategy. It not only explores the prototypes of weighted base prototypes but also fuses the biased novel prototypes with weighted base prototypes as the corresponding classifier weights (see Sec. 6.2.3). Furthermore, it is not necessary to optimize or adjust the parameters of feature extractor once the base training finished. We further address the concept drift problem by introducing a source-free adaptation strategy, called **R**eliable Surrogate **G**ap **S**harpness-aware **M**inimization (RGSM). It not only minimizes the perturbation loss and the surrogate gap simultaneously, but it also integrates a reliable indicator function to filter out samples with high entropy, as explained in Sec. 6.2.4. This strategy benefits distribution alignment between the source and target streams as well as improves the generalization capability for changing target distributions.

6.2 Proposed Method

6.2.1 Problem Formulation

As shown in Figure 6.1, we assume there are two data streams in the GILCD setting and each stream contains $N+1$ sessions, including base session and N incremental sessions. We present a sequence of disjoint source sessions by $D_S = \{D_{S0}, D_{S1}, ..., D_{Sn}\}$, where D_{S0} is a large-scale base dataset with abundant labeled samples and the following $D_{Si}, i > 0$ are all novel sessions with scare labeled samples. For the source data D_{Si} in the i-th session, we further define it as $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|N_i|}$ with the corresponding label space C_{Si}, where $C_{Si} \cap C_{Sj} = \emptyset$. Accordingly, a evolving target stream is defined as $D_T = \{D_{T0}, D_{T1}, ..., D_{Tn}\}$. The target label space and distribution of session i are denoted as C_{Ti} and P_{Ti}, respectively. Specifically, the target label space of i-th session contains

all seen classes during inference. Furthermore, the distributions of target sessions also change over time, i.e., $P_{Ti} \neq P_{Tj}, i \neq j$. Overall, the challenges which arise in the GILCD setting can be summarized as follows,

Challenge 6.1 ***Class-Incremental.*** *For the source stream D_S, the corresponding label space C_{Si} of each session $D_{Si}, i > 0$ incrementally evolving, i.e., $C_{Si} \cap C_{Sj} = \emptyset$, whereas the target label space of the i-th session contains all seen classes during inference, i.e., $C_{Ti} = \bigcup_{j=0}^{i} C_{Sj}$.*

Challenge 6.2 ***Scarcity of Labels.*** *Only limited labeled samples are provided to the source streams $D_S = \{D_{S0}, D_{S1}, ..., D_{Sn}\}$, leaving the target stream entirely unlabelled $D_T = \{D_{T0}, D_{T1}, ..., D_{Tn}\}$. A session in the source stream can alternatively be described as an N-way K-shot classification task, which involves N classes and K labeled examples for each class. Consequently, the challenge lies in achieving accurate predictions in the target stream, where no labeled samples are available.*

Challenge 6.3 ***Covariate Shift.*** *Denoting P_{Si} and P_{Ti} as the distributions of D_{Si} and D_{Ti} in the incremental sessions, all streams at the same session step are related but with covariate shift, i.e., $P_{Si}(\boldsymbol{x}) \neq P_{Ti}(\boldsymbol{x})$*

Challenge 6.4 ***Target Drift.*** *Target drift refers to the distributions of target sessions that change over time, i.e., $P_{Ti} \neq P_{Tj}, i \neq j$.*

The primary objective of GILCD is to develop a unified classification model $f(\boldsymbol{x})$ capable of effectively handling all these challenges. This task requires the model to achieve three key goals: *1) incrementally learn new classes with limited labeled data without forgetting old classes*; *2) continuously make adaptations from the labeled source stream to the unlabeled target stream*; and *3) dynamically adapt to new distributions for target stream prediction*.

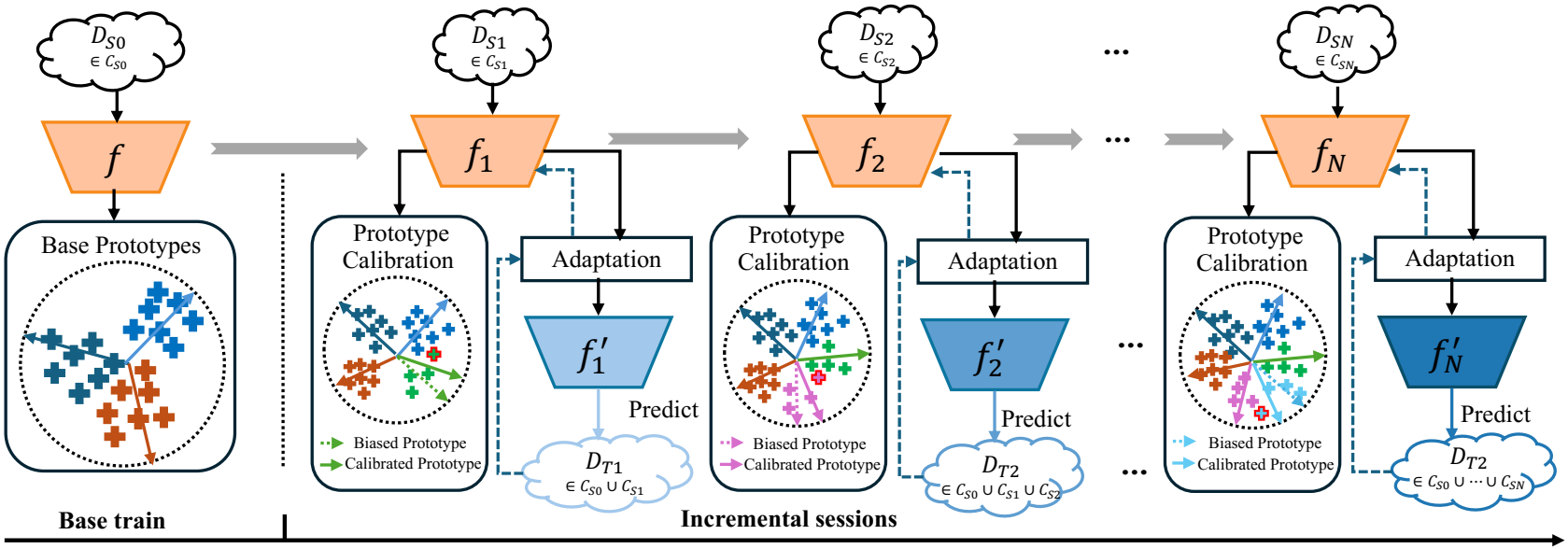


Figure 6.2: Illustration of the proposed CSFA framework. Firstly, a base model is trained on the large-scale base session D_{S0}. When dealing with new sessions with limited samples, the pre-trained feature extractor is frozen, and a calibrated prototype-based strategy is adopted to learn novel prototypes incrementally. In addition, it further addresses the covariate shift and target drift by introducing a source-free adaptation strategy, which jointly minimizes the entropy and the sharpness of the entropy of those reliable target samples from the new distribution.

6.2.2 Calibrated Source-Free Adaptation

To address the inherent challenges of the GILCD task, we propose a novel method called Calibrated Source-Free Adaptation (CSFA), illustrated in Figure 6.2. Initially, in response to Challenges 6.1 and 6.2, we commence by pre-training a base model on abundant labeled data from the base session, following the precedent set by previous FSCIL methods. Subsequently, we freeze this model and utilize it as a feature extractor when encountering new classes, thus mitigating the adverse effects of catastrophic forgetting. We introduce a simple yet effective calibrated prototype-based strategy to overcome the challenge of limited data availability during incremental learning. This strategy not only examines the prototypes of weighted base prototypes but also integrates biased prototypes of new classes with base prototypes to determine the corresponding classifier weights (see Sec. 6.2.3). Importantly, this approach eliminates the need for additional optimization procedures after the base training.

Furthermore, we address Challenges 6.3 and 6.4 by devising a source-free adaptation strategy. This strategy operates by simultaneously minimizing the perturbation loss and the surrogate gap as well as integrating a reliable indicator function to filter out samples with high entropy. RGSM not only removes the high loss within a neighborhood but also ensures that the obtained minimum is situated within a flat region. A detailed analysis is elucidated in Sec. 6.2.4. Through this multifaceted approach, we aim to fortify the robustness and adaptability of incremental learning systems, ensuring their efficacy in dynamic environments.

6.2.3 Class-Incremental Learning

Base Model Training: Firstly, we use the collected data $D_{S0} \in \{(x_i, y_i)\}_{i=1}^{|N_0|}$ with abundant labeled training data to train a base model $f(\boldsymbol{x})$, which follows a standard classifica-

tion pipeline and is optimized by cross-entropy loss,

$$\sum_{(\boldsymbol{x}_j, y_j) \in D_{S0}} L_E \left(f\left(\boldsymbol{x}_j; \theta\right), y_j \right) \tag{6.1}$$

where L_E denotes the cross-entropy loss. $x_i \in \mathbb{R}^d$ is a training sample with label y_i from the base label space. Following Rebuffi et al. (2017); Wang et al. (2022f), the model can be denoted as $f(\boldsymbol{x}) = W^\top \phi(\boldsymbol{x})$, where $\phi(\cdot) : \mathbb{R}^D \rightarrow \mathbb{R}^d$ is the feature extractor and $W \in \mathbb{R}^{d \times N}$ is the classification head. If there are N classes, the classifier head can be denoted by the N prototypes, i.e., $W = [c_1, c_2, \ldots, c_N]$.

Incremental Learning: Prototype classifier Snell et al. (2017) is a popular approach employed in few-shot learning tasks. To achieve this, the feature extractor $\phi(\cdot)$ trained by base session remains fixed to minimize catastrophic forgetting and then we utilize the mean feature $\mathbf{c}_k$ as the prototype to capture the most common pattern observed in each class,

$$\mathbf{c}_k = \frac{1}{num_k} \sum_{y_i = k} f_\phi(\boldsymbol{x}_i), \tag{6.2}$$

where num_k is the sample number belonging to the kth class. The prototypes of new classes are utilized as the corresponding classifier weights Zhang et al. (2021a); Zhou et al. (2022). Then we can estimate the probability of each class k via the dot products between the features and each prototype via,

$$p\left(y_i = k | \phi(\boldsymbol{x_i})\right) = \frac{\exp(\phi(\boldsymbol{x_i}) \cdot \boldsymbol{c}_k)}{\sum_j \exp(\phi(\boldsymbol{x_i}) \cdot \boldsymbol{c}_j)}. \tag{6.3}$$

However, in GILCD settings, the number of samples in the new class is extremely limited, leading to severe bias in the empirical prototypes of the novel classes. Inspired by Wang et al. (2024a), we utilize the well-learned base prototypes to calibrate the biased prototypes of novel classes during incremental sessions. To achieve this, we define the base session as comprising n classes and the incremental session as containing C additional classes. Hence, the base prototypes and new prototypes are denoted as

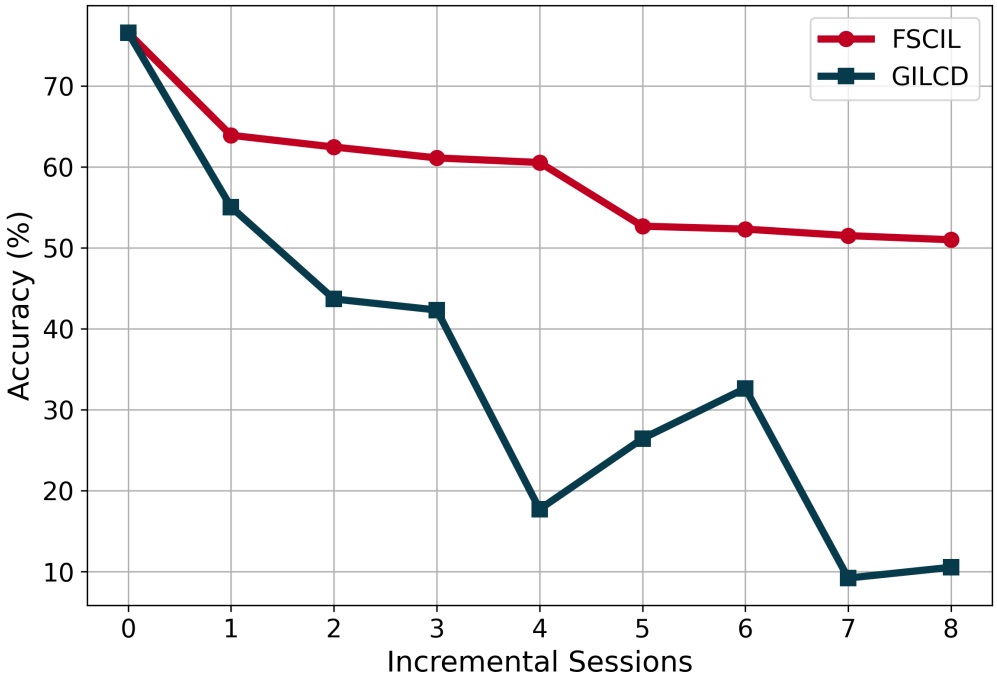


Figure 6.3: Comparison of TEEN's performance between FSCIL and GILCD settings. We compare the performance of TEEN Wang et al. (2020) in the stationary FSCIL scenario (i.e., CIFAR100) with our proposed GILCD setting with concept drift (i.e., CIFAR100-C). It is evident that the current FSCIL methods fail to address the distribution shift in the GILCD scenario.

$c_b (1 \leq b \leq n)$ and $c_{new} (n < b \leq n + C)$, respectively. Since the base session includes adequate samples in every class, the model can effectively capture the distribution of base classes and derive guaranteed prototypes. Consequently, we can leverage these base prototypes c_b to calibrate the prototypes for newly coming classes c_{new} by,

$$\bar{c}_{new} = \alpha c_n + (1 - \alpha) \Delta c_{new}, \tag{6.4}$$

where $\bar{c}_{new}$ is the calibrated new prototype, and hyper parameter α controls the calibration strength of new prototypes. Δc_{new} denotes the calibration item, which is constructed by weighted base prototypes. Specifically, the cosine similarity $S_{b,new}$ between c_{new} and

Algorithm 6 The pipeline of proposed CSFA

Input: Source stream $D_S = \{D_{S0}, D_{S1}, ..., D_{Sn}\}$; Target stream $D_T = \{D_{T0}, D_{T1}, ..., D_{Tn}\}$, and model f

Output: Predictions for target stream.

1: **Base train:**
2: Train a based model using D_{S0}.
3: Calculate the base prototypes via Eq.(6.2)
4: **Incremental learning:**
5: **for** $i = 1 : n$ **do**
6: Calculate the weight of new prototypes via Eq.(6.6).
7: Calculate the calibrated prototypes via Eq.(6.7).
8: Adapt parameters by minimizing Eq.(6.15).
9: **end for**

a base prototype c_b is obtained by,

$$S_{b,new} = \frac{c_b \cdot c_{new}}{\|c_b\| \cdot \|c_{new}\|} \cdot \tau, \tag{6.5}$$

where τ is the scaling hyperparameter and $\tau > 0$. Then the weight of the new class prototype c_{new} can be represented as the softmax output over all base prototypes,

$$w_{b,new} = \frac{e^{S_{b,new}}}{\sum_{i=1}^{n} e^{S_{i,new}}}, \tag{6.6}$$

Finally, the well-calibrated prototypes of novel classes is defined as follows,

$$\begin{aligned} \bar{c}_{new} &= \alpha c_{new} + (1-\alpha)\Delta c_{new} \\ &= \alpha c_{new} + (1-\alpha)\sum_{b=1}^{n} w_{b,new} c_b. \end{aligned} \tag{6.7}$$

6.2.4 Source-Free Drift Adaptation

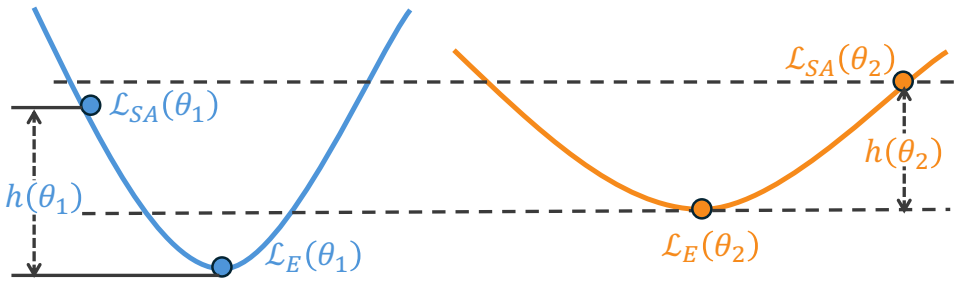
In addition to the challenges posed by class incremental learning, the GILCD setting also faces the significant issue of concept drift. When there is a substantial covariate shift between the target stream and the source stream used for class incremental learning, the learned model will suffer from severe performance degradation. As shown in Figure 6.3, we compare the performance of the classic FSCIL method TEEN Wang et al. (2020)

in a stable scenario (i.e., CIFAR100) with our proposed GILCD setting with distribution changes (i.e., CIFAR100-C). The current FSCIL methods fail to address the distribution shift issue in the GILCD scenario. This necessitates the continuous adaptation of the model from the labeled source stream to the unlabeled target stream, while dynamically adapting to new distributions for target stream prediction. Moreover, only limited labeled samples can be available in the source stream while leaving the target stream unlabeled, which exacerbates the difficulty of knowledge transfer.

Inspired by TTA methods Wang et al. (2020), we can alleviate distribution gaps by minimizing empirical risk on model predictions of unlabeled samples from diverse distributions. This approach presents a promising paradigm for addressing distribution shifts across domains without requiring access to source data. For example, TENT Wang et al. (2020) aims to minimize the entropy of model predictions during test-time adaptation. However, these empirical risk minimization approaches frequently encounter overfitting issues during training, leading to convergence towards sharp minima Li et al. (2023b). This phenomenon can subsequently lead to diminished performance when applied to out-of-distribution samples. Thus, the Sharpness-Aware Minimization (SAM) method Foret et al. (2021) is proposed to guide the model towards a flat region on the entropy loss surface. To achieve this goal, SAM solves the following min-max optimization problem:

$$\begin{aligned} &\min_{\theta} L_{SA}(D;\theta), \\ &\text{where } L_{SA}(D;\theta) \triangleq \max_{\|\epsilon\|_2 \leq \rho} L_E(D;\theta+\epsilon). \end{aligned} \tag{6.8}$$

where ρ denotes a predefined parameter to control the radius of the neighborhood, which is set by the default value 0.05 following Niu et al. (2023). $L_E(\boldsymbol{x};\theta)$ is the cross-entropy. Given θ, the maximization in Eq.(6.8) aims to find the weight perturbation $\boldsymbol{\epsilon}$ in the Euclidean ball with radius ρ that maximizes the empirical loss. The sharpness is quantified by the maximal change of entropy between θ and $\theta+\epsilon$. To address this



$$Sharpness(\theta_1) > Sharpness(\theta_2);$$
$$\mathcal{L}_{SA}(\theta_1) < \mathcal{L}_{SA}(\theta_2); \quad h(\theta_1) > h(\theta_2)$$

Figure 6.4: Consider a scenario with a sharp local minimum θ_1 and a flat local minimum θ_2. The loss surface around θ_2 appears flatter than that around θ_1. However, SAM exhibits bias towards selecting θ_1 over θ_2 because $L_{SA}(\theta_1) < L_{SA}(\theta_2)$. Instead, the surrogate gap $h(\theta)$ provides a better description of the sharpness of the loss surface. A smaller $h(\theta_2)$ indicates that θ_2 is flatter than θ_1.

problem, it utilizes the first-order Taylor expansion to approximate its solution by,

$$\begin{aligned}\epsilon^*(\theta) &\triangleq \underset{\|\epsilon\|_2 \le \rho}{\arg\max} L_E(D;\theta+\epsilon) \\ &\approx \underset{\|\epsilon\|_2 \le \rho}{\arg\max} L_E(D;\theta) + \epsilon^T \nabla L_E(D;\theta) \\ &= \underset{\|\epsilon\|_2 \le \rho}{\arg\max} \epsilon^T \nabla_\theta L_E(D;\theta).\end{aligned} \tag{6.9}$$

Subsequently, the solution to this approximation, denoted as $\hat{\epsilon}(\theta)$, is derived from resolving a classical dual norm problem,

$$\hat{\epsilon}(\theta) = \rho\, \text{sign}(\nabla L_E(D;\theta)) \frac{|\nabla L_E(D;\theta)|}{\|\nabla L_E(D;\theta)\|_2}. \tag{6.10}$$

Substituting $\hat{\boldsymbol{\epsilon}}(\Theta)$ into Eq.(6.8) and omitting second-order terms to expedite computation yields the final gradient approximation,

$$\nabla L_{SA}(D;\theta) \approx \nabla L_E(D;\theta)|_{\theta+\hat{\epsilon}(\theta)}, \tag{6.11}$$

In other words, SAM aims to find a minimum on the surface of the perturbed loss $L_{SA}(D;\theta+\hat{\epsilon})$ instead of the original entropy $L_E(D;\theta)$. However, it has been proven that a smaller $L_{SA}(D;\theta+\hat{\epsilon})$ cannot guarantee a flatter region Zhuang et al. (2022). As

shown in Figure 6.4, the loss surface around the local minimum θ_2 appears flatter than that around θ_1. However, SAM is biased towards selecting θ_1 over θ_2 because $L_{SA}(\theta_1) < \mathbf{L}_{SA}(\theta_2)$, which favors a sharper minimum and could potentially compromise the model's ability to generalize effectively. Therefore, Zhuang et al. Zhuang et al. (2022) proposed the Surrogate Gap Guided Sharpness-Aware Minimization (GSAM) to minimize the sharpness-aware entropy and the surrogate gap simultaneously by,

$$L_{GSAM} = \min_{\theta}(L_{SA}(D;\theta), h(\theta)), \tag{6.12}$$

where,

$$h(\theta) \triangleq L_{SA}(D;\theta) - L_E(D;\theta). \tag{6.13}$$

The surrogate gap $h(\theta)$ quantifies the difference between the maximum loss within the neighborhood and the minimum point, offering a more precise characterization of sharpness. To solve this problem, GSAM adopts a two-step approach for updating: 1) it employs gradient descent $\nabla L_{SA}(D;\theta)$ to minimize the sharpness-aware entropy loss $L_{SA}(D;\theta)$, following SAM Foret et al. (2021); 2) it decomposes the gradient $\nabla L_E(D;\theta)$ of the entropy loss $L_E(D;\theta)$ into two components that are parallel and orthogonal to $\nabla L_{SA}(D;\theta)$, i.e., $\nabla L_E(D;\theta)_{\parallel}$ and $\nabla L_E(D;\theta)_{\perp}$. Subsequently, it performs an ascent step in $\nabla L_E(D;\theta)_{\perp}$ to minimize the surrogate gap $h(\theta)$. As a result, the final gradient direction of GSAM can be expressed as,

$$\nabla L_{GSAM} = \nabla L_{SA}(D;\theta) - \beta \nabla L_E(D;\theta)_{\perp}, \tag{6.14}$$

where β represents a hyperparameter used to determine the ascent step size. Despite the fact that updating along the direction of the orthogonal component $\nabla L_E(D;\theta)_{\perp}$ does not alter the value of the perturbed loss $L_{SA}(D;\theta)$, it does lead to an increase in $L_E(D;\theta)$ Wang et al. (2023b). Given that the surrogate gap is always non-negative, a high value of $L_E(D;\theta)$ can impede the decrease of $L_{SA}(D;\theta)$, potentially harming the model's

generalization performance when dealing with diverse target distributions. Moreover, target samples from different distributions can yield substantial gradients, which may negatively impact adaptation and result in model collapse Niu et al. (2023); Foret et al. (2021).

Hence, we suggest the additional step of discarding samples with significant gradients based on their entropy. Consequently, the comprehensive proposed **R**eliable Surrogate **G**ap **S**harpness-aware **M**inimization (RGSM) loss can be formulated as follows:

$$L_{RGSM} = \min_{\theta}(G(D)L_{SA}(D;\theta), h(\theta)), \tag{6.15}$$

where $G(D) \triangleq \mathbb{I}_{\{L_E(D;\Theta)<E_0\}}(D)$ is a reliable indicator function which can remove the samples that have large gradients or low confidence out of adaptation, and E_0 is a pre-defined parameter, which is set to $0.4 \times \ln 1000$ following Niu et al. (2023). Similarly, we adopt a two-step approach for each update. Firstly, we employ gradient descent $\nabla G(D)L_{SA}(D;\theta)$ to minimize the reliable sharpness-aware entropy loss $G(D)L_{SA}(D;\theta)$. Furthermore, during the adaptation process, model collapse is prone to occur, leading to extremely small entropy losses that hinder model updates. To mitigate this issue, a moving average e_m (which is set to 0.9) of entropy loss values is also employed here. Once e_m falls below a predefined threshold e_0 (which is set to 0.2), the learned parameter θ will be reset to its original value. Secondly, it decomposes the gradient $\nabla G(D)L_E(D;\theta)$ of the reliable entropy loss $G(D)L_E(D;\theta)$ into two components that are parallel and orthogonal to $\nabla G(D)L_{SA}(D;\theta)$, i.e., $\nabla G(D)L_E(D;\theta)_{\|}$ and $\nabla G(D)L_E(\theta;D)_{\perp}$. Subsequently, it performs an ascent step in $\nabla G(D)L_E(\theta;D)_{\perp}$ to minimize the surrogate gap $h(\theta)$. Therefore, the final gradient direction of RGSM can be formulated as,

$$\nabla L_{RGSM} = \nabla G(D)L_{SA}(D;\theta) - \beta \nabla G(D)L_E(D;\theta)_{\perp}. \tag{6.16}$$

Our proposed RGSM not only simultaneously minimizes the perturbation loss $L_{SA}(\theta;D)$ and the surrogate gap $h(\theta)$ but also introduces a reliable indicator function $G(D)$ to filter

out samples with high entropy. Thus, RGSM ensures that the loss within a neighborhood of the desired minimum is adequately low, and the resulting minimum is situated within a flat loss surface.

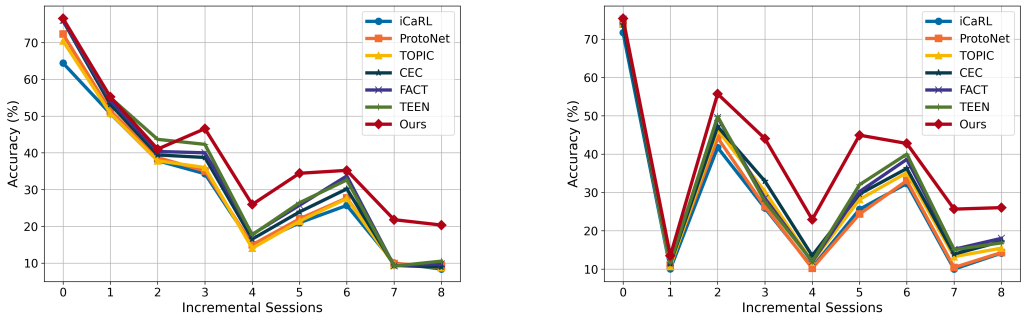
6.3 Experiments

This section begins by introducing the dataset benchmarks utilized in the experiments and elucidates how to simulate the GILCD setting. Subsequently, we talk about the compared methods, including the state-of-the-art methods of CIL, FSL, FSCIL, and source-free adaptation methods. In our experiments, we initially empirically showcase that CSFA consistently surpasses the current methods, underscoring its robustness and superiority. Following this, we validate the effectiveness of each component proposed in CSFA through an ablation study. Additionally, we delve into the influence of incremental shots and conduct the parameter analysis.

6.3.1 Datasets

In our experiment, we employed three commonly used datasets for few-shot class incremental learning, i.e., CIFAR100 Krizhevsky et al. (2009), CUB200-2011 Wah et al. (2011), and miniImageNet Russakovsky et al. (2015). CIFAR-100 is a dataset comprising 100 classes with 600 32 × 32 images each. CUB200 is a dataset consisting of 200 bird species, with 11,788 images in total. It is commonly used for fine-grained visual categorization and bird species recognition tasks in computer vision research. The images are resized to 256 × 256 and then cropped to 224 × 224 for training. miniImageNet is a subset of the ImageNet dataset with 100 classes and each class contains 600 84 × 84 images.

Source Stream: Following the FSCIL paradigm Bai et al. (2020); Tao et al. (2020), we organize the training sets of CIFAR-100, miniImageNet, and CUB-200-2011 into base and incremental sessions to construct the source stream. Specifically, for CIFAR-100 and



(a) CIFAR100-C (b) miniImageNet-C

Figure 6.5: Comparison with CIL/FSL/FSCIL baselines on CIFAR100-C and miniImageNet-C datasets. The corruptions for sessions 1-8 are 'contrast', 'elastic transform', 'zoom blur', 'glass blur', 'frost', 'fog', 'Gaussian noise', 'shot noise'.

miniImageNet, the training data is initially partitioned into 60 base classes, with the remaining 40 classes divided into eight 5-way 5-shot few-shot classification tasks. As for CUB200, the 200 classes are divided into 100 base classes and 100 incremental classes. The new incremental classes are then structured into 10-way 5-shot incremental tasks.

Target Stream: To simulate various distributions, we generated three datasets, namely CIFAR100-C, CUB200-C, and miniImageNet-C, by introducing visual corruptions Hendrycks and Dietterich (2018). Specifically, we added 15 types of corruption across four main categories (noise, blur, weather, and digital) with severity levels 5. For the drifting target stream, we randomly selected one type of corruption from the 15 available types for each incremental session, ensuring that the target stream exhibits a covariate shift compared to the source stream. Furthermore, this selection methodology guarantees the presence of distribution drift in the target stream with new sessions arriving.

Table 6.1: Comparison with CIL/FSL/FSCIL baselines on CUB200-C dataset. The corruptions for sessions 1-10 are 'pixelate', 'elastic transform', 'defocus blur', 'zoom blur', 'glass blur', 'snow', 'fog', 'Gaussian noise', 'shot noise', 'impulse noise'.

Method	Accuracy in each session (%)											Average Accuracy (%)
	0	1	2	3	4	5	6	7	8	9	10	
iCaRL	76.73	56.25	50.21	41.79	23.98	20.42	10.76	15.49	10.04	9.79	9.42	29.53
ProtoNet	77.81	56.97	53.42	44.32	25.01	23.18	10.91	17.62	10.71	11.01	9.37	30.94
TOPIC	77.49	58.32	55.14	45.01	25.32	24.19	12.11	19.73	10.91	11.57	9.09	31.72
CEC	79.62	60.79	57.48	48.32	27.03	26.21	14.37	22.79	12.35	13.2	10.17	33.85
FACT	80.73	61.01	58.76	47.40	28.63	27.42	15.48	23.21	13.72	14.37	10.09	34.62
TEEN	80.29	62.39	61.31	47.59	28.40	26.77	16.02	24.10	13.39	15.54	11.10	37.58
Ours	**81.87**	**65.23**	**62.25**	**49.21**	**28.77**	**34.23**	**33.96**	**36.37**	**26.61**	**27.53**	**20.83**	**42.44**

Table 6.2: Comparison with TTA/DG baselines on CIFAR100-C and miniImageNet-C datasets. The corruptions for sessions 1-8 are 'contrast', 'elastic transform', 'zoom blur', 'glass blur', 'frost', 'fog', 'Gaussian noise', 'shot noise'.

Datasets	Method	Accuracy in each session (%)									Average Accuracy (%)
		0	1	2	3	4	5	6	7	8	
CIFAR100-C	+TENT	76.55	53.62	38.79	43.21	23.62	30.87	32.59	18.76	18.06	37.34
	+SAM	76.55	54.57	39.51	45.18	24.42	32.19	34.01	19.73	18.23	38.27
	+SAR	76.55	55.13	40.42	45.86	24.73	33.07	34.62	21.09	19.74	39.02
	+GSAM	76.55	55.07	40.43	46.03	24.35	32.67	34.71	20.10	19.85	38.86
	Ours	76.55	**55.23**	**41.02**	**46.56**	**25.94**	**34.44**	**35.23**	**21.81**	**20.35**	**39.68**
miniImageNet-C	+TENT	75.37	12.74	50.78	41.59	19.07	39.45	38.39	22.71	22.64	35.86
	+SAM	75.37	13.21	52.79	41.76	20.41	41.72	39.02	23.22	23.07	36.73
	+SAR	75.37	13.16	54.13	43.22	21.76	43.23	40.37	24.43	25.92	37.95
	+GSAM	75.37	**14.67**	54.68	43.74	21.19	43.01	41.04	25.04	**26.41**	38.35
	Ours	**75.37**	13.51	**55.71**	**44.03**	**22.93**	**44.94**	**42.78**	**25.65**	26.04	**38.99**

6.3.2 Baselines

To validate our approach, we commence by comparing it with the state-of-the-art methods of CIL, FSL, and FSCIL:

- iCaRL Rebuffi et al. (2017) stands out as a prominent approach for class-incremental learning. It adopts nearest-mean-of-exemplars classifiers to store exemplars from each class and utilizes a herding-based strategy for exemplar selection. By integrating these exemplars with distillation, it effectively learns data representations to overcome catastrophic forgetting.

- ProtoNet Snell et al. (2017) is a widely recognized method in the field of few-shot learning. It constructs a metric space where classification is executed by measuring distances to prototype representations of individual classes.

- TOPIC Tao et al. (2020) introduces a neural gas network designed to grasp and maintain feature topology across diverse classes. It can effectively prevent the forgetting problem and enhance representation learning for new classes by dynamically adapting the network to new samples.

- CEC Zhang et al. (2021a) proposes a continually evolved classifier for few-shot class-incremental learning. It incorporates a graph attention network to dynamically update classifier weights, leveraging a global context across all sessions.

- FACT Zhou et al. (2022) emphasizes the necessity of constructing forward-compatible models for few-shot class incremental learning. By pre-assigning virtual prototypes in an embedding space, the model becomes anticipatory and expandable, effectively mitigating the impact of model updates and improving inference performance.

- TEEN Wang et al. (2024a) addresses the issue of biased prototypes for new classes by calibrating them with well-calibrated prototypes from old classes. Also, this

method eliminates the need for additional optimization procedures.

In addition, we also compare our proposed adaptation method with current Test-time Adaptation (TTA) and Domain Generalization (DG) methods to demonstrate its adaptability to different distributions.

- TENT Wang et al. (2020) is a fully test-time adaptation method that reduces generalization error on shifted data through entropy minimization without accessing the source data.
- SAR Niu et al. (2023) is a more stable and reliable fully test-time adaptation method, which mitigates the impact of noisy test samples with large gradients.
- SAM Foret et al. (2021) is proposed for domain generalization, enhancing generalization by minimizing both loss value and sharpness simultaneously.
- GSAM Zhuang et al. (2022) introduces a surrogate gap to quantify the difference between the maximum loss within the neighborhood and the minimum point. This approach offers a more precise characterization of sharpness and enhances generalization by simultaneously minimizing sharpness-aware entropy and the surrogate gap.

Table 6.3: Comparison with TTA/DG baselines on CUB200-C dataset. The corruptions for sessions 1-10 are 'pixelate', 'elastic transform', 'defocus blur', 'zoom blur', 'glass blur', 'snow', 'fog', 'Gaussian noise', 'shot noise', 'impulse noise'.

Method	Accuracy in each session (%)											Average Accuracy (%)
	0	1	2	3	4	5	6	7	8	9	10	
+TENT	81.87	62.08	59.72	46.67	25.49	32.41	30.87	32.51	24.33	24.72	18.64	39.93
+SAM	81.87	63.19	61.49	48.22	26.98	33.79	32.19	34.10	25.32	25.79	18.87	41.07
+SAR	81.87	64.03	62.01	**49.65**	27.72	33.94	33.41	35.29	26.07	27.25	20.27	41.95
+GSAM	81.87	64.76	61.72	48.64	27.57	34.07	33.21	35.45	25.97	**27.62**	19.42	41.84
Ours	81.87	**65.23**	**62.25**	49.21	**28.77**	**34.23**	**33.96**	**36.37**	**26.61**	27.53	**20.83**	**42.44**

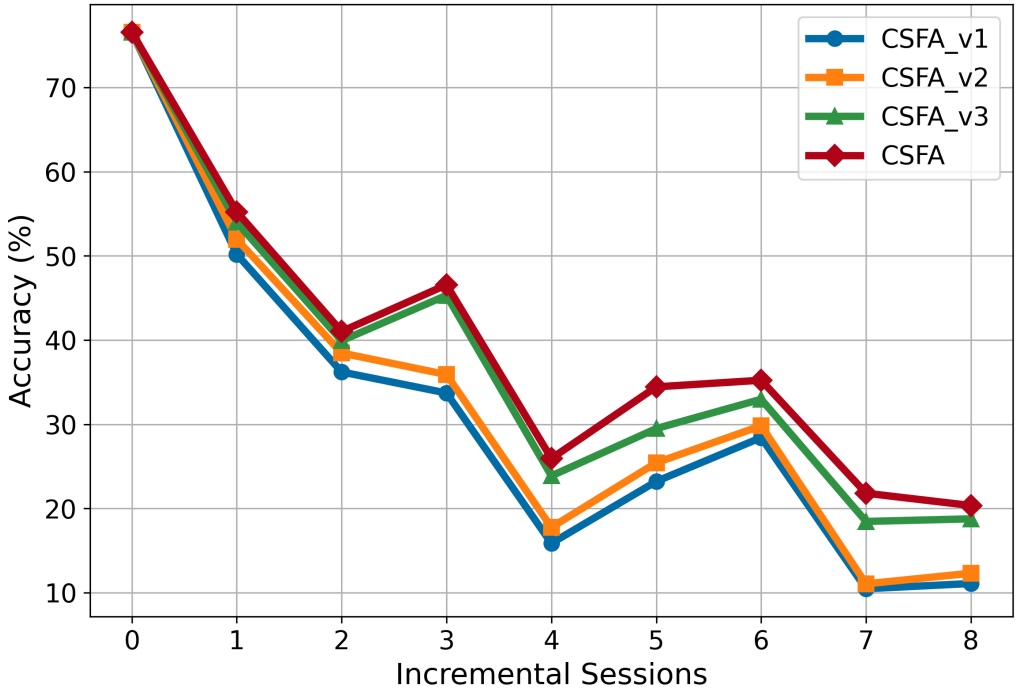


Figure 6.6: Incremental accuracy (%) of CSFA variants.

6.3.3 Implementation

In the experiment, we adopt a ResNet20 architecture for CIFAR100-C, a pre-trained ResNet18 for CUB200-C, and a randomly initialized ResNet18 for MiniImageNet-C. To ensure a fair comparison, all methods under evaluation utilize identical backbone networks and initialization protocols. For training the feature extractor on CIFAR100-C and MiniImageNet-C, we employ a learning rate of 0.1 and a batch size of 256. Furthermore, we adjust the learning rate to 0.004, set the batch size to 128, and conduct training over 400 epochs on CUB200-C. To manage learning rate adjustments, we implement a cosine scheduler. For our RGSM adaptation process, we use SGD to update the model with a momentum of 0.9 and the learning rate is 0.0001. All experiments are conducted using PyTorch on a single A100 GPU.

6.3.4 Results Comparison

Comparison with CIL/FSL/FSCIL methods. To validate our method, we conducted comparisons with current approaches in CIL/FSL/FSCIL on three benchmarks, i.e., CIFAR100-C, CUB200-C, and miniImagenet-C. Table 6.1 shows all incremental and average accuracies on the CUB200-C dataset. Considering the average accuracy across all sessions, our method achieves an average accuracy of 42.44%, surpassing the performance of the compared methods. This indicates that our approach exhibits superior performance and stability in handling generalized incremental learning under concept drift task. Specifically, iCaRL and ProtoNet initially demonstrate relatively high performance in the base session (session 0). However, due to the challenge of label scarcity, these methods fail to acquire information about new classes. Moreover, when faced with the covariate shift between the source and target streams, the model performance further deteriorates, resulting in the poorest performance of all the methods. In contrast, few-shot class incremental learning methods such as TOPIC, CEC, FACT, and TEEN effectively mitigate the label scarcity issue, leading to improved performance. However, these methods still struggle to overcome distribution shift problems. Our proposed method addresses these challenges effectively. It not only learns new classes with limited labeled data incrementally but also makes adaptations from the labeled source stream to the unlabeled target stream. Consequently, our method achieves the best performance.

The same conclusion can also be observed in Figure 6.5, which shows the overall accuracy on the CIFAR100-C and miniImagenet-C datasets. It provides a more intuitive comparison between our method and other baselines across all sessions. Our method consistently outperforms the others in the performance. In particular, as new sessions come, our method demonstrates superior performance in subsequent sessions. This further illustrates the superior performance of our method in handling class-incremental and distribution changes in dynamic environments.

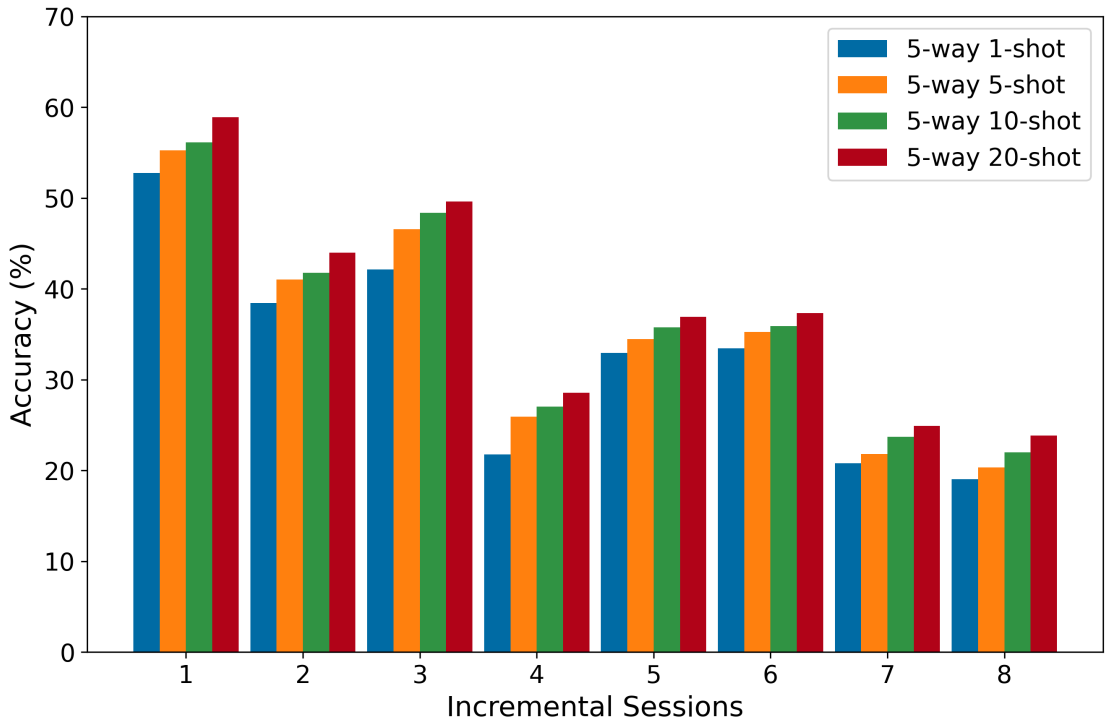
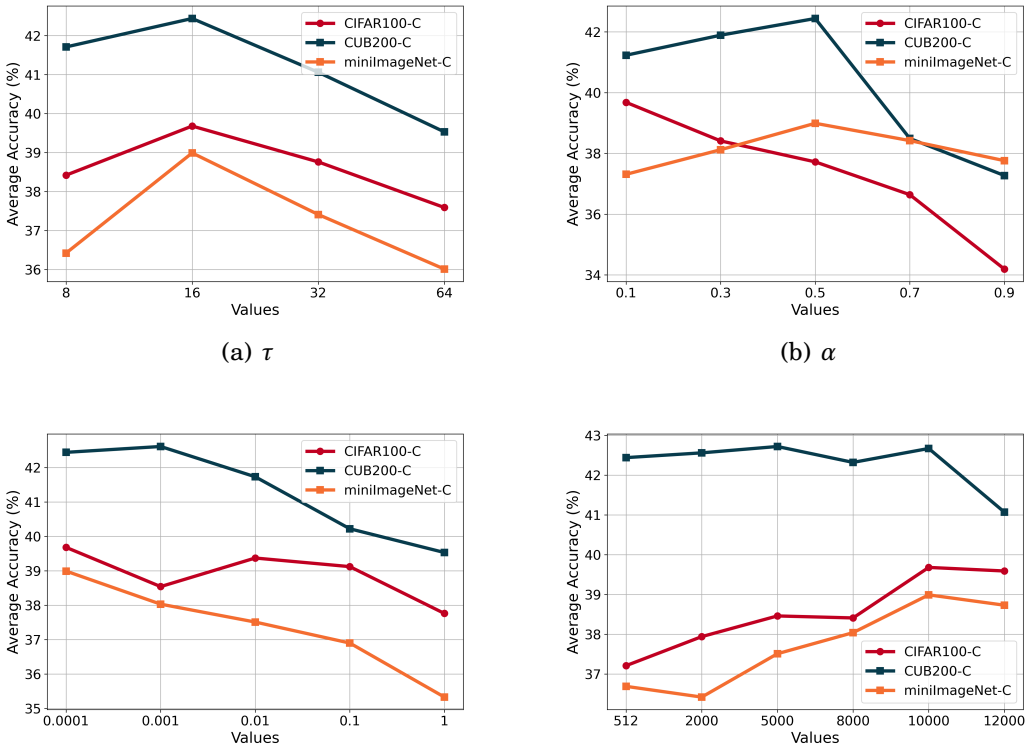


Figure 6.7: Influence of incremental shot.

Comparison with TTA/DG methods. To demonstrate the adaptation capability of the **R**eliable Surrogate **G**ap **S**harpness-aware **M**inimization (RGSM) algorithm, we conducted comparisons with the TTA and DG methods on the CIFAR100-C and CUB200-C datasets. To ensure fair comparisons, we kept the training-free calibrated prototypes strategy fixed for the incremental learning of new classes and only employed different methods during the adaptation stage.

As shown in Table 6.2, it is evident that our proposed method consistently outperforms the TTA/DG baselines across all incremental sessions on the CIFAR100-C and miniImageNet-C datasets. Of these, TENT performs the worst, with an average accuracy of 2.34% and 3.13% lower than our method on the CIFAR100-C and miniImageNet-C datasets, respectively. This is because TENT only minimizes the entropy, which can lead to overfitting issues during training and convergence towards sharp minima. Therefore, SAM minimizes the sharpness of the entropy, enhancing the model's generalization. SAR



(a) τ (b) α

(c) β (d) Batch size

Figure 6.8: The effect of different parameters on average classification accuracy.

further removes samples with high entropy to improve model generalization. However, the perturbed loss is not always sharpness-aware, so GSAM minimizes the surrogate gap and perturbed loss simultaneously. Our RGSM method comprehensively addresses all these issues. It not only minimizes the perturbation loss and the surrogate gap simultaneously but also integrates a reliable indicator function to filter out samples with high entropy. Therefore, it can achieve the best results, highlighting its robustness and superiority in handling incremental learning tasks under distribution shifts. The same results can also be verified in Table 6.3, which demonstrates its superior performance on the CUB200-C dataset.

6.3.5 Ablation study

To comprehensively evaluate the efficacy of each component in CSFA, we perform an ablation study on the CIFAR100-C dataset. Specifically, we devised three variations of CSFA to validate the rationale of each component and its impact on the overall classification results.

As depicted in Figure 6.6, CSFA$_{v1}$ serves as a baseline design that does not consider limited source samples and drift adaptation. This method solely employs incremental learning to fine-tune the model with a small number of new samples for predicting the target stream. Consequently, for the GILCD task, CSFA$_{v1}$ performs the poorest and exhibits significantly lower performance than CSFA. CSFA$_{v2}$ incorporates a prototype classifier without calibrations for few-shot class-incremental learning, leading to improved performance. However, due to the evident covariate shift between the source and target streams, its predictive results remain unsatisfactory. Furthermore, we introduce our proposed RGSM adaptation strategy into CSFA$_{v3}$, resulting in a significant improvement in predictive performance. Lastly, the complete CSFA further utilizes the well-learned base prototypes to calibrate the biased prototypes of novel classes during incremental sessions, achieving the best performance.

6.3.6 Further Analysis

Influence of incremental shot. In the GILCD setting, we assume that each incoming session follows an N-way K-shot setup. Therefore, the incremental learning performance depends on the number of new samples provided in each session, i.e., K. Thus, we vary the shot number to investigate its impact on the final accuracy. We keep the incremental way consistent with the benchmark setting and change the shot number K among {1, 5, 10, 20} on the CIFAR100-C dataset. As inferred from Figure 6.7, with more instances per class in the source stream, the model can learn more information about new classes.

Consequently, the estimation of prototypes becomes more precise, leading to improved performance.

Parameter Sensitivity. In the proposed CSFA method, there are four main parameters affecting the performance, i.e., the scaling hyperparameter τ, calibration coefficient α, ascent step size β, and the batch size during adaptation. To analyze their impact on the overall performance, we conduct experiments under various values of all parameters on these three datasets. Specifically, we search the optimal parameters by setting $\tau \in \{8, 16, 32, 64\}$, $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, $\beta \in \{0.0001, 0.001, 0.01, 0.1, 1\}$ and $batch\_size \in \{512, 2000, 5000, 8000, 10000, 12000\}$. During the experiment, each parameter is tuned while the others are kept fixed, and the results are shown in Figure 6.8.

During the class incremental learning, as our method does not require additional model updates after the base training, we need to determine the optimal performance by adjusting the scaling hyperparameter τ and calibration coefficient α. Observations from Figure 6.8(a) and Figure 6.8(b) reveal that various values of τ and α yield distinct outcomes. In our experiment, we set τ to 16 across all datasets, while assigning α values of 0.1 for CIFAR100-C and 0.5 for CUB200-C and miniImageNet-C datasets. During the adaptation process, the parameter β dictates the ascent step size. Figure 6.8(c) illustrates the varied effects of different settings of β on model adaptation. Therefore, we opt for $\beta = 0.0001$ for CIFAR100-C and miniImageNet-C datasets, and $\beta = 0.001$ for the CUB200-C dataset. Additionally, as depicted in Figure 6.8(d), the choice of batch size significantly influences the adaptation process. Smaller batch sizes may introduce randomness, thereby yielding unstable results. Through meticulous experimental analysis, we designate the adaptation batch sizes to be 512 for CUB200-C and 10000 for CIFAR100-C and miniImageNet-C to achieve optimal performance.

6.4 Summary

By considering the continuous evolution of classes and distributions in real-world applications, we introduce a novel Generalized Incremental Learning under Concept Drift (GILCD) setting. Furthermore, a novel Calibrated Source-Free Adaptation (CSFA) framework is proposed to effectively tackle all the challenges in GILCD by incrementally learning novel classes with limited labeled samples and adapting to distribution shifts in target streams. CSFA leverages training-free calibrated prototypes and a source-free adaptation strategy to achieve robust and adaptable incremental learning. By fusing biased prototypes with weighted base prototypes, CSFA mitigates the catastrophic forgetting problem and enhances prediction stability for new classes. Additionally, the source-free adaptation strategy RGSM minimizes the perturbation loss and the surrogate gap simultaneously but also integrates a reliable indicator function to filter out samples with high entropy. It ensures distribution alignment between source and target streams as well as improving the generalization capability for changing target distributions. Overall, this work advances the field of incremental learning and concept drift adaptation by providing practical solutions to the challenges posed by evolving classes and concept drift in real-world scenarios.

CHAPTER 7

OUTDOOR VISUAL RESTORATION UNDER CONCEPT DRIFT

RQ5 mentions that the issue of concept drift is prevalent in many real-world scenarios, significantly affecting the decision-making of intelligent systems. Therefore, in this chapter, we focus on the task of video content restoration within the context of intelligent visual systems. For instance, in auto-driving systems, different cameras capture video streams from various views, generating multiple video streams. In outdoor environments, the collection of videos often results in background drifts inevitably, which can significantly impact the restoration of video content. One of the most noticeable drifts is the transition from day to night, which serves as a clear instance of concept drift. To verify this hypothesis and achieve RO5, this chapter first designs a robust baseline for visual snow removal in a static environment, which is a critical research task in video content restoration. Subsequently, testing is conducted in dynamic environments with day-night transitions to validate the significant impact of such drifts on the model. Additionally, this chapter proposes a novel adaptive model to address the concept drift problem in

video content restoration tasks.

In this chapter, the detailed analysis of motivations and challenges is introduced in Section 7.1. In Section 7.2, a base model, Dual-Stream Temporal Transformer (DSTT), for snow removal, is introduced and tested in a static environment. And we investigate the influence caused by drifting scenarios. To overcome the concept drift problem, an Adaptive DSTT model is proposed in Section 7.3. Section 7.4 concludes this chapter.

7.1 Introduction

Outdoor vision systems often grapple with challenges posed by inclement weather conditions, such as rain, haze, and snow. These conditions can lead to serious reductions in visibility, which detrimentally affect decision-making in vision-based intelligent systems Patil et al. (2023); Zhang et al. (2022c); Liu et al. (2018). Notably, while much research has been conducted on removing rain or haze from images or videos Yang et al. (2022); Yin et al. (2020); Wang et al. (2024c), there remains a significant gap in the literature concerning the removal of snow Yang et al. (2020); Özdenizci and Legenstein (2023); Gui et al. (2021). The complexity of addressing snow is higher than that of rain or haze. This is primarily due to the opacity of snowflakes, which obscures the view of objects in the background. Additionally, snowflakes, unlike raindrops, exhibit a variety of shapes as they fall Zhang et al. (2021b). It is precisely because of the opacity and irregularity of snowflakes that it is challenging to fully restore the background content. Therefore, in this chapter, our focus is on removing snowflakes from the air to recover a clean high-quality video.

In real-world applications, one decision-making system may contain many cameras and capture various video streams from various angles, generating multiple video streams. Videos captured from different angles contain complementary information, thereby enhancing decision-making capabilities. For example, with binocular vision,

the same snowflake will obscure different objects depending on whether one is looking through the left or the right lens Poggi et al. (2021). As shown in Figure 7.1, the degraded region caused by one snowflake from the left view, indicated by the red blur, is different from the degraded region captured from the right view (the blue blur). It is difficult to fully solve this problem in a 3D scene with monocular desnowing methods. Yet, it has been demonstrated that exploiting information from different views can help to improve performance with many vision tasks, such as scene understanding Eslami et al. (2016), depth estimation Jeon et al. (2018), and even deraining Yamashita et al. (2005). Despite a few existing works on snow removal, the current efforts predominantly concentrate on a single-image scenario Jiang et al. (2021); Yuan et al. (2021). They usually leverage the spatial or semantic priors to learn a snow map or directly design an end-to-end deep convolutional neural network to remove snow. Although these methods have achieved some success with single-image desnowing, it is hard to adapt them to video desnowing directly because one needs to consider how to model and exploit the inherent temporal correlations between streaming video frames Song et al. (2021b); Zhang et al. (2022b). Therefore, it is vital to pay attention to developing a base model for removing snow by considering different video streams.

In addition, most of the few methods that deal with desnowing are dominated by convolutional neural networks (CNNs). These methods focus on developing more complicated CNN architectures by adding extensive connections Chen et al. (2020b), multi-scale operations Zhang et al. (2021b), and so on. Currently, Transformer has demonstrated great potential in computer vision tasks Xu et al. (2022). This shows a much stronger ability to explore the long-range interactions between different visual regions than CNNs Dosovitskiy et al. (2020); Liu et al. (2022). Furthermore, it also enhances the interactions between scene content and attention maps Liang et al. (2021). In recent years, Transformer has also proven to have great success with low-level vision tasks as

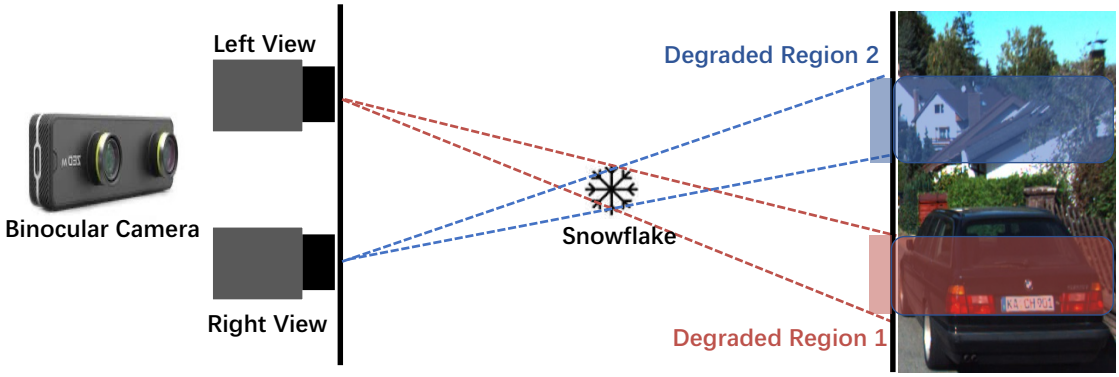


Figure 7.1: An illustration of how snowflakes obscure different parts of an image with binocular cameras. In an outdoor surveillance scene, one identical snowflake causes different occlusions captured from different views. From the left view, it causes the degraded Region 1 (the red blur) while, from the right view, the degradation occurs in Region 2 (the blue blur).

well, including denoising Wang et al. (2022e), super-resolution Zhang et al. (2022e), and deraining Zamir et al. (2022). However, it has not been investigated for its power with desnowing.

Therefore, we propose the Dual-Stream Temporal Transformer (DSTT) framework for binocular video desnowing that captures the spatial-temporal information in a video from different views. As shown in Figure 7.3, given the advantages of the convolutional operation in early visual information extraction Xiao et al. (2021); Yuan et al. (2023), a convolutional layer is first constructed to extract the low-level features. Additionally, the low-level feature learning improves the stability of the optimization process in the deep learning architecture Yu et al. (2022a); Wang et al. (2023c). Next, a Dual-Stream Weight-shared Transformer (DSWT) module captures the spatial information from the two different views. The DSWT is constructed based on the Residual Swin Transformer blocks (RSTBs) with hierarchical weight-shared connections, which exchange the shared information across different views from the low-level to the high-level layers. Concretely, RSTBs composed of several Swin Transformer layers are built along with a convolution layer and a shortcut connection between the input and output Liang et al. (2021).

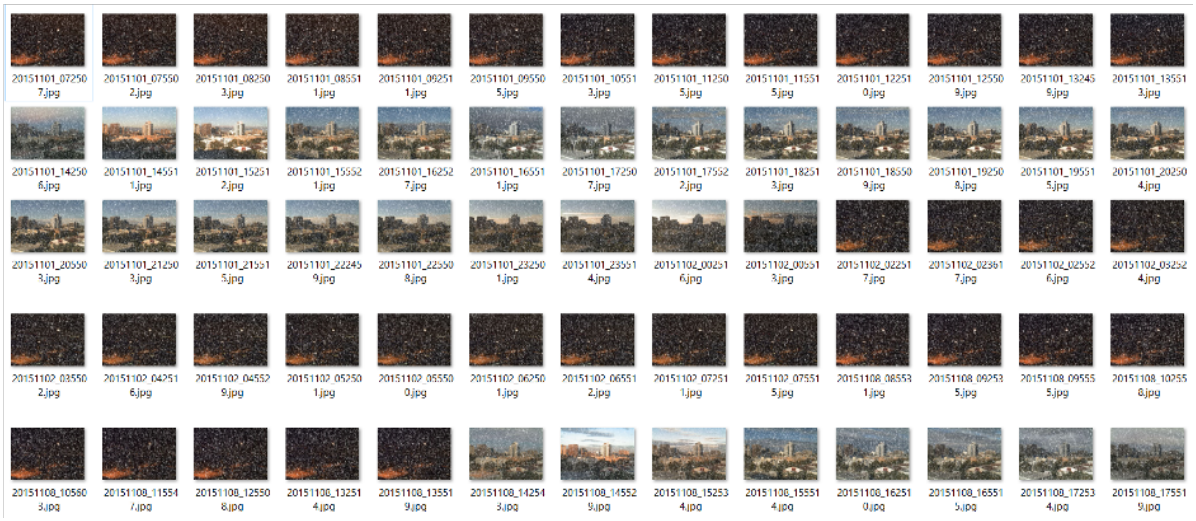


Figure 7.2: Example of drifting snowy scenes (DNIM Dataset).

Furthermore, we design a Dual-Stream ConvLSTM (DS-CLSTM) module to exploit the temporal correlations across streaming frames to help remove the snow. Unlike a traditional LSTM, DS-CLSTM addresses features from multiple views simultaneously. It also integrates a convolutional operation that can take inputs at various scales. Thus, we can make direct use of the spatial features captured from the previous module to generate temporal representations. This also enhances the interaction correlations between multiple views of the video streams. The last step is to recover clean videos from the final recovery module, which preserves as many details as possible from the original content.

Extensive experiments have demonstrated that our developed DSTT model exhibits superior performance in video content restoration under stable environments, where the background remains unchanged. However, in outdoor environments, the collection of videos often leads to inevitable background drifts as shown in Figure 7.2, significantly affecting the restoration of video content. One of the most noticeable drifts is the transition from day to night, which serves as a clear instance of concept drift. Testing DSTT in such background-drifting environments revealed that the model's performance is

severely impacted when drift occurs, greatly reducing its ability to restore video content. Therefore, for video content restoration tasks in environments with concept drift, we propose the AdaDSTT model framework. This model adopts DSTT as a baseline and incorporates an adaptation module designed to optimize the model using unsupervised contrastive learning. This approach aims to achieve stable performance of the model in drifting environments.

7.2 Base Model: Dual-Stream Temporal Transformer

Our ultimate goal is to remove falling snowflakes from videos so as to recover clear videos. As shown in Figure 7.3, DSTT consists of four modules: shallow feature extraction, spatial representation, temporal representation, and a recovery module.

Shallow Feature Extraction. A binocular camera generates two frame streams from the left and right view, *i.e.*, $\boldsymbol{X_L} \in \{x_{L1}, x_{L2}, ..., x_{Lt}\}$ and $\boldsymbol{X_R} \in \{x_{R1}, x_{R2}, ..., x_{Rt}\}$, where $x_{Lt}, x_{Rt} \in \mathbb{R}^{H \times W \times 3}$ are the t-th frame of $\boldsymbol{X_L}$ and $\boldsymbol{X_R}$, respectively. Due to the advantages of the convolutional layer in the early visual information extraction Xiao et al. (2021), the first step is to use a 3×3 convolutional layer H_{conv} to extract the shallow features:

$$\boldsymbol{F_L^0}, \boldsymbol{F_R^0} = H_{conv}(\boldsymbol{X_L}, \boldsymbol{X_R}), \tag{7.1}$$

where H_{conv} is the shallow feature extraction function, and $\boldsymbol{F_L^0}$, $\boldsymbol{F_R^0}$ are the features extracted from left-view and right-view videos, respectively. In this process, the input videos can be mapped to any dimensional feature space, offering flexibility for the next processing module. This mapping also improves the stability of the optimization process within the deep feature learning architecture.

Spatial Representation. Next, the DSWT module captures the high-level spatial representations. To make full use of both the low-level and high-level features, we also

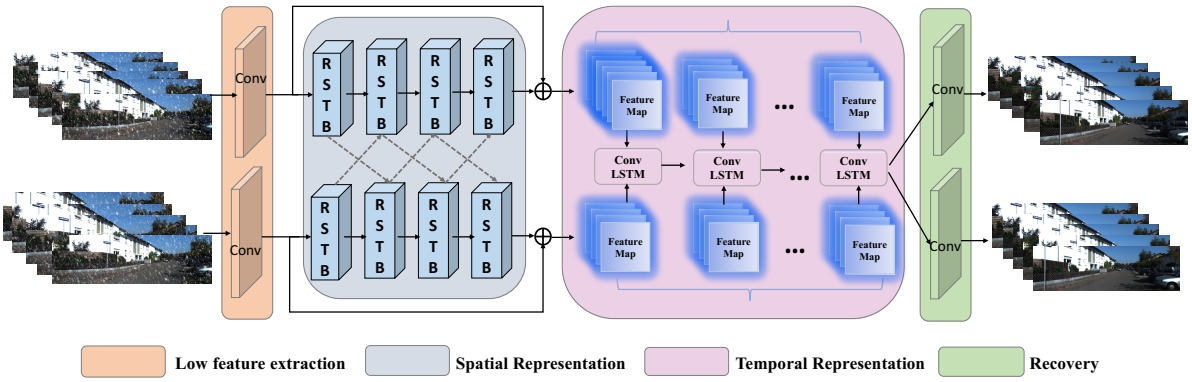


Figure 7.3: The DSTT framework. DSTT consists of four modules: shallow feature extraction, spatial representation, temporal representation and a final recovery module. The process begins by extracting low-level features via a convolutional layer. Then, the DSWT module captures the spatial information from the left and right views via information interaction in a layer-by-layer manner. Next, the DS-CLSTM module exploits the temporal correlations across streaming frames to help remove the snow. Finally, the clean videos are recovered via the final recovery module while preserving the detail of the original content.

adopt the residual operation to aggregate these features. This is formulated as,

$$\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{s}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{s}} = H_S\left(\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{0}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{0}}\right) + \left(\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{0}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{0}}\right) \tag{7.2}$$

where H_S denotes the spatial representation function, $\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{s}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{s}}$ are the final spatial representations of the left- and right-views, respectively.

Temporal Representation. The DS-CLSTM then exploits the correlations between streaming frames, which works together with the DSWT module to restore clean videos. This is formulated as,

$$\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{t}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{t}} = H_T\left(\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{s}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{s}}\right) \tag{7.3}$$

where H_T denotes the temporal representation function and $\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{t}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{t}}$ are the final temporal representations of the left- and right-views, respectively.

Recovery. Finally, the clean videos are restored by using the final features with the spatial-temporal information via a 3× 3 convolutional layer,

$$\boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{clean}},\boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{clean}} = H_{conv}\left(\boldsymbol{F}_{\boldsymbol{L}}^{\boldsymbol{t}},\boldsymbol{F}_{\boldsymbol{R}}^{\boldsymbol{t}}\right) \tag{7.4}$$

where H_{conv} represents the final convolutional operation and $\boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{clean}}$ and $\boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{clean}}$ are the final recovered videos.

Loss Function. Parameters of this model are optimized by minimizing the L_1 pixel loss,

$$\mathscr{L} = \left\| \boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{clean}} - \boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{GD}} \right\|_1 + \left\| \boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{clean}} - \boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{GD}} \right\|_1, \tag{7.5}$$

where $\boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{clean}}$ and $\boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{clean}}$ are the outputs of DSTT, and $\boldsymbol{X}_{\boldsymbol{L}}^{\boldsymbol{GD}}$ and $\boldsymbol{X}_{\boldsymbol{R}}^{\boldsymbol{GD}}$ are the corresponding ground-truth videos.

7.2.1 Dual-Stream Weights-shared Transformer

Swin Transformer Liu et al. (2021) integrates the advantages of both a CNN and the Transformer model. This configuration not only has the ability to capture long-range dependencies within the input but also extracts the local information via a convolution operation. Hierarchical feature maps are constructed with linear computational complexity in relation to the image size. This implementation can be thought of as a general-purpose backbone for computer vision tasks. Unlike standard multi-head self-attention (MSA) (i.e., global self-attention) Vaswani et al. (2017), the Swin Transformer applies a shifted window based on a local self-attention mechanism. As shown in Figure 7.4 (a), the Swin Transformer block contains a shifted window in an MSA layer and a 2-layer multi-layer perceptron (MLP) with GELU non-linearity. In addition, there are two LayerNorm (LN) layers before the MSA module and the MLP, along with a residual connection applied after each module. The overall operation process is formulated as,

$$\begin{aligned} \boldsymbol{X} &= MSA(LN(\boldsymbol{X}) + \boldsymbol{X}), \\ \boldsymbol{X} &= MLP(LN(\boldsymbol{X}) + \boldsymbol{X}). \end{aligned} \tag{7.6}$$

Here, the Swin Transformer first computes the self-attention within each local window. Assuming the size of the input is $H \times W \times 3$, it is first partitioned into non-overlapping $M \times M$ patches and reshaped to a $\frac{HW}{M^2} \times M^2 \times C$ scale. Each vector actually represents three different vectors: the *query* vector q, the *key* k and the *value* v of d dimensions. All the reshaped vectors derived from the patches are then packed into three matrices, *i.e.*, the *query* matrix Q, the *key* matrix K and the *value* matrix V. In this way, the scores for different input vectors can be computed by $S = QK^T$, and then normalized via $S_n = S/\sqrt{d}$ Vaswani et al. (2017). To maintain the relative position information, the learned positional encoding B is added to the normalized score to generate an all-around vision embedding $E = S_n + B$ Raffel et al. (2020); Han et al. (2022). This is then translated into probability form with a softmax function. The final weighted value matrix is produced from $Z = SoftMax(E)V$.

The process for calculating the unified self-attention is formulated as,

$$\text{Attention}(Q,K,V) = \text{SoftMax}\left(QK^T/\sqrt{d} + B\right)V, \tag{7.7}$$

where $Q,K,V \in \mathbb{R}^{M^2 \times d}$ are the *query*, *key* and *value* matrices, B is the learned relative positional encoding, d is the *query*/*key* dimension, and M^2 is the total number of patches.

However, this fixed window partitioning strategy lacks connections across local windows. Therefore, the partitioned window size is changed from $M \times M$ to $\left(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor\right)$ and a shifted window is generated. Finally, the shifted window and the regular window partitioning strategies are used in an alternating fashion to enhance the connection across local windows.

The first step in generating the spatial representations is to construct a residual Swin Transformer block (RSTB) following Liang et al. (2021). As shown in Figure 7.4 (b), the RSTB contains n Swin Transformer layers and a convolutional layer with a shortcut between the input and the output. Given the input features F_{input}, the overall process of

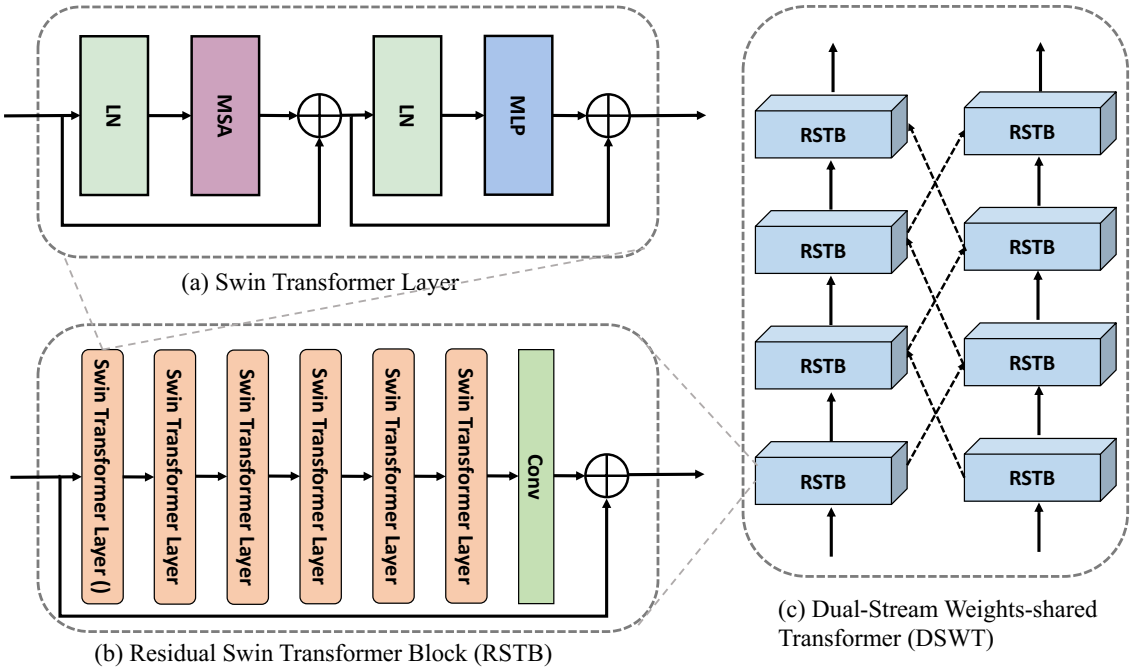


Figure 7.4: The illustration of Transformer. (a) shows the architecture of the standard Swin Transformer layer. (b) illustrates the Residual Swin Transformer Block, which consists of several Swin Transformer layers and a convolutional layer with a residual connection. Our proposed Multi-Stream Weight-shared Transformer (DSWT) is shown in (c). The output of each block from the left view is concatenated with the output from the right view, and the complementary information from different views is conducted from low-level to high-level layers.

the RSTB is formulated as,

$$F_i = H_{RSTB_i}\left(F_{input(i-1)}\right), \quad i = 1,2,\ldots,n, \tag{7.8}$$
$$F_{out} = H_{conv}(F_n) + F_{input},$$

where H_{RSTB_i} denotes the i-th Swin Transformer layers of RSTB and n is the total number of Swin Transformer layers.

To make full use of the information from different views, the DSWT module processes the features in a parallel manner. As shown in Figure 7.4 (C), given the feature volumes (F_L, F_R) extracted from the pre-processing modules, the parallel multi-stream Transformer exchanges the information across different views from the low-level to the high-level layers. The output of each RSTB from the left branch is combined with the

output of each RSTB from the right branch, and then the fused volumes are fed into the next layer. The whole process is formulated as,

$$\begin{aligned} F_{L(i)} &= H^{left}_{DSWS-RST_{(i)}}\left(F_{L(i-1)} \odot F_{R(i)}\right), \\ F_{R(i)} &= H^{right}_{DSWS-RST_{(i)}}\left(F_{R(i-1)} \odot F_{L(i)}\right), \\ & i = 1,2,\ldots,n, \end{aligned} \tag{7.9}$$

where $H^{left}_{DSWS-RST_{(i)}}$ and $H^{left}_{DSWS-RST_{(i)}}$ denote the i-th block of the DSWT from the left and right views, respectively. n denotes the total number of DSWT blocks, and $\odot$ is the concatenation operation.

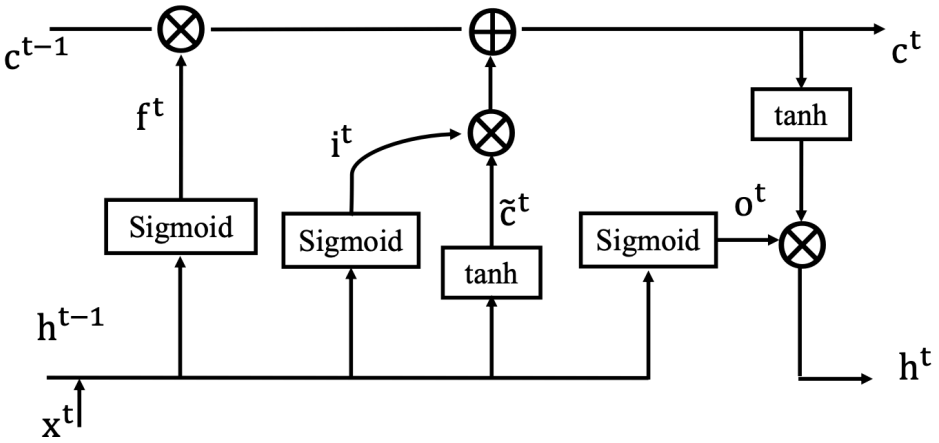
7.2.2 Dual-Stream ConvLSTM

With spatial representations of the input frames in hand, the DS-CLSTM next learns the temporal information. This module is based on an LSTM model Jiang et al. (2017). There are three components in a traditional LSTM: the forget gate, the input gate and the output gate. As shown in Figure 7.5 (a), the forget gate simultaneously considers the hidden state of the last time stamp $h^{(t-1)}$ and the current input data $x^{(t)}$ to determine the information that should be forgotten via a Sigmoid function. This process is formulated as,

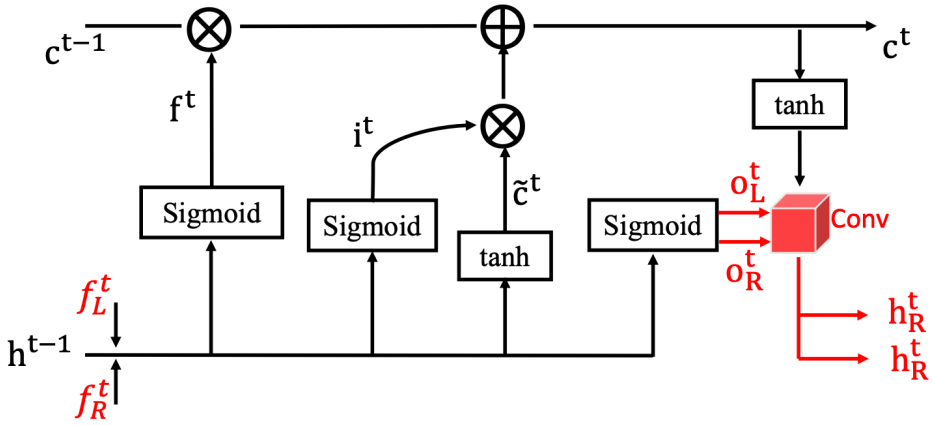
$$f^{(t)} = \sigma\left(W^{(f)}\left[h^{(t-1)}, x^{(t)}\right] + b^{(f)}\right), \tag{7.10}$$

Then, the input gate adds the current information to the hidden state. Here, the hidden state of the last time stamp $h^{(t-1)}$ and the current input data $x^{(t)}$ are fed into a Sigmoid and a Tanh function to get the current input state $i^{(t)}$ and the updated cell state $\widetilde{C}^{(t)}$. The current cell state can then be derived by fusing the forget and input states. This whole process is formulated as,

$$\begin{aligned} i^{(t)} &= \sigma\left(W^{(i)}\left[x^{(t)}, h^{(t-1)}\right] + b^{(i)}\right), \\ \widetilde{C}^{(t)} &= \tanh\left(W^{(C)}\left[x^{(t)}, h^{(t-1)}\right] + b^{(C)}\right), \\ C^{(t)} &= f^{(t)} \odot C^{(t-1)} + i^{(t)} \odot \widetilde{C}^{(t)}, \end{aligned} \tag{7.11}$$



(a) Traditional LSTM cell



(b) Dual-Stream ConvLSTM cell

Figure 7.5: Comparison of traditional LTSM and our proposed DS-CLSTM. The inputs ($f_L^{(t)}$ and $f_R^{(t)}$) are the spatial feature maps extracted from DSWT illustrated in Figure 7.4 (c). The red part is the convolutional layer which generates the temporal features.

The output gate integrates the updated input and the current cell state to generate the final output with the long-short memory,

$$\begin{aligned} o^{(t)} &= \sigma\left(W^{(o)}\left[x^{(t)}, h^{(t-1)}\right] + b^{(o)}\right), \\ h^{(t)} &= o^{(t)} \odot \tanh C^{(t)}, \end{aligned} \tag{7.12}$$

where $W^{(\cdot)}$ is the input-to-hidden matrix, and $b^{(\cdot)}$ is the bias vector. σ denotes the Sigmoid activation function, and $\odot$ is the point-wise multiplication.

To ensure traditional LSTMs are compatible with the spatial representations, the

Hadamard product in the LSTM is replaced with a convolutional operation. Additionally, the convolutional operation is used to calculate the hidden state following ConvLSTM Jiang et al. (2017). Plus, the traditional single-input architecture has been replaced with a dual-stream architecture by processing the left and right views simultaneously. These differences between a traditional LSTM and DS-CLSTM are illustrated in Figure 7.5 (b). Overall, the DS-CLSTM module is formulated as,

$$
\begin{aligned}
f^{(t)} &= \sigma\left(W^{(f)} * \left[f_L^{(t)}, f_R^{(t)}, h^{(t-1)}\right] + b^{(f)}\right), \\
i^{(t)} &= \sigma\left(W^{(i)} * \left[f_L^{(t)}, f_R^{(t)}, h^{(t-1)}\right] + b^{(i)}\right), \\
\widetilde{C}^{(t)} &= \tanh\left(W^{(C)} * \left[f_L^{(t)}, f_R^{(t)}, h^{(t-1)}\right] + b^{(C)}\right), \\
C^{(t)} &= f^{(t)} \odot C^{(t-1)} + i^{(t)} \odot \widetilde{C^{(t)}}, \\
o_L^{(t)} &= \sigma\left(W^{(o)} * \left[f_L^{(t)}, h^{(t-1)}\right] + b^{(o)}\right), \\
o_R^{(t)} &= \sigma\left(W^{(o)} * \left[f_R^{(t)}, h^{(t-1)}\right] + b^{(o)}\right), \\
h_L^{(t)}, h_R^{(t)} &= \mathrm{Conv}\left(\left[o_L^{(t)}, \tanh C^{(t)}\right], \left[o_R^{(t)}, \tanh C^{(t)}\right]\right),
\end{aligned}
\tag{7.13}
$$

where $*$ is the convolution operation, and $f_L^{(t)}$ and $f_R^{(t)}$ are the spatial feature maps extracted from frame t by the DSWT module. These two feature maps from left and right views are fed to update the states of the forget gate $f^{(t)}$, the input gate $i^{(t)}$, the memory cell $C^{(t)}$, and the dual-stream output gate $o_*^{(t)}$. Then the output and the updated memory cell are concatenated and input into the two convolutional layers, from which the final output results $h_*^{(t)}$ are derived. This approach enhances the temporal correlations across the streaming frames and the different views for final snow removal.

7.2.3 Experiment in Static Scenarios

To test the DSTT framework, we assembled two binocular snow removal datasets using Photoshop. The first, called **SnowKITTI2012**, is based on the KITTI 2012 dataset Geiger et al. (2013). KITTI 2012 dataset records the traffic scenarios using the binocular camera from a moving platform. It comprises 194 pairs of binocular video shots for the

training set while the testing set comprises 195 pairs of binocular video shots. Each video shot contains 21 frames with a resolution of 1226 × 370. The second dataset, **SnowKITTI2015**, is based on the stereo KITTI 2015 dataset Geiger et al. (2013). Both the training set and testing set contain 200 pairs of binocular video shots. Each video shot contains 21 frames with a resolution of 1241 × 376.

We compared the performance of DSTT against both single- and dual-stream models. DesnowNet Liu et al. (2018), DDMSNet Zhang et al. (2021b) and SwinIR Liang et al. (2021) served as our single-stream comparators. Plus, we created an ablation of DSTT, called Single-Stream Temporal Transformer (SSTT), which considers the spatial-temporal correlations across the streaming frames. EPRRNet Zhang et al. (2022c) for rain removal served as a baseline for the binocular video desnowing task. And, again, we prepared an ablation, being the multi-stream weight-shared Transformer (DSWT) module on its own. More details for each of the baselines are provided below. To ensure the comparison was fair, we used the open-source codes of all models, ensuring all followed the same configuration, and retrained each network on our synthetic snow datasets. The results are reported in terms of PSNR and SSIM. This means that, unlike the original version of SwinIR, our implementation had 4-layer RSTB blocks to keep the same configuration as our proposed model.

7.3 Adaptation Model: Adaptive Dual-Stream Temporal Transformer

Since we have obtained a reliable snow removal model DSTT, we conducted tests on this model in drifting environments (i.e., environments with alternating day and night conditions). Specifically, we selected a video stream from the DNIM dataset as shown in Figure 7.2. Then, we evaluated the performance of the model pretrained on the

Table 7.1: Quantitative evaluation of DSTT on the SnowKITTI2012 dataset. The best results are indicated in bold.

	Methods	Left view		Right view		Average	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Single	DesnowNet	34.21	0.9711	34.14	0.9707	34.18	0.9709
	DDMSnet	35.59	0.9731	36.17	0.9734	35.88	0.9724
	SwinIR	39.98	0.9814	40.72	0.9886	40.35	0.9850
Dual	EPRRnet	39.71	0.9897	41.06	0.9911	40.39	0.9904
	Ours	**42.41**	**0.9934**	**43.28**	**0.9935**	**42.85**	**0.9935**

Table 7.2: Quantitative evaluation of DSTT on the SnowKITTI2015 dataset. The best results are indicated in bold.

	Methods	Left view		Right view		Average	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Single	DesnowNet	34.06	0.9708	33.55	0.9689	33.81	0.9698
	DDMSnet	35.63	0.9720	36.70	0.9761	36.17	0.9741
	SwinIR	39.71	0.9897	41.06	0.9912	40.39	0.9905
Dual	EPRRnet	40.32	0.9904	41.50	0.9913	40.91	0.9908
	Ours	**41.70**	**0.9919**	**42.14**	**0.9921**	**41.92**	**0.9920**

SnowyKITTI2012 dataset, the results of which are illustrated in Figure 7.6. It is evident that the model's performance is significantly affected when the background of the video drifts. This underscores the impact of concept drift on the performance of models in low-level visual tasks. Therefore, there is a pressing need to develop new models capable of detecting and adapting to novel dynamic scenes.

Figure 7.6: Online SSIM of DSTT on the DNIM dataset.

7.3.1 Drift Detection

Due to the difficulty in obtaining ground truth in real-world scenarios, we also employ an unsupervised detection algorithm based on Gaussian Mixture Models (GMM) to detect changes in data distribution, thus achieving drift detection. Similar to the methods described in Chapters 3 and 4, we utilize the GMM to evaluate the distributions of the old and new distributions.

In order to detect the drift in the target stream without utilizing labels, we use the stored data X in SnowyKITTI2012, which only contains the videos captured during day time to initialize a GMM model and deploy two sliding windows to detect the changes over time. Specifically, we design two sliding windows, i.e., Reference Window $W_{ref} = \{x_1, \cdots, x_n\}$ and Detect Window $W_{det} = \{x_{n+1}, \cdots, x_{2n}\}$, where n is the instance number within the window. Then, the average conditional probability of the reference window can be calculated by a point estimation of the mean for the normal distribution,

$$\mu_{\text{ref}} = \frac{1}{n} \sum_{t=1}^{n} \max_{k \in \{1,2,\ldots,K\}} P\left(T(x) \mid C_k\right). \tag{7.14}$$

The confidence interval estimation of the μ_{ref} is known to be $\left[\mu_{\text{ref}} - z_\alpha(\sigma/\sqrt{n}), \mu_{\text{ref}} + z_\alpha(\sigma/\sqrt{n})\right]$, where σ is the standard deviation and z_α is the significance level which is set as 3 Kim and Park (2017). The decision is made that the change has occurred when the point

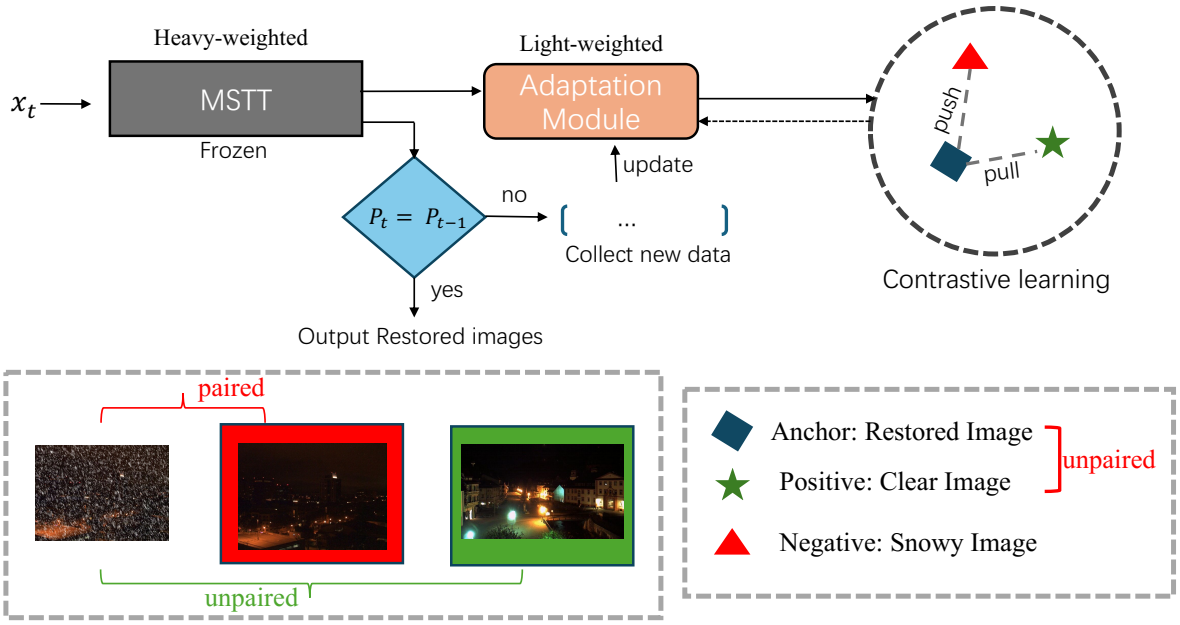


Figure 7.7: The AdaDSTT framework.

estimation by the mean μ_{ref} in the detection window satisfies $\mu_{\text{det}} \geq \mu_{\text{ref}} + z_\alpha \times \sigma/\sqrt{n}$. Otherwise, W_{ref} and W_{det} move step by step to receive new incoming data.

7.3.2 Drift Adaptation

Due to the large number of parameters in the DSTT model, updating the entire DSTT model incurs significant computational and time costs when drift occurs. To achieve rapid adaptation, as illustrated in Figure 7.7, we have designed a light-weighted adaptation module, consisting of two convolutional layers. When no drift is detected, the model directly outputs the restored image. Once drift is detected, the reference window is cleared entirely, and new data is collected to update the adaptation module, enabling it to cope with the newly arrived drifting data.

Given the challenge of lacking real-time ground truth for newly arrived data, we employ contrastive learning to train the adaptation module in an unsupervised manner Chen et al. (2020a); He et al. (2020b). This approach involves learning embeddings that

bring positive samples closer together while pushing apart embeddings of negative samples. By doing so, the technique aims to enhance the model's discriminative capabilities between different samples, thereby improving its performance in adapting to dynamic environments.

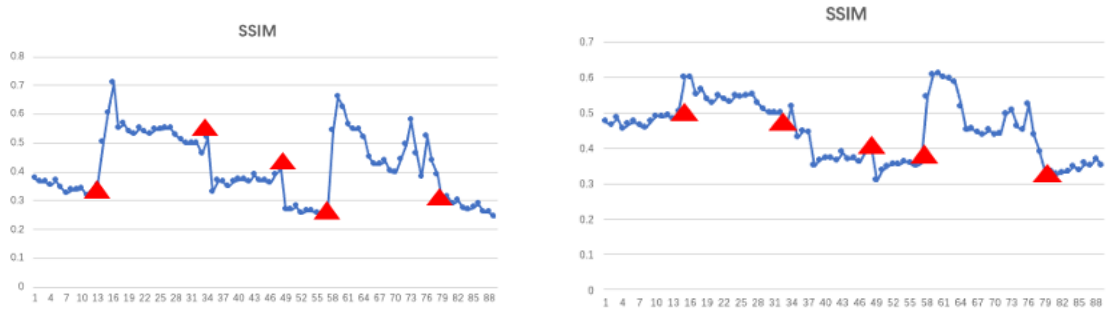
There are two challenges to construct the contrastive regularization, one is to construct the positive pairs and negative pairs, and the other one is to find the latent feature space of these pairs for contrast. In this method, we denote the group of a real-world clean image (R_c) and a preliminary restored image ($G(x)$) as the positive pair. Similarly, the negative pair is constructed by a group of real-world snowy image R_s and the restored image $G(x)$. Note that all these positive and negative samples are randomly chosen from real-world scenarios and are unpaired from input data. In addition, we need to find a latent feature space of these pairs for contrast. Here, we employ a pre-trained VGG-16 network Simonyan and Zisserman (2014) to extract the feature maps of different samples. Therefore, the pixel-wise contrastive loss can be expressed as:

$$Loss = \sum_{i=1}^{L} \omega_i \cdot \frac{\left\| \psi_i(R_c) - \psi_i(G(x)) \right\|_1}{\left\| \psi_i(R_s) - \psi_i(G(x)) \right\|_1} \tag{7.15}$$

where $\psi_i(.), i = 1, 2, \cdots L$ extracts the i-th hidden features by the pretrained VGG-16 model. Here we choose the 2-nd, 3-rd, and 5-th max-pooling layers. ω_i are the weight coefficients, and they are set to 0.4, 0.6, and 1 Wang et al. (2024d).

7.3.3 Experiment in Dynamic Scenarios

To validate our approach, we processed the DNIM dataset using Photoshop to simulate snowy conditions, resulting in the creation of the SnowyDNIM dataset, which exhibits day-night alternations with snow. The original DNIM dataset comprises 17 image sequences totaling 1722 RGB color images. These images are categorized into day and night sequences and are captured at various times of the day using fixed webcams. As illustrated in Figure 7.8, we compare the online SSIM metric. Initially, for DSTT (pre-



(a) DSTT (b) AdaDSTT

Figure 7.8: Online SSIM of DSTT and AdaDSTT on the SnowyDNIM dataset.

trained under daytime conditions), the model's performance is significantly affected when the video background undergoes drift. In contrast, our proposed AdaDSTT demonstrates better stability in response to such drift. Furthermore, the results for PSNR and SSIM are presented in Table 7.3, where AdaDSTT outperforms DSTT on the SnowyDNIM dataset, with an average increase of approximately 2 for PSNR and 0.04 for SSIM.

Table 7.3: Quantitative evaluation of DSTT and AdaDSTT on the DNIM dataset.

Model	PSNR	SSIM
DSTT	13.27	0.4114
AdaDSTT	15.39	0.4507

7.4 Summary

In this chapter, we presented a basic framework called Multi-Stream Temporal Transformer (DSTT) for removing snow from outdoor surveillance videos by considering the shared information from multiple views. DSTT comprises a multi-stream weight-shared Transformer (DSWT) module that captures shared spatial information from the left and the right views of binocular videos. High-level spatial features are then fused with the low-level information and fed into a multi-stream convLSTM (DS-CLSTM) module,

which exploits the temporal correlations across streaming frames. In this way, the DSTT model extracts better spatial-temporal features across the different views to help remove falling snow. Our experimental results verify that DSTT significantly outperforms other state-of-the-art desnowing models both quantitatively and qualitatively.

In the process of developing DSTT, we identified two promising research directions for future study. First, this research focuses on removing snow in static environments, yet, in real-world applications, environments are generally dynamic. For example, the background may change from day to night. Hence, learning to desnow in a dynamic environment would be a significant and interesting research direction. Additionally, from this research, we found that shared information from multiple views substantially helps in the process of removing snow. However, in this chapter, we only considered two views, i.e., the left and the right. Hence, constructing datasets made up of multiple video streams from different views and developing multi-stream models is a demand we may look to fulfill in the future.

CHAPTER 8

CONCLUSION AND FUTURE RESEARCH

8.1 Conclusion

In this thesis, we have delved into the challenges posed by concept drift in the context of multiple data streams, achieving two primary tasks: developing novel algorithms and theories for multistream classification and exploring concept drift in complex intelligent applications.

For multistream classification, we identified four newly exposed challenges, i.e., 1) how to model the dependency among data streams and exploit the positive information to help decision-making; 2) how to exploit guaranteed joint representations of different streams; 3) how to develop guaranteed and efficient adaptation methods to deal with real-world high-dimensional data; 4) how to handle the class-changing problem with the data streams evolving. To tackle these challenges, we devised four concept drift adaptation methods:

- Online Boosting Adaptive Learning (OBAL): This method aligns covariate shift and addresses dynamic correlation issues between source and target streams,

enhancing positive knowledge transfer while preventing negative transfer effects.

- Fuzzy Shared Representation Learning (FSRL): FSRL learns robust and compact common features for different streams, mitigating the impact of redundant features and uncertainties while addressing the drifting problem.

- Learn-to-Adapt Framework (L2A): L2A leverages efficient deep neural networks to handle complex and high-dimensional data streams effectively while adapting to changing distributions via a flexible meta-learning strategy.

- Calibrated Source-Free Adaptation (CSFA): This method employs a calibrated prototype classifier to overcome class incremental problems and adapt distribution shifts via source-free adaptation strategies over time.

Moreover, we extended our investigation to address concept drift issues in real-world applications, such as autonomous navigation in dynamically changing environments. Hence, 5) expanding the real-world applications of the concept drift problem and designing autonomous learning algorithms to address it have also become imperative demands. This led to the development of a transformer-based model (Adaptive Dual-Stream Temporal Transformer, i.e., AdaDSTT) capable of adapting video restoration capabilities in dynamic environments, considering factors like adverse weather conditions and day-night transitions.

In conclusion, our research offers a comprehensive set of drift adaptation methods to address challenges in multiple data streams while exploring concept drift problems in complex real-world scenarios. Therefore, we contribute to the advancement of adaptive learning under concept drift in the context of multiple data streams, paving the way for more robust and efficient intelligent systems in dynamic environments.

8.2 Future Research

This thesis identifies the following directions as future work:

- **Detection for distribution and class changes.** While our current research endeavors to address class incremental and distribution-changing problems, it relies on certain assumptions, such as the known distribution shifts between each session or batch. However, these assumptions are relatively strong and may not always hold in real-world scenarios. Thus, future work will focus on enhancing detection mechanisms to automatically identify distribution and class changes. This automated approach will ensure greater adaptability to evolving data without relying on pre-defined assumptions, thereby improving the robustness of the learning process.

- **Exploring label distribution shifts.** While our research primarily investigates changes in data distribution, we acknowledge the importance of considering label distribution shifts, particularly concerning issues like the long-tail problem. Future studies will delve deeper into understanding and addressing these label distribution shifts, thereby enhancing the overall effectiveness of our adaptive learning approach.

- **Enhancing flexibility and transferability.** In this study, we addressed the classification problem of multiple data streams and real-world visual restoration tasks. It is evident that for various tasks in real-world scenarios, different models are still required to handle concept drift issues. With the advancement of large language models (LLMs) containing numerous parameters and extensive knowledge, they exhibit promising knowledge transfer capabilities across different downstream tasks. Therefore, there is a need to establish a more generalized large model to

enhance its flexibility and transferability to deal with the concept drift problem across diverse tasks.

- **Improving reliability and trustworthiness.** Future work will focus on investigating the effects caused by noisy data or noisy labels. This involves categorizing various types of noise present in the dataset and exploring their sources and characteristics. Advanced data analysis techniques and algorithms will be employed to identify and understand the impact of noise data or labels. After identifying the noise, we will study different noise correction strategies, including data cleaning, noise filtering, and model robustness enhancement methods, to mitigate the impact of noise on the final results. we aim to delve into the effects of noisy data or labels on data analysis and machine learning and propose effective strategies to support the improvement of model performance in both theory and practice.

BIBLIOGRAPHY

Abadifard, S., Bakhshi, S., Gheibuni, S. & Can, F., 2023, 'Dyned: Dynamic ensemble diversification in data stream classification', *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 3707–3711.

Alippi, C., Boracchi, G. & Roveri, M., 2016, 'Hierarchical change-detection tests', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 2, pp. 246–258.

Alippi, C. & Roveri, M., 2008a, 'Just-in-time adaptive classifiers‚Äîpart i: Detecting nonstationary changes', *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1145–1153.

Alippi, C. & Roveri, M., 2008b, 'Just-in-time adaptive classifiers‚Äîpart ii: Designing the classifier', *IEEE Transactions on Neural Networks*, vol. 19, no. 12, pp. 2053–2064.

Bach, S. H. & Maloof, M. A., 2008, 'Paired learners for concept drift', *2008 Eighth IEEE International Conference on Data Mining*, IEEE, pp. 23–32.

Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R. & Morales-Bueno, R., 2006, 'Early drift detection method', *Fourth International Workshop on Knowledge Discovery from Data Streams*, , vol. 6pp. 77–86.

Bai, G., Ling, C. & Zhao, L., 2023, 'Temporal domain generalization with drift-aware dynamic neural networks', *arXiv e-prints*, pp. arXiv–2205.

Bai, J., Yuan, A., Xiao, Z., Zhou, H., Wang, D., Jiang, H. & Jiao, L., 2020, 'Class incremental learning with few-shots based on linear programming for hyperspectral image classification', *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 5474–5485.

Bifet, A. & Gavalda, R., 2007, 'Learning from time-changing data with adaptive windowing', *Proceedings of the 2007 SIAM International Conference on Data Mining*, SIAM, pp. 443–448.

Bifet, A., Holmes, G. & Pfahringer, B., 2010, 'Leveraging bagging for evolving data streams', *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 135–150.

Bitarafan, A., Baghshah, M. S. & Gheisari, M., 2016, 'Incremental evolving domain adaptation', *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2128–2141.

Bossu, J., Hautiere, N. & Tarel, J.-P., 2011, 'Rain or snow detection in image sequences through use of a histogram of orientation of streaks', *International journal of computer vision*, vol. 93, no. 3, pp. 348–367.

Caccia, M., Rodriguez, P., Ostapenko, O., Normandin, F., Lin, M., Page-Caccia, L., Laradji, I. H., Rish, I., Lacoste, A., Vázquez, D. et al., 2020, 'Online fast adaptation and knowledge accumulation (osaka): a new approach to continual learning', *Advances in Neural Information Processing Systems*, vol. 33.

Cai, J.-F., Candès, E. J. & Shen, Z., 2010, 'A singular value thresholding algorithm for matrix completion', *SIAM Journal on optimization*, vol. 20, no. 4, pp. 1956–1982.

Cao, J., Li, Y., Zhang, K. & Van Gool, L., 2021, 'Video super-resolution transformer', *arXiv preprint arXiv:2106.06847*.

Celik, B. & Vanschoren, J., 2021, 'Adaptation strategies for automated machine learning on evolving data', *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chandra, S., Haque, A., Khan, L. & Aggarwal, C., 2016, 'An adaptive framework for multistream classification', *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 1181–1190.

Chen, D., Wang, D., Darrell, T. & Ebrahimi, S., 2022, 'Contrastive test-time adaptation', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305.

Chen, D.-Y., Chen, C.-C. & Kang, L.-W., 2014, 'Visual depth guided color image rain streaks removal using sparse coding', *IEEE transactions on circuits and systems for video technology*, vol. 24, no. 8, pp. 1430–1455.

Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., Xu, C. & Gao, W., 2021, 'Pre-trained image processing transformer', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12299–12310.

Chen, J., Tan, C.-H., Hou, J., Chau, L.-P. & Li, H., 2018, 'Robust video content alignment and compensation for rain removal in a cnn framework', *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6286–6295.

Chen, T., Kornblith, S., Norouzi, M. & Hinton, G., 2020a, 'A simple framework for contrastive learning of visual representations', *International conference on machine learning*, PMLR, pp. 1597–1607.

Chen, W.-T., Fang, H.-Y., Ding, J.-J., Tsai, C.-C. & Kuo, S.-Y., 2020b, 'Jstasr: Joint size and transparency-aware snow removal algorithm based on modified partial convolution and veiling effect removal', *European Conference on Computer Vision*, Springer, pp. 754–770.

Chu, F. & Zaniolo, C., 2004, 'Fast and light boosting for adaptive mining of data streams', *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, pp. 282–292.

Chu, X., Tian, Z., Zhang, B., Wang, X., Wei, X., Xia, H. & Shen, C., 2021, 'Conditional positional encodings for vision transformers', *arXiv preprint arXiv:2102.10882*.

Dalal, N. & Triggs, B., 2005, 'Histograms of oriented gradients for human detection', *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, , vol. 1Ieee, pp. 886–893.

Dasu, T., Krishnan, S., Venkatasubramanian, S. & Yi, K., 2006, 'An information-theoretic approach to detecting changes in multi-dimensional data streams', *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*, Citeseer.

Denevi, G., Stamos, D., Ciliberto, C. & Pontil, M., 2019, 'Online-within-online meta-learning', *Advance in Neural Information Processing Systems (NeurIPS 2019)*, , vol. 32pp. 1–11.

Deng, Y., Ren, Z., Kong, Y., Bao, F. & Dai, Q., 2017, 'A Hierarchical Fused Fuzzy Deep Neural Network for Data Classification', *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 4, pp. 1006–1012.

Dietterich, T. G. et al., 2002, 'Ensemble learning', *The handbook of Brain Theory and Neural Networks*, vol. 2, pp. 110–125.

Ditzler, G. & Polikar, R., 2012, 'Incremental learning of concept drift from streaming imbalanced data', *IEEE transactions on knowledge and data engineering*, vol. 25, no. 10, pp. 2283–2301.

Domingos, P. & Hulten, G., 2000, 'Mining high-speed data streams', *Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 71–80.

Dong, B., Gao, Y., Chandra, S. & Khan, L., 2019, 'Multistream classification with relative density ratio estimation', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 33pp. 3478–3485.

Dong, F., Lu, J., Song, Y., Liu, F. & Zhang, G., 2021, 'A drift region-based data sample filtering method', *IEEE Transactions on Cybernetics*, pp. 1–14.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. et al., 2020, 'An image is worth 16x16 words: Transformers for image recognition at scale', *arXiv preprint arXiv:2010.11929*.

Du, H., Minku, L. L. & Zhou, H., 2019, 'Multi-source transfer learning for non-stationary environments', *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Du, L., Song, Q., Zhu, L. & Zhu, X., 2015, 'A selective detector ensemble for concept drift detection', *The Computer Journal*, vol. 58, no. 3, pp. 457–471.

Eddine Marouf, I., Roy, S., Tartaglione, E. & Lathuilière, S., 2023, 'Rethinking class-incremental learning in the era of large pre-trained models via test-time adaptation', *arXiv e-prints*, pp. arXiv–2310.

Elwell, R. & Polikar, R., 2011, 'Incremental learning of concept drift in nonstationary environments', *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531.

Eslami, S., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Hinton, G. E. et al., 2016, 'Attend, infer, repeat: Fast scene understanding with generative models', *Advances in Neural Information Processing Systems*, vol. 29.

Finn, C., Abbeel, P. & Levine, S., 2017, 'Model-agnostic meta-learning for fast adaptation of deep networks', *International Conference on Machine Learning*, PMLR, pp. 1126–1135.

Foret, P., Kleiner, A., Mobahi, H. & Neyshabur, B., 2021, 'Sharpness-aware minimization for efficiently improving generalization', *International Conference on Learning Representations*, .

Frias-Blanco, I., del Campo-Ávila, J., Ramos-Jimenez, G., Morales-Bueno, R., Ortiz-Diaz, A. & Caballero-Mota, Y., 2014, 'Online and non-parametric drift detection methods based on hoeffding‚Äôs bounds', *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 810–823.

Fu, X., Huang, J., Ding, X., Liao, Y. & Paisley, J., 2017, 'Clearing the skies: A deep network architecture for single-image rain removal', *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2944–2956.

Gama, J. & Castillo, G., 2006, 'Learning with local drift detection', *International conference on advanced data mining and applications*, Springer, pp. 42–55.

Gama, J., Medas, P., Castillo, G. & Rodrigues, P., 2004, 'Learning with drift detection', *Brazilian Symposium on Artificial Intelligence*, Springer, pp. 286–295.

Gama, J., Rocha, R. & Medas, P., 2003, 'Accurate decision trees for mining high-speed data streams', *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 523–528.

Gao, J., Fan, W., Han, J. & Yu, P. S., 2007, 'A general framework for mining concept-drifting data streams with skewed distributions', *Proceedings of the 2007 Siam International Conference on Data Mining*, SIAM, pp. 3–14.

Gao, X., Wang, Y., Cheng, J., Xu, M. & Wang, M., 2021, 'Meta-learning based relation and representation learning networks for single-image deraining', *Pattern Recognition*, vol. 120, p. 108124.

Geiger, A., Lenz, P., Stiller, C. & Urtasun, R., 2013, 'Vision meets robotics: The kitti dataset', *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237.

Geiger, A., Lenz, P. & Urtasun, R., 2012, 'Are we ready for autonomous driving? the kitti vision benchmark suite', *2012 IEEE conference on computer vision and pattern recognition*, IEEE, pp. 3354–3361.

Gidaris, S., Singh, P. & Komodakis, N., 2018, 'Unsupervised representation learning by predicting image rotations', *arXiv preprint arXiv:1803.07728*.

Glorot, X., Bordes, A. & Bengio, Y., 2011, 'Deep sparse rectifier neural networks', *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, pp. 315–323.

Gomes, H. M., Barddal, J. P., Enembreck, F. & Bifet, A., 2017a, 'A survey on ensemble learning for data stream classification', *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–36.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G. & Abdessalem, T., 2017b, 'Adaptive random forests for evolving data stream classification', *Machine Learning*, vol. 106, no. 9, pp. 1469–1495.

Gu, Y., Yang, X., Wei, K. & Deng, C., 2022, 'Not just selection, but exploration: Online class-incremental continual learning via dual view consistency', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7442–7451.

Gui, J., Cong, X., Cao, Y., Ren, W., Zhang, J., Zhang, J. & Tao, D., 2021, 'A comprehensive survey on image dehazing based on deep learning', *arXiv preprint arXiv:2106.03323*.

Halstead, B., Koh, Y. S., Riddle, P., Pears, R., Pechenizkiy, M., Bifet, A., Olivares, G. & Coulson, G., 2021, 'Analyzing and repairing concept drift adaptation in data stream classification', *Machine Learning*, pp. 1–35.

Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y. et al., 2022, 'A survey on vision transformer', *IEEE transactions on pattern analysis and machine intelligence*.

Haque, A., Tao, H., Chandra, S., Liu, J. & Khan, L., 2018, 'A framework for multistream regression with direct density ratio estimation', *Proceedings of the AAAI Conference on Artificial Intelligence*, .

Haque, A., Wang, Z., Chandra, S., Dong, B., Khan, L. & Hamlen, K. W., 2017, 'Fusion: An online method for multistream classification', *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 919–928.

He, J., Mao, R., Shao, Z. & Zhu, F., 2020a, 'Incremental learning in online scenario', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13926–13935.

He, K., Fan, H., Wu, Y., Xie, S. & Girshick, R., 2020b, 'Momentum contrast for unsupervised visual representation learning', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738.

Hendrycks, D. & Dietterich, T., 2018, 'Benchmarking neural network robustness to common corruptions and perturbations', *International Conference on Learning Representations*, .

Hinder, F., Vaquet, V. & Hammer, B., 2023, 'One or two things we know about concept drift–a survey on monitoring evolving environments', *arXiv preprint arXiv:2310.15826*.

Hoffman, J., Darrell, T. & Saenko, K., 2014, 'Continuous manifold based adaptation for evolving visual domains', *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 867–874.

Hospedales, T., Antoniou, A., Micaelli, P. & Storkey, A., 2020, 'Meta-learning in neural networks: A survey', *arXiv preprint arXiv:2004.05439*.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B. & Smola, A., 2006, 'Correcting sample selection bias by unlabeled data', *Advances in Neural Information Processing Systems*, vol. 19, pp. 601–608.

Huang, Z. & Zhang, J., 2022, 'Contrastive unfolding deraining network', *IEEE Transactions on Neural Networks and Learning Systems*.

Hulten, G., Spencer, L. & Domingos, P., 2001, 'Mining time-changing data streams', *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 97–106.

Isele, D. & Cosgun, A., 2018, 'Selective experience replay for lifelong learning', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 32.

Jeon, D. S., Baek, S.-H., Choi, I. & Kim, M. H., 2018, 'Enhancing the spatial resolution of stereo images using a parallax prior', *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1721–1730.

Ji, R., Cao, D., Zhou, Y. & Chen, F., 2016, 'Survey of visual sentiment prediction for social media analysis', *Frontiers of Computer Science*, vol. 10, no. 4, pp. 602–611.

Jiang, K., Wang, Z., Yi, P., Chen, C., Wang, G., Han, Z., Jiang, J. & Xiong, Z., 2021, 'Multi-scale hybrid fusion network for single image deraining', *IEEE Transactions on Neural Networks and Learning Systems*.

Jiang, L., Xu, M. & Wang, Z., 2017, 'Predicting video saliency with object-to-motion cnn and two-layer convolutional lstm', *arXiv preprint arXiv:1709.06316*.

Jiao, B., Guo, Y., Gong, D. & Chen, Q., 2022a, 'Dynamic ensemble selection for imbalanced data streams with concept drift', *IEEE Transactions on Neural Networks and Learning Systems*.

Jiao, B., Guo, Y., Yang, S., Pu, J. & Gong, D., 2022b, 'Reduced-space multistream classification based on multi-objective evolutionary optimization', *IEEE Transactions on Evolutionary Computation*.

Kang, L.-W., Lin, C.-W. & Fu, Y.-H., 2011, 'Automatic single-image-based rain streaks removal via image decomposition', *IEEE transactions on image processing*, vol. 21, no. 4, pp. 1742–1755.

Kang, M., Park, J. & Han, B., 2022, 'Class-incremental learning by knowledge distillation with adaptive feature consolidation', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16071–16080.

Kifer, D., Ben-David, S. & Gehrke, J., 2004, 'Detecting change in data streams', *VLDB*, , vol. 4Toronto, Canada, pp. 180–191.

Kim, Y. & Park, C. H., 2017, 'An efficient concept drift detection method for streaming data under limited labeling', *IEICE Transactions on Information and systems*, vol. 100, no. 10, pp. 2537–2546.

Kolter, J. Z. & Maloof, M. A., 2007, 'Dynamic weighted majority: An ensemble method for drifting concepts', *The Journal of Machine Learning Research*, vol. 8, pp. 2755–2790.

Korycki, L. & Krawczyk, B., 2021, 'Streaming decision trees for lifelong learning', *ECML PKDD*, .

Krizhevsky, A., Hinton, G. et al., 2009, 'Learning multiple layers of features from tiny images', .

Lao, Q., Jiang, X., Havaei, M. & Bengio, Y., 2020, 'Continuous domain adaptation with variational domain-agnostic feature replay', *arXiv preprint arXiv:2003.04382*.

Lei, T., Liu, P., Jia, X., Zhang, X., Meng, H. & Nandi, A. K., 2020, 'Automatic Fuzzy Clustering Framework for Image Segmentation', *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 9, pp. 2078–2092.

Li, K., Lu, J., Zuo, H. & Zhang, G., 2023a, 'Source-free multi-domain adaptation with fuzzy rule-based deep neural networks', *IEEE Transactions on Fuzzy Systems*.

Li, P., Wu, X., Hu, X. & Wang, H., 2015, 'Learning concept-drifting data streams with random ensemble decision trees', *Neurocomputing*, vol. 166, pp. 68–83.

Li, R., Cheong, L.-F. & Tan, R. T., 2019, 'Heavy rain image restoration: Integrating physics model and conditional adversarial learning', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1633–1642.

Li, R., Tan, R. T. & Cheong, L.-F., 2020, 'All in one bad weather removal using architectural search', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3175–3185.

Li, W., Yang, X., Liu, W., Xia, Y. & Bian, J., 2022, 'Ddg-da: Data distribution generation for predictable concept drift adaptation', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 36pp. 4092–4100.

Li, Y., Tan, R. T., Guo, X., Lu, J. & Brown, M. S., 2016, 'Rain streak removal using layer priors', *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2736–2744.

Li, Y., Xu, X., Su, Y. & Jia, K., 2023b, 'On the robustness of open-world test-time training: Self-training with dynamic prototype expansion', *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11836–11846.

Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L. & Timofte, R., 2021, 'Swinir: Image restoration using swin transformer', *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1833–1844.

Liang, J., He, R. & Tan, T., 2023, 'A comprehensive survey on test-time adaptation under distribution shifts', *arXiv preprint arXiv:2303.15361*.

Liang, J., Hu, D. & Feng, J., 2020, 'Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation', *International conference on machine learning*, PMLR, pp. 6028–6039.

Liu, A., Lu, J. & Zhang, G., 2020a, 'Concept drift detection: dealing with missing values via fuzzy distance estimations', *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 11, pp. 3219–3233.

Liu, A., Lu, J. & Zhang, G., 2020b, 'Diverse instance-weighting ensemble based on region drift disagreement for concept drift adaptation', *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 293–307.

Liu, A., Zhang, G. & Lu, J., 2017, 'Fuzzy time windowing for gradual concept drift adaptation', *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, pp. 1–6.

Liu, D., Wu, Y. & Jiang, H., 2016, 'Fp-elm: An online sequential learning algorithm for dealing with concept drift', *Neurocomputing*, vol. 207, pp. 322–334.

Liu, H., Long, M., Wang, J. & Wang, Y., 2020c, 'Learning to adapt to evolving domains.', *NeurIPS*, .

Liu, J., Yang, W., Yang, S. & Guo, Z., 2019, 'D3r-net: Dynamic routing residue recurrent network for video rain removal', *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 699–712.

Liu, L., Yuan, S., Liu, J., Guo, X., Yan, Y. & Tian, Q., 2022, 'Siamtrans: zero-shot multi-frame image restoration with pre-trained siamese transformers', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 36pp. 1747–1755.

Liu, Y.-F., Jaw, D.-W., Huang, S.-C. & Hwang, J.-N., 2018, 'Desnownet: Context-aware deep network for snow removal', *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 3064–3073.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S. & Guo, B., 2021, 'Swin transformer: Hierarchical vision transformer using shifted windows', *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022.

Long, M., Zhu, H., Wang, J. & Jordan, M. I., 2017, 'Deep transfer learning with joint adaptation networks', *International conference on machine learning*, PMLR, pp. 2208–2217.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S. & Zhang, G., 2015, 'Transfer learning using computational intelligence: A survey', *Knowledge-Based Systems*, vol. 80, pp. 14–23.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J. & Zhang, G., 2018a, 'Learning under concept drift: A review', *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363.

Lu, J., Xuan, J., Zhang, G. & Luo, X., 2018b, 'Structural property-aware multilayer network embedding for latent factor analysis', *Pattern Recognition*, vol. 76, pp. 228–241.

Lu, J., Zuo, H. & Zhang, G., 2019, 'Fuzzy multiple-source transfer learning', *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 12, pp. 3418–3431.

Lu, N., Zhang, G. & Lu, J., 2014, 'Concept drift detection via competence models', *Artificial Intelligence*, vol. 209, pp. 11–28.

Luo, Y., Xu, Y. & Ji, H., 2015, 'Removing rain from a single image via discriminative sparse coding', *Proceedings of the IEEE international conference on computer vision*, pp. 3397–3405.

Marcus de Carvalho, M., Pratama, M., Zhang, J. & Yapp, E., 2021, 'Acdc: Online unsupervised cross-domain adaptation', *arXiv preprint arXiv:2110.01326*.

Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D. & Van De Weijer, J., 2022, 'Class-incremental learning: survey and performance evaluation on image classification', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 5513–5533.

McKay, H., Griffiths, N., Taylor, P., Damoulas, T. & Xu, Z., 2019, 'Online transfer learning for concept drifting data streams.', *BigMine@KDD*, .

Miyaguchi, K. & Kajino, H., 2019, 'Cogra: Concept-drift-aware stochastic gradient descent for time-series forecasting', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 33pp. 4594–4601.

Montiel, J., Read, J., Bifet, A. & Abdessalem, T., 2018, 'Scikit-multiflow: A multi-output streaming framework', *Journal of Machine Learning Research*, vol. 19, no. 72, pp. 1–5, <http://jmlr.org/papers/v19/18-251.html>.

Nishida, K. & Yamauchi, K., 2007, 'Detecting concept drift using statistical testing', *International Conference on Discovery Science*, Springer, pp. 264–269.

Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P. & Tan, M., 2023, 'Towards stable test-time adaptation in dynamic wild world', *The Eleventh International Conference on Learning Representations*, .

Niu, S., Wu, J., Zhang, Y., Xu, G., Li, H., Zhao, P., Huang, J., Wang, Y. & Tan, M., 2022, 'Boost test-time performance with closed-loop inference', *arXiv preprint arXiv:2203.10853*.

Oliveira, G., Minku, L. L. & Oliveira, A. L., 2021, 'Tackling virtual and real concept drifts: An adaptive gaussian mixture model approach', *IEEE Transactions on Knowledge and Data Engineering*.

Oza, N. C. & Russell, S., 2001, 'Experimental comparisons of online and batch versions of bagging and boosting', *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 359–364.

Özdenizci, O. & Legenstein, R., 2023, 'Restoring vision in adverse weather conditions with patch-based denoising diffusion models', *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Patil, P. W., Gupta, S., Rana, S. & Venkatesh, S., 2022, 'Dual-frame spatio-temporal feature modulation for video enhancement', *Pattern Recognition*, vol. 130, p. 108822.

Patil, P. W., Gupta, S., Rana, S., Venkatesh, S. & Murala, S., 2023, 'Multi-weather image restoration via domain translation', *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21696–21705.

Poggi, M., Tosi, F., Batsos, K., Mordohai, P. & Mattoccia, S., 2021, 'On the synergies between machine learning and binocular stereo for depth estimation from images: a survey', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5314–5334.

Polikar, R., 2012, 'Ensemble learning', *Ensemble Machine Learning*, Springer, pp. 1–34.

Pratama, M., de Carvalho, M., Xie, R., Lughofer, E. & Lu, J., 2019, 'Atl: Autonomous knowledge transfer from many streaming processes', *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 269–278.

Pratama, M., Lu, J., Lughofer, E., Zhang, G. & Er, M. J., 2016, 'An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks', *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1175–1192.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J. et al., 2020, 'Exploring the limits of transfer learning with a unified text-to-text transformer.', *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67.

Rajderkar, D. & Mohod, P., 2013, 'Removing snow from an image via image decomposition', *2013 IEEE International Conference ON Emerging Trends in Computing, Communication and Nanotechnology (ICECCN)*, IEEE, pp. 576–579.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G. & Lampert, C. H., 2017, 'icarl: Incremental classifier and representation learning', *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010.

Ren, C., Jiang, B., Lu, N., Simani, S. & Gao, F., 2023, 'Meta-learning with distributional similarity preference for few-shot fault diagnosis under varying working conditions', *IEEE Transactions on Cybernetics*.

Renchunzi, X. & Pratama, M., 2022, 'Automatic online multi-source domain adaptation', *Information Sciences*, vol. 582, pp. 480–494.

Ritter, H., Botev, A. & Barber, D., 2018, 'Online structured laplace approximations for overcoming catastrophic forgetting', *Advances in Neural Information Processing Systems*, vol. 31.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. et al., 2015, 'Imagenet large scale visual recognition challenge', *International journal of computer vision*, vol. 115, pp. 211–252.

Settles, B., 2009, 'Active learning literature survey', .

Shi, K., Lu, J., Fang, Z. & Zhang, G., 2024, 'Unsupervised domain adaptation enhanced by fuzzy prompt learning', *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 7, pp. 4038–4048.

Simon, C., Faraki, M., Tsai, Y.-H., Yu, X., Schulter, S., Suh, Y., Harandi, M. & Chandraker, M., 2022, 'On generalizing beyond domains in cross-domain continual learning', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9265–9274.

Simonyan, K. & Zisserman, A., 2014, 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*.

Snell, J., Swersky, K. & Zemel, R., 2017, 'Prototypical networks for few-shot learning', *Advances in neural information processing systems*, vol. 30.

Song, Y., Lu, J., Liu, A., Lu, H. & Zhang, G., 2021a, 'A segment-based drift adaptation method for data streams', *IEEE Transactions on Neural Networks and Learning Systems*.

Song, Y., Lu, J., Lu, H. & Zhang, G., 2019, 'Fuzzy clustering-based adaptive regression for drifting data streams', *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 3, pp. 544–557.

Song, Y., Lu, J., Lu, H. & Zhang, G., 2021b, 'Learning data streams with changing distributions and temporal dependency', *IEEE Transactions on Neural Networks and Learning Systems*.

Song, Y., Lu, J., Lu, H. & Zhang, G., 2021c, 'Learning data streams with changing distributions and temporal dependency', *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14.

Song, Y., Zhang, G., Lu, H. & Lu, J., 2020, 'A fuzzy drift correlation matrix for multiple data stream regression', *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, pp. 1–6.

Song, Y., Zhang, G., Lu, J. & Lu, H., 2017, 'A fuzzy kernel c-means clustering model for handling concept drift in regression', *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, pp. 1–6.

Storkey, A., 2009, 'When training and test sets are different: characterizing learning transfer', *Dataset Shift in Machine Learning*, vol. 30, pp. 3–28.

Street, W. N. & Kim, Y., 2001, 'A streaming ensemble algorithm (sea) for large-scale classification', *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 377–382.

Su, T. & Dy, J. G., 2007, 'In search of deterministic methods for initializing k-means and gaussian mixture clustering', *Intelligent Data Analysis*, vol. 11, no. 4, pp. 319–338.

Sun, B., Feng, J. & Saenko, K., 2016, 'Return of frustratingly easy domain adaptation', *Proceedings of the AAAI conference on artificial intelligence*, , vol. 30.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A. & Hardt, M., 2020, 'Test-time training with self-supervision for generalization under distribution shifts', *International conference on machine learning*, PMLR, pp. 9229–9248.

Takagi, T. & Sugeno, M., 1985, 'Fuzzy identification of systems and its applications to modeling and control', *IEEE transactions on systems, man, and cybernetics*, , no. 1, pp. 116–132.

Tao, H., Wang, Z., Li, Y., Zamani, M. & Khan, L., 2019, 'Comc: A framework for online cross-domain multistream classification', *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Tao, X., Hong, X., Chang, X., Dong, S., Wei, X. & Gong, Y., 2020, 'Few-shot class-incremental learning', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12183–12192.

Van Der Maaten, L., Postma, E. & Van den Herik, J., 2009, 'Dimensionality reduction: a comparative', *J Mach Learn Res*, vol. 10, no. 66-71, p. 13.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I., 2017, 'Attention is all you need', *Advances in neural information processing systems*, vol. 30.

Vyas, A., Kannao, R., Bhargava, V. & Guha, P., 2014, 'Commercial block detection in broadcast news videos', *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, pp. 1–7.

Wah, C., Branson, S., Welinder, P., Perona, P. & Belongie, S., 2011, 'The caltech-ucsd birds-200-2011 dataset', .

Wang, D., Shelhamer, E., Liu, S., Olshausen, B. & Darrell, T., 2020, 'Tent: Fully test-time adaptation by entropy minimization', *arXiv preprint arXiv:2006.10726*.

Wang, F.-Y., Zhou, D.-W., Ye, H.-J. & Zhan, D.-C., 2022a, 'Foster: Feature boosting and compression for class-incremental learning', *European conference on computer vision*, Springer, pp. 398–414.

Wang, H. & Abraham, Z., 2015, 'Concept drift detection for streaming data', *2015 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–9.

Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W. & Yu, P., 2022b, 'Generalizing to unseen domains: A survey on domain generalization', *IEEE Transactions on Knowledge and Data Engineering*.

Wang, K., Lu, J., Liu, A., Song, Y., Xiong, L. & Zhang, G., 2022c, 'Elastic gradient boosting decision tree with adaptive iterations for concept drift adaptation', *Neurocomputing*, vol. 491, pp. 288–304.

Wang, K., Lu, J., Liu, A. & Zhang, G., 2023a, 'An augmented learning approach for multiple data streams under concept drift', *Australasian Joint Conference on Artificial Intelligence*, Springer, pp. 391–402.

Wang, K., Lu, J., Liu, A., Zhang, G. & Xiong, L., 2021, 'Evolving gradient boost: A pruning scheme based on loss improvement ratio for learning under concept drift', *IEEE Transactions on Cybernetics*.

Wang, P., Zhang, Z., Lei, Z. & Zhang, L., 2023b, 'Sharpness-aware gradient matching for domain generalization', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3769–3778.

Wang, Q.-F., Geng, X., Lin, S.-X., Xia, S.-Y., Qi, L. & Xu, N., 2022d, 'Learngene: From open-world to your learning task', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 36pp. 8557–8565.

Wang, Q.-W., Zhou, D.-W., Zhang, Y.-K., Zhan, D.-C. & Ye, H.-J., 2024a, 'Few-shot class-incremental learning via training-free prototype calibration', *Advances in Neural Information Processing Systems*, vol. 36.

Wang, R., Zuo, H., Fang, Z. & Lu, J., 2024b, 'Towards robustness prompt tuning with fully test-time adaptation for clip's zero-shot generalization', *ACM Multimedia 2024*, .

Wang, T., Tao, G., Lu, W., Zhang, K., Luo, W., Zhang, X. & Lu, T., 2024c, 'Restoring vision in hazy weather with hierarchical contrastive learning', *Pattern Recognition*, vol. 145, p. 109956.

Wang, Y., Liu, J., Ma, J., Zeng, H., Zhang, L. & Li, J., 2023c, 'Dynamic dual graph networks for textbook question answering', *Pattern Recognition*, vol. 139, p. 109441.

Wang, Y., Yan, X., Wang, F. L., Xie, H., Yang, W., Zhang, X.-P., Qin, J. & Wei, M., 2024d, 'Ucl-dehaze: Towards real-world image dehazing via unsupervised contrastive learning', *IEEE Transactions on Image Processing*.

Wang, Z., Cun, X., Bao, J., Zhou, W., Liu, J. & Li, H., 2022e, 'Uformer: A general u-shaped transformer for image restoration', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17683–17693.

Wang, Z., Dai, Z., Póczos, B. & Carbonell, J., 2019, 'Characterizing and avoiding negative transfer', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11293–11302.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J. & Pfister, T., 2022f, 'Learning to prompt for continual learning', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149.

Wang, Z., Zhou, C., Du, B. & He, F., 2022g, 'Self-paced supervision for multi-source domain adaptation', *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, .

Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., Tomizuka, M., Gonzalez, J., Keutzer, K. & Vajda, P., 2020, 'Visual transformers: Token-based image representation and processing for computer vision', *arXiv preprint arXiv:2006.03677*.

Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollár, P. & Girshick, R., 2021, 'Early convolutions help transformers see better', *Advances in Neural Information Processing Systems*, vol. 34, pp. 30392–30400.

Xu, F., Liu, J., Lin, Q., Pan, Y. & Zhang, L., 2022, 'Logiformer: a two-branch graph transformer network for interpretable logical reasoning', *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1055–1065.

Xu, J., Zhao, W., Liu, P. & Tang, X., 2012, 'Removing rain and snow in a single image using guided filter', *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, , vol. 2IEEE, pp. 304–307.

Xu, K., Ma, Y., Wei, B. & Li, W., 2023, 'Open-ended diverse solution discovery with regulated behavior patterns for cross-domain adaptation', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 37pp. 10585–10593.

Xu, P., Deng, Z., Wang, J., Zhang, Q., Choi, K.-S. & Wang, S., 2019, 'Transfer representation learning with tsk fuzzy system', *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 3, pp. 649–663.

Xu, S. & Wang, J., 2017, 'Dynamic extreme learning machine for data stream classification', *Neurocomputing*, vol. 238, pp. 433–449.

Yamashita, A., Tanaka, Y. & Kaneko, T., 2005, 'Removal of adherent waterdrops from images acquired with stereo camera', *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, pp. 400–405.

Yan, C., Chang, X., Li, Z., Guan, W., Ge, Z., Zhu, L. & Zheng, Q., 2021a, 'Zeronas: Differentiable generative adversarial networks search for zero-shot learning', *IEEE transactions on pattern analysis and machine intelligence*.

Yan, S., Xie, J. & He, X., 2021b, 'Der: Dynamically expandable representation for class incremental learning', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3014–3023.

Yang, C., Cheung, Y.-m., Ding, J. & Tan, K. C., 2021, 'Concept drift-tolerant transfer learning in dynamic environments', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3857–3871.

Yang, C., Deng, Z., Choi, K.-S. & Wang, S., 2015, 'Takagi–sugeno–kang transfer learning fuzzy logic system for the adaptive recognition of epileptic electroencephalogram signals', *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 5, pp. 1079–1094.

Yang, F., Ren, J., Lu, Z., Zhang, J. & Zhang, Q., 2022, 'Rain-component-aware capsule-gan for single image de-raining', *Pattern Recognition*, vol. 123, p. 108377.

Yang, W., Tan, R. T., Wang, S., Fang, Y. & Liu, J., 2020, 'Single image deraining: From model-based to data-driven and beyond', *IEEE Transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 4059–4077.

Yao, H., Wang, Y., Wei, Y., Zhao, P., Mahdavi, M., Lian, D. & Finn, C., 2021, 'Meta-learning with an adaptive task scheduler', *Advances in Neural Information Processing Systems*, vol. 34.

Yin, S., Wang, Y. & Yang, Y.-H., 2020, 'A novel image-dehazing network with a parallel attention block', *Pattern Recognition*, vol. 102, p. 107255.

Yoon, S., Lee, Y., Lee, J.-G. & Lee, B. S., 2022, 'Adaptive model pooling for online deep anomaly detection from a complex evolving data stream', *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2347–2357.

Yu, E., Lu, J., Zhang, B. & Zhang, G., 2024a, 'Online boosting adaptive learning under concept drift for multistream classification', *Proceedings of the AAAI Conference on Artificial Intelligence*, , vol. 38pp. 16522–16530.

Yu, E., Lu, J. & Zhang, G., 2024b, 'Fuzzy shared representation learning for multistream classification', *IEEE Transactions on Fuzzy Systems*.

Yu, E., Song, Y., Zhang, G. & Lu, J., 2022a, 'Learn-to-adapt: Concept drift adaptation for hybrid multiple streams', *Neurocomputing*, vol. 496, pp. 121–130.

Yu, H., Liu, W., Lu, J., Wen, Y., Luo, X. & Zhang, G., 2023, 'Detecting group concept drift from multiple data streams', *Pattern Recognition*, vol. 134, p. 109113.

Yu, H., Lu, J. & Zhang, G., 2020a, 'An online robust support vector regression for data streams', *IEEE Transactions on Knowledge and Data Engineering*.

Yu, H., Lu, J. & Zhang, G., 2020b, 'Topology learning-based fuzzy random neural networks for streaming data regression', *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 2, pp. 412–425.

Yu, H., Zhang, Q., Liu, T., Lu, J., Wen, Y. & Zhang, G., 2022b, 'Meta-add: A meta-learning based pre-trained model for concept drift active detection', *Information Sciences*, vol. 608, pp. 996–1009.

Yu, S. & Abraham, Z., 2017, 'Concept drift detection with hierarchical hypothesis testing', *Proceedings of the 2017 SIAM International Conference on Data Mining*, SIAM, pp. 768–776.

Yu, S., Wang, X. & Príncipe, J. C., 2018, 'Request-and-reverify: Hierarchical hypothesis testing for concept drift detection with expensive labels', *arXiv preprint arXiv:1806.10131*.

Yuan, F., Zhang, Z. & Fang, Z., 2023, 'An effective cnn and transformer complementary network for medical image segmentation', *Pattern Recognition*, vol. 136, p. 109228.

Yuan, F., Zhou, Y., Xia, X., Qian, X. & Huang, J., 2021, 'A confidence prior for image dehazing', *Pattern Recognition*, vol. 119, p. 108076.

Yue, L., Chen, W., Li, X., Zuo, W. & Yin, M., 2019, 'A survey of sentiment analysis in social media', *Knowledge and Information Systems*, vol. 60, no. 2, pp. 617–663.

Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S. & Yang, M.-H., 2022, 'Restormer: Efficient transformer for high-resolution image restoration', *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5728–5739.

Zhang, A., Xing, L., Zou, J. & Wu, J. C., 2022a, 'Shifting machine learning for healthcare from development to deployment and from models to data', *Nature Biomedical Engineering*, vol. 6, no. 12, pp. 1330–1345.

Zhang, C., Song, N., Lin, G., Zheng, Y., Pan, P. & Xu, Y., 2021a, 'Few-shot incremental learning with continually evolved classifiers', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12455–12464.

Zhang, H. & Patel, V. M., 2018, 'Density-aware single image de-raining using a multi-stream dense network', *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 695–704.

Zhang, K., Li, D., Luo, W., Ren, W. & Liu, W., 2022b, 'Enhanced spatio-temporal interaction learning for video deraining: A faster and better framework', *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, K., Li, R., Yu, Y., Luo, W. & Li, C., 2021b, 'Deep dense multi-scale network for snow removal using semantic and depth priors', *IEEE Transactions on Image Processing*, vol. 30, pp. 7419–7431.

Zhang, K., Luo, W., Yu, Y., Ren, W., Zhao, F., Li, C., Ma, L., Liu, W. & Li, H., 2022c, 'Beyond monocular deraining: Parallel stereo deraining network via semantic prior', *International Journal of Computer Vision*, pp. 1–16.

Zhang, M., Levine, S. & Finn, C., 2022d, 'Memo: Test time robustness via adaptation and augmentation', *Advances in neural information processing systems*, vol. 35, pp. 38629–38642.

Zhang, P., Zhang, K., Luo, W., Li, C. & Wang, G., 2022e, 'Blind face restoration: Benchmark datasets and a baseline model', *arXiv preprint arXiv:2206.03697*.

Zhang, P., Zhu, X. & Shi, Y., 2008, 'Categorizing and mining concept drifting data streams', *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 812–820.

Zhang, W., Deng, Z., Zhang, T., Choi, K.-S. & Wang, S., 2023, 'Multi-view fuzzy representation learning with rules based model', *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, Y., Chu, G., Li, P., Hu, X. & Wu, X., 2017, 'Three-layer concept drifting detection in text data streams', *Neurocomputing*, vol. 260, pp. 393–403.

Zhao, B., Chen, C. & Xia, S.-T., 2022, 'Delta: Degradation-free fully test-time adaptation', *The Eleventh International Conference on Learning Representations*, .

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H. et al., 2021, 'Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6881–6890.

Zhong, L., Fang, Z., Liu, F., Lu, J., Yuan, B. & Zhang, G., 2021, 'How does the combined risk affect the performance of unsupervised domain adaptation approaches?', *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, .

Zhou, D.-W., Wang, F.-Y., Ye, H.-J., Ma, L., Pu, S. & Zhan, D.-C., 2022, 'Forward compatible few-shot class-incremental learning', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9046–9056.

Zhou, D.-W., Wang, Q.-W., Qi, Z.-H., Ye, H.-J., Zhan, D.-C. & Liu, Z., 2023a, 'Deep class-incremental learning: A survey', *arXiv preprint arXiv:2302.03648*.

Zhou, M., Lu, J., Song, Y. & Zhang, G., 2023b, 'Multi-stream concept drift self-adaptation using graph neural network', *IEEE Transactions on Knowledge and Data Engineering*.

Zhou, M., Song, Y., Zhang, G., Zhang, B. & Lu, J., 2021, 'An efficient bayesian neural network for multiple data streams', *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Zhou, Z., Guo, L.-Z., Jia, L.-H., Zhang, D. & Li, Y.-F., 2023c, 'Ods: Test-time adaptation in the presence of open-world data shift', .

Zhu, X., Su, W., Lu, L., Li, B., Wang, X. & Dai J F, D. D., 2021, 'Deformable transformers for end-to-end object detection', *Proceedings of the 9th International Conference on Learning Representations. Virtual Event, Austria: OpenReview. net*, .

Zhuang, J., Gong, B., Yuan, L., Cui, Y., Adam, H., Dvornek, N., Tatikonda, S., Duncan, J. & Liu, T., 2022, 'Surrogate gap minimization improves sharpness-aware training', *arXiv preprint arXiv:2203.08065*.