Faculty of Computer and Information

Menoufia University

Learning in Nonstationary Environment:Concept Drift handling techniques

A thesis submitted to the Faculty of Computers and Information, Menoufia University, in partial fulfillment of the requirements for the degree of Master of Computers and Information

In

[Information Systems]

By

Ahmed Hamdy Madkour

Teaching Assessment at Information Systems Department, Faculty of Computers and Information, Menoufia University

Supervised By

Prof. Hatem Mohamed
Professor of Information Systems,
Dean for Faculty of Computers and Information, Menoufia University

[ ]

Dr. Amgad Monir
Lecturer of Information Systems,
Faculty of Computers and Information, Menoufia University

[ ]

2024

Faculty of Computer and Information

Menoufia University

Learning in Nonstationary Environment:Concept Drift handling techniques

A thesis submitted to the Faculty of Computers and Information, Menoufia University, in partial fulfillment of the requirements for the degree of Master of Computers and Information

In

[Information Systems]

By

Ahmed Hamdy Madkour

Teaching Assessment at Information Systems Department, Faculty of Computers and Information, Menoufia University

Examination Committee

Prof. Hatem Mohamed
Professor of Information Systems,
Dean for Faculty of Computers and Information, Menoufia University
[ ]

Prof. Hatem Mohamed
Professor of Information Systems,
Faculty of Computers and Information,
Menoufia University
[ ]

Prof. Hatem Mohamed
LProfessor of Information Systems,
Faculty of Computers and Information,
Menoufia University
[ ]

2024

Dedicated to everyone who wish for a better life!

Abstract

This thesis addresses challenges in machine learning models when dealing with multi-class imbalanced data streams in non-stationary environments. These challenges result in biases, unreliable predictions, and diminished model performance. To overcome these issues, the thesis introduces innovative approaches that combines Dynamic Ensemble Selection (DES) technique, an adaptive method for handling imbalanced multi-class data streams, a concept drift detector, eigen vector technique, and the K-Nearest Neighbors (KNN) algorithm to address class overlap. The main objective is to improve the classification of imbalanced multi-class drifted data streams. The adaptive oversampling method generates synthetic samples to mitigate imbalanced data stream issues, with KNN ensuring non-overlapping generated samples. A drift detector assists in deciding whether to keep existing classifiers or create new ones to handle incoming data streams. Dynamic Ensemble Selection (DES) optimizes the classification task by selecting the most appropriate classifier for incoming data. The proposed method offers an effective solution for achieving accurate and resilient classification in the context of imbalanced multi-class drifted data streams. Additionally, the thesis discusses challenges posed by emerging new classes in dynamic data streams for incremental learning. Existing approaches struggle with high false positive rates, long prediction times, and the impractical assumption of having access to true labels for all instances when new classes emerge. To address these challenges, the thesis proposes a novel framework that integrates concept drift detection using the ADWIN method and ensemble stratified bagging. By leveraging true labels during the learning process, the framework effectively detects concept drifts and adapts to accommodate emerging

new classes. Furthermore, the thesis discusses challenges faced by machine learning models in heterogeneous multi-source streams and non-stationary conditions, which can introduce biases and unreliable predictions. In response, the thesis introduces a comprehensive framework that combines dynamic ensemble selection, eigenvector techniques, and a concept drift detector to enhance transfer learning of multi-class data streams subject to drift. During the training phase, the proposed weighted method assigns appropriate weights to each classifier, mitigating adverse learning effects. Eigenvectors handle heterogeneous multi-source streams, and Dynamic Ensemble Selection (DES) determines the most fitting classifiers for incoming data. A concept drift detector identifies and addresses drifted chunks within the data stream. The experimental results on various datasets, encompassing benchmark datasets, a real application stream dataset, and synthetic data streams, consistently demonstrate the superior performance of the proposed approach in effectively addressing challenges inherent in such dynamic data streams. Through evaluations conducted on both real-world and synthetic datasets, the framework exhibits exceptional accuracy, precision, recall, geometric mean, and F1-measure, all while maintaining an efficient runtime. This robust performance positions the proposed approach as a frontrunner in enabling incremental learning within real-time scenarios, particularly in the context of emerging new classes. The framework provides a comprehensive solution to the intricate complexities associated with incremental learning in dynamic data streams. Moreover, the experimental findings underscore the efficacy and superiority of the proposed approach, particularly when confronted with the unique challenges posed by imbalanced drifted streams, emerging new classes, and heterogeneous multi-source drifted data streams. Comparisons against state-of-the-art chunk-based methods consistently highlight the effectiveness of this approach in successfully managing the complexities inherent in these multifaceted data stream challenges. In essence, the proposed approach stands out as a powerful and versatile solution, showcasing its ability to outperform

existing methods and addressing the evolving demands of imbalanced drifted streams and heterogeneous multi-source streams. Keywords: - Auto machine learning, Concept drift, Imbalanced Stream, Transfer Learning, Emerging new Classes, and Dynamic Ensemble Selection.

Summary

The thesis proposes three innovative solutions to challenges faced by machine learning models in handling multi-class imbalanced data streams, emerging new classes, and heterogeneous muti-class streams in non-stationary environments. The primary issues include bias, unreliable predictions, and decreased overall model performance. The first proposed approach integrates the Dynamic Ensemble Selection (DES), an adaptive oversampling method for handling imbalanced multi-class data streams, a concept drift detector, and the K-Nearest Neighbors (KNN) algorithm to address class overlap. The adaptive oversampling method generates synthetic samples, leveraging KNN to ensure non-overlapping samples. A drift detector helps decide whether to keep existing classifiers or create new ones for incoming data streams. Dynamic Ensemble Selection optimizes classifier performance. The approach demonstrates effectiveness in achieving accurate and resilient classification in imbalanced multi-class drifted data streams, as validated through experiments on various datasets. Similarly, the second paper addresses the challenge of emerging new classes in dynamic data streams, affecting incremental learning. The second proposed approach integrates concept drift detection using the ADWIN method and ensemble stratified bagging. Leveraging true labels during the learning process, this approach detects concept drifts and adapts to emerging new classes. Evaluation using real and synthetic datasets shows superiority in accuracy, precision, recall, geometric mean, and F1 measure while maintaining efficient runtime. The framework promises to enable incremental learning in real-time scenarios, offering a comprehensive solution by combining concept drift detection (ADWIN) and ensemble stratified bagging. Lastly, the third approach tackles challenges

arising from heterogeneous multi-source streams and non-stationary conditions in machine learning models. The third proposed approach combines DES, eigen vector techniques, and a concept drift detector to enhance transfer learning of multi-class data streams subject to drift. The weighted method assigns appropriate weights to each classifier during training, mitigating adverse learning effects. Eigenvectors handle heterogeneous multi-source streams, and DES determines fitting classifiers for incoming data. The concept drift detector identifies and addresses drifted chunks within the data stream. Experiments on various datasets demonstrate the efficacy and superiority of the proposed approach in addressing challenges presented by transfer learning from heterogeneous multi-sources and concept drift, outperforming state-of-the-art chunk-based methods.

Acknowledgements

First and foremost, I give my deep thanks to Allah for giving me the opportunity and the strength to accomplish this work.

I would like to thank my supervisors Dr. Hatem Mohammed and Dr. Amgad Monir for their help and support during my work for creating a very inspiring research environment. They have been helpful with background information and have continually encouraged me and helped me with comments on my work. They always had time to discuss new ideas and give feedback on early ideas and research problems.

I would like to thank my family who was a constant source of rising and supporting my spirit. Thanks go out especially to my father, my mother, my wife, and my brothers for support and encouragement.

Finally, special thanks to my faculty, department, and my colleagues.

Thank you everyone,

Ahmed Madkour

August 2024

Success is the sum of small efforts,
repeated day-in and day-out.

Robert Collier

CHAPTER

1

Introduction

Governments and companies are producing vast streams of data and require effective data analytics and machine learning methods to assist in making predictions and decisions promptly. One crucial aspect is the machine learning pipeline, which involves training a prepared dataset to construct a model and subsequently utilizing this model to predict new instance outputs. As depicted in Fig. (1.1), the process entails fetching historical data from the database during the training phase to construct the machine learning model. Then, the system can input new instances from the database to predict the output.

Nevertheless, when endeavoring to forecast outcomes for fresh instances sourced from an alternative database, as illustrated in Fig. (1.2), there frequently emerges a conspicuous decline in accuracy. This disparity accentuates the imperative for model developers to intervene and rectify the issue. Addressing this, developers must adjust and retrain the model utilizing datasets from the new environment to ameliorate accuracy. This iterative process aims to refine the model's precision and ensure its efficacy across diverse contexts, thereby bolstering the reliability of decision-making and predictive capabilities. To confront this challenge, the field of auto machine learning endeavors to facilitate online updates to the model without necessitating direct intervention from developers for modification.

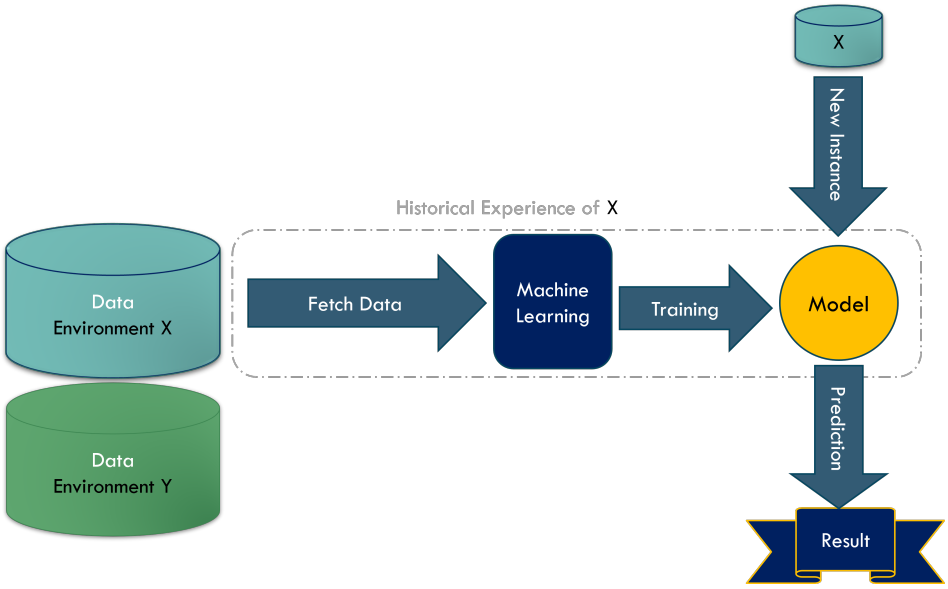
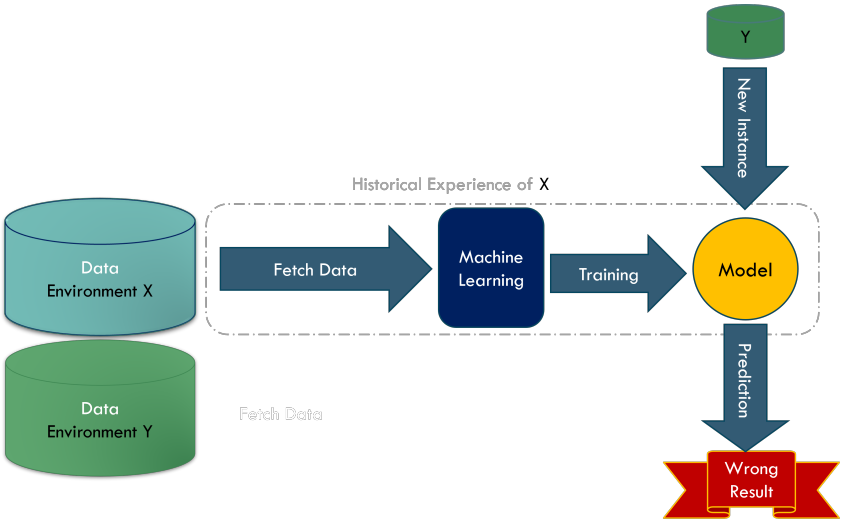


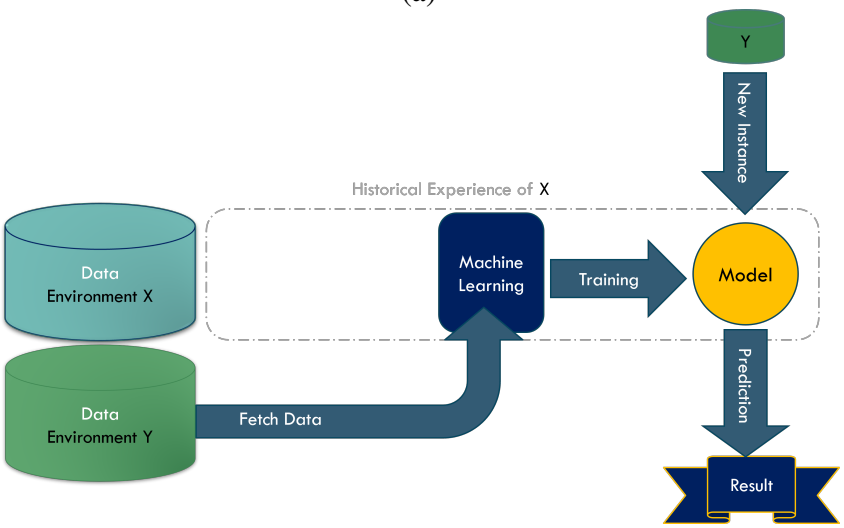
Figure 1.1: The research methodology of the thesis.

In recent years, the surge in high-speed data streams has posed notable challenges for machine learning models, particularly in the context of streaming data analysis. These data streams, characterized by continuous, dynamic, and high-volume data arrivals, demand adaptive learning algorithms that can effectively cope with their evolving nature [1] [2] [3]. Within these evolving data environments, two paramount challenges have emerged: concept drift, class imbalance, Emerging new class, and heterogenous transfer learning.

Concept drift, a phenomenon defined by the evolving statistical properties of a data generation process over time [4] [5]. introduces a dynamic element to the data, necessitating continuous adaptation of machine learning models. This shift can manifest as changes in underlying concepts, relationships between variables, or alterations in data distribution. Traditional models trained on historical data may suffer diminished accuracy or become inadequate when confronted with new data influenced by concept drift, highlighting the need for effective concept drift detection mechanisms. Addressing concept drift involves the utilization of concept drift detectors, which are methods capable of identifying changes in data stream distributions. These detectors rely on information related to classifier performance or incoming data items to signal the need for model updates, retraining, or even



(a)



(b)

Figure 1.2: The research methodology of the thesis.

replacing the old model with a new one. The dynamic nature of concept drift necessitates ongoing monitoring and adaptation to maintain the model's efficacy.

Data streams also present challenges related to class imbalance, a condition characterized by uneven distribution among different classes [6] [7]. This scenario, especially prevalent in multi-class settings, poses a significant challenge for traditional classifiers. The risk of misclassifying minority class samples due to their

limited representation demands specialized techniques to ensure accurate classification without sacrificing the performance of the majority class [8] [9] [10] [11]. To tackle class imbalance, three primary methods are commonly employed: sampling methods, algorithm adaptation methods, and hybrid methods. Sampling methods involve undersampling the majority class or oversampling the minority class to balance class distribution. Algorithm adaptation methods modify existing algorithms to handle imbalanced data [12] [13] [14], while hybrid methods combine data preprocessing with classification techniques, often utilizing ensemble classifiers to effectively mitigate class imbalance and enhance overall classifier performance [15] [16] [17] [18].

Another challenge arising in the context of class imbalance is class overlap, where instances from different classes share the same region in data space [17] [18]. This overlap complicates the task of distinguishing between representative instances of different classes, leading to performance challenges for traditional classifiers referred to as overlapping problems. Recent research introduces class-overlap undersampling methods to address this issue, leveraging local similarities among minority instances to identify potentially overlapping majority instances.

Therefore, both class imbalance and class overlap present significant hurdles in the realm of data stream analysis. Consequently, addressing class imbalance has become crucial in multi-class learning, leading to research efforts focusing on both concept drift and class imbalance challenges. Researchers have explored dynamic ensemble selection (DES) and multi-class oversampling techniques to tackle these issues. Dynamic classifier ensembles offer a unique ability to adapt their composition based on data characteristics, making them valuable in situations with evolving data conditions [19]. Researchers focus on the overproduce-and-select approach for classifier ensemble selection methods. The objective of classifier ensemble selection is to choose an optimal subset of classifiers from a larger ensemble. The selection process is guided by various criteria, including individual performance measures, diversity metrics, meta-learning techniques, and performance estimation approaches. This optimization is particularly important in scenarios where a balance between accuracy and computational resource constraints is critical. There are two distinct approaches: static and dynamic selection. Static selection involves

assigning classifiers to predefined feature partitions, while dynamic selection adaptively selects classifiers based on their competency [20]. Dynamic selection offers two choices: individual models, known as Dynamic Classifier Selection (DCS), and ensemble models, called Dynamic Ensemble Selection (DES). DCS algorithms enable the selection of the most appropriate classifier for each data point based on its local competencies. In contrast, DES focuses on selecting the optimal classifiers for each instance based on their competence within localized regions [21] [22] [23] Competency assessment relies on a dynamic selection dataset (DSEL) containing labeled samples. Moreover, innovative techniques like the Randomized Reference Classifier introduce randomness into class supports to enhance adaptability in addressing challenges related to imbalanced data.

Additionally, transfer learning assumes a pivotal role in addressing the intricate challenges posed by dynamic data streams and inherent concept drift. This domain of research focuses on enhancing a model's learning performance within a target domain by harnessing knowledge gleaned from source domains [4] [5].Techniques in transfer learning include reducing domain gaps through instance re-weighting and feature matching, along with strategies to mitigate negative knowledge transfer by down-weighting irrelevant source data.

Lastly, in the study, the focus extends to the specific scenario of Streams with Emerging New Classes (SENC). This refers to situations where new classes, not present during the initial training of a learning model, emerge in the data stream. Traditional learning approaches, designed for fixed or predefined class distributions, face challenges in effectively recognizing and adapting to these novel classes in real-time. The need for adaptive learning mechanisms that can handle the emergence of new classes underscores the complexity of real-world data stream scenarios.

In this chapter, the motivation for this research along with the research questions that naturally arise are discussed in Section **??**. After this, the objectives and contributions are presented in Sections **??** and **??**, respectively. Next, the research methodology is summarised in Section **??**. Finally, the research context and the outline of this thesis are presented in Sections **??** and **??**, respectively.

1.1 challenges

Within the swiftly evolving realm of machine learning, the proliferation of high-speed data streams presents a significant hurdle — the proficient management of non-stationary data environments. This challenge is marked by dynamic fluctuations in statistical attributes, fundamental concepts, and data distributions over time. The crux of the issue emerges as models, initially trained on historical data, experience a decline in accuracy when confronted with new data influenced by concept drift. This includes scenarios such as:

- Multi-class imbalanced data streams
- Overlapping classes
- Emergence of new classes
- Heterogeneous transfer learning

However, addressing these challenges yields substantial benefits. By effectively managing non-stationary data environments, organizations can enhance the adaptability and resilience of their machine learning systems. This enables more robust decision-making processes, improved predictive capabilities, and heightened performance across diverse contexts. Moreover, it fosters innovation and agility, empowering organizations to stay ahead in dynamic markets and evolving scenarios.

1.2 Motivation and Scope

1.2.1 Thesis Motivations

The rapid advancements in machine learning, particularly in the realm of real-time data streams, have ushered in a new era of challenges that demand innovative solutions. The primary motivations driving this thesis stem from the necessity to overcome the limitations of traditional models and methods when faced with dynamic, non-stationary data environments. These motivations are outlined as follows:

- **Adapting to Emerging Classes in Real-Time Scenarios:** In today's fast-paced data-driven world, the sudden emergence of new classes within data streams presents a critical challenge. Existing models often falter when confronted with such unforeseen changes, leading to significant declines in predictive accuracy. Our motivation lies in developing robust, real-time adaptive mechanisms that enable models to swiftly and effectively accommodate new classes, ensuring that the system remains relevant and accurate despite the evolving data landscape.

- **Proactive Management of Concept Drift:** As data streams evolve, the underlying concepts and statistical properties often shift, resulting in concept drift. This phenomenon can severely degrade the performance of models if not addressed promptly. Our research is motivated by the need to create sophisticated strategies that proactively detect and manage concept drift, ensuring that models can maintain high accuracy and adapt to changes in the data environment over time.

- **Dynamic Optimization of Classifier Ensembles:** The dynamic nature of non-stationary data streams necessitates the continuous optimization of classifier ensembles to handle emerging classes and shifting data distributions effectively. We are motivated to pioneer techniques that enable the real-time adjustment of classifier ensembles, thereby enhancing the overall performance of classification algorithms in evolving contexts.

- **Addressing Multi-Class Imbalance in Streaming Data:** Imbalanced data distributions, especially in multi-class scenarios, pose significant challenges for accurate classification. Traditional approaches often fail to adequately represent minority classes, leading to biased outcomes. Our motivation is to develop innovative methods that can dynamically address class imbalances, ensuring fair and accurate classification across all classes, even in non-stationary environments.

- **Enhancing Transfer Learning in Non-Stationary Contexts:** Transfer learning has emerged as a powerful technique for leveraging knowledge from diverse source domains. However, in non-stationary environments, the risk

of negative knowledge transfer can undermine model performance. Our research is driven by the need to create frameworks that facilitate effective transfer learning, minimizing negative transfer and maximizing the potential for positive knowledge transfer, thereby ensuring robust model performance across diverse and shifting data landscapes

1.2.2 Thesis Scope

This thesis aims to address the multifaceted challenges associated with managing non-stationary data streams through the development of innovative frameworks and methodologies. The scope of this research is defined by the following key areas:

- **Development of Adaptive Classifier Ensembles:** We will design and implement a dynamic classifier ensemble framework capable of real-time adaptation to emerging classes. This framework will be tested and validated across various streaming data scenarios to ensure its effectiveness in maintaining model accuracy amidst the continuous appearance of new classes.
- **Concept Drift Detection and Response Mechanisms:** The research will explore advanced methods for detecting and responding to concept drift within non-stationary data streams. We will focus on creating mechanisms that not only identify drift but also adapt the classification model in real-time to maintain high predictive accuracy.
- **Innovative Solutions for Multi-Class Imbalance:** The thesis will delve into the complexities of multi-class imbalanced data in streaming environments. We will propose and evaluate novel techniques for dynamically adjusting class distributions and optimizing classifier performance, with a particular focus on preserving the representation of minority classes.
- **Framework for Heterogeneous Transfer Learning:** A significant part of the research will involve developing a framework for heterogeneous transfer learning that addresses the challenges posed by non-stationary data streams. This framework will leverage the eigenvector technique to facilitate effective knowledge transfer, ensuring that models can adapt to new domains without suffering from negative transfer effects.

- **Real-Time Model Adaptation:** The scope of the thesis will include the creation of methodologies that enable real-time model adaptation in response to both emerging classes and shifting data distributions. These methodologies will be designed to operate efficiently within the constraints of high-speed data streams, minimizing computational complexity while maximizing classification accuracy.

1.2.3 Solution Methodology

Through these focused areas of research, the thesis aims to contribute significantly to the field of machine learning by providing practical, scalable solutions for the challenges posed by non-stationary data streams. The outcomes of this research are expected to enhance the adaptability, resilience, and performance of machine learning systems in real-time, dynamic environments.This thesis endeavors to devise solutions to its contributions through three distinct approaches:

- **Dynamic Classification Ensembles for handling Imbalanced Multi-class Drifted Data streams:** This approach is designed to address challenges posed by imbalanced multi-class streams and overlapping classes within data streams.

- **Addressing Emerging New Classes in Incremental Streams via Concept Drift Techniques :** TThis initiative aims to manage the emergence of new classes in non-stationary environments effectively.

- **Addressing Heterogeneous Transfer Learning Problem in data streams via Concept Drift:** This approach aims to handle the complexities of heterogeneous transfer learning from multiple sources in non-stationary environments.

1.3 Thesis Objectives

The thesis objectives for our research stems from the imperative need to address the following key motivations:

- **Objective 1**. Thandling Imbalanced Multiclass Drifted Data and overlapping classes streams.
- **Objective 2**. addressing Emerging New Classes in Incremental Streams via Concept Drift and K-means Techniques.
- **Objective 3**. addressing Heterogeneous Transfer Learning Problem in data streams via Concept Drift and Eigenvector Techniques.

1.4 Thesis Questions

- Q_1. What is the impact of reduced and consistent data on the performance of ensemble learning?
- Q_2. Is it possible with the search capability of swarm intelligence to enhance the combination of classifiers?
- Q_3. What is the effect of combining multiple pruning metrics together?
- Q_4. What is the effect of downsizing data and downsizing the number of classifiers simultaneously?
- Q_5. How the initial classifier pool size and the required subensemble size affect on the performance of heuristic pruning metrics?
- Q_6. How the heuristic pruning metrics are affected by the individual classifier type?
- Q_7. How the efficacy of the pruning metrics could be affected by binary and multiclass tasks?
- Q_8. How the pruning metrics are effective to reduce the performance variance?
- Q_9. How the efficiency of the heuristic pruning metrics differs in terms of time and space complexities?

1.5 Contributions

Our research endeavors to create advanced frameworks designed for effectively managing non-stationary data streams while prioritizing high accuracy and minimizing computational complexity. The key contributions of our work can be summarized as follows:

- **Incorporation of Concept Drift Detection and Ensemble Classifier:** Our primary innovation involves incorporating a concept drift detection method alongside an ensemble classifier. This integration enables real-time adaptation and refinement of our proposed framework in response to transfer learning in non-stationary environments. This methodology ensures the continuous evolution of our classification model in alignment with the changing data landscape.

- **Dynamic Classifier Ensemble Framework for Emergence Class Problem:** We introduce a classification framework that harnesses dynamic classifier ensemble techniques to address the emergence class problem. This framework enhances the performance of classification algorithms when dealing with non-stationary data streams featuring emerging new classes. By dynamically adjusting the ensemble based on data characteristics, our framework improves the accuracy and effectiveness of classification models, effectively tackling challenges associated with emerging new classes.

- **Introduction of Precise Weighting Method for Local Classifiers of the transfer learning framework:** A significant advancement in our research is the introduction of a precise weighting method to assess the significance of each local classifier within the ultimate classifier. This method enhances the accuracy of the overall classifier by providing a nuanced evaluation of individual classifier contributions.

- **Development of Innovative Framework using Eigenvector Technique for addressing heterogenous transfer learning problem:** A significant stride in our research involves the creation of an innovative framework, harnessing the power of the eigenvector technique. This framework is specifically designed to tackle the challenges posed by heterogeneous transfer learning problems. By leveraging the eigenvector technique, our framework enables the seamless transfer of knowledge from diverse source domains to the target domain, thereby enhancing adaptability and overall performance.

- **Dynamic Adjustment Method to Multi-class Imbalanced Data:** Our classification framework introduces a dynamic adjustment method tailored for

multi-class imbalanced data scenarios. It seamlessly incorporates concept drift detection mechanisms and optimizes classifier ensemble selections, aiming to significantly improve classification accuracy in the context of multi-class imbalanced non-stationary streams.

- **Adaptive Method for Class Imbalance:** Additionally, we propose an adaptive method for addressing the class imbalance issue. This method considers data distributions and historical instances of class imbalance, especially in cases where class overlap occurs within multi-class and drifted data streams. The adaptive approach improves classification performance by selecting the most suitable oversampling method based on the unique characteristics of the data stream.

1.6 Publications

During the research activities of this thesis, several international peer-reviewed journal articles were published to disseminate the obtained results. The publications can be found in Table 1.1.

sectionResearch Methodology The research field of this thesis is moving fast due to technological advances and the continuous generation of new contributions in ML. Consequently, an iterative research methodology was followed. The main idea of this cyclical process is that the knowledge acquired in its initial phase helps to design an increasingly promising technique, either in terms of accurate results, or in terms of concept, offering originality and remarkable contributions. Figure 1.1 shows the different phases of this research methodology. These phases are briefly described below:

1. **Review of the current state-of-the-art:** The main objective of this phase is to investigate the state of the art related to the field under consideration to identify problems and/or challenges. To achieve this, the related bibliography is used, reviewing publications from the scientific community published in journals, and proceedings of international congresses. The knowledge acquired in this phase should lead to a proposal to address the identified challenges.

Table 1.1: Publications in journals and conferences conducted during this thesis.

Title: Vertical and Horizontal Data Partitioning for Classifier Ensemble Learning.
Authors: AM Mohammed, E. Onieva, M. Woźniak.
Congress: The 11th International Conference on Computer Recognition Systems, 2019, Poland.

Title: Training set selection and swarm intelligence for enhanced integration in multiple classifier systems.
Authors: AM Mohammed, E. Onieva, M. Woźniak.
Journal: Applied Soft Computing (Impact Factor = 5.472 $\rightarrow$ Q1).
Status: Published.

Title: Selective Ensemble of Classifiers Trained on Selective Samples.
Authors: AM Mohammed, E. Onieva, M. Woźniak.
Journal: Neurocomputing (Impact Factor = 4.438 $\rightarrow$ Q1).
Status: Under review.

Title: An Analysis of Heuristic Metrics For Classifier Ensemble Pruning Based on Ordered Aggregation.
Authors: AM Mohammed, E. Onieva, M. Woźniak, G Martínez-Muñoz.
Journal: Pattern Recognition (Impact Factor = 7.196 $\rightarrow$ Q1).
Status: Under review.

---

2. **Design and development:** In this phase, a novel proposal to solve the identified challenges is designed and developed. To this end, previously acquired or updated knowledge is used to ensure that the solution is always up-to-date with the current state of the art.

3. **Experimentation and evaluation:** The goal of this phase is to test the proposals resulting from the previous step to a process of experimentation. To

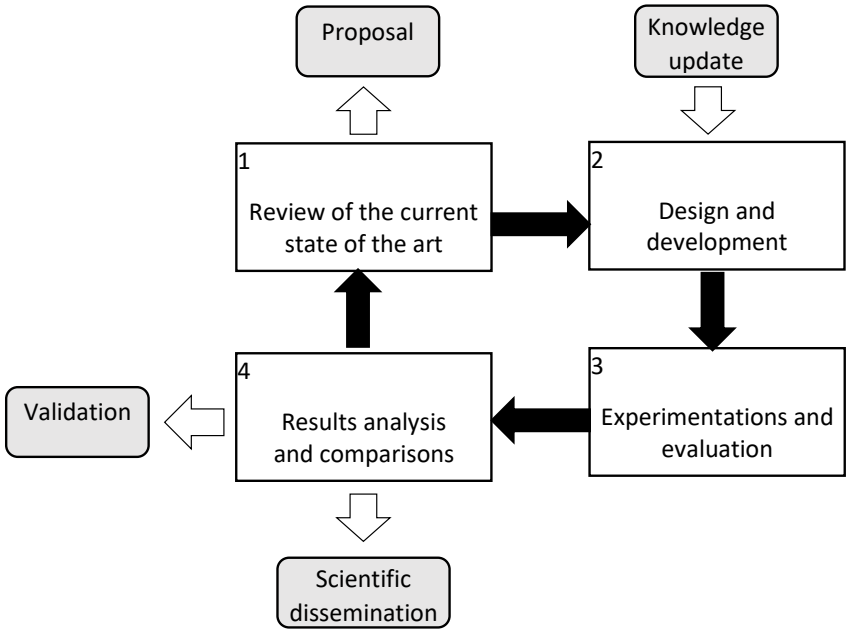


Figure 1.3: The research methodology of the thesis.

carry out this procedure, it is crucial to provide some criteria and evaluation methods with which the results will be compared in the subsequent phase. All these criteria and methods must be built using the knowledge acquired in the first stage of the methodology.

4. **Results analysis and comparison:** After carrying out experimentation, results must be analyzed and compared with those obtained in the state-of-the-art. At this point, it is needed to check if the results obtained are enough to address the challenges identified in the first phase. In such a case, another methodological cycle begins to approach the following challenge identified or to keep working with the challenge under consideration if it was not still solved. In this stage, conclusions must be drawn from analyses of results, and knowledge obtained must be materialized in scientific dissemination, either through journals, or conferences.

1.7 Structure of the Dissertation

The structure of the remainder of this thesis dissertation is outlined below.

Chapter 2 reviews background about concept drift, concept drift types, concept drift components, adapting types.

Chapter 3 reviews state-of-the-art concept drift, classifier ensemble selection, imbalanced data streams, Streams with Emerging New Classes (SENC), and transfer learning.

Chapter ?? presents our first proposal to build an effective proposed approch for handling Imbalanced Multi-class Drifted Data streams.

Chapter ?? provides our second proposal to adressing emerging new classes in incremental streams via concept drift techniques.

Chapter ?? presents our third proposal to addressing heterogeneous transfer learning problem in incremental streams via concept drift techniques.

Chapter 5 revisits the main goal and specific objectives posted earlier. In this chapter, we summarise the main contributions of this thesis and outline possible future research.

The good thing about science is that it's true whether or not you believe in it.

Neil deGrasse Tyson

CHAPTER

2

Background

Amidst the surge of vast streaming data, governments and businesses find themselves in an urgent need for sophisticated data analysis and machine learning analytics approaches. These tools are indispensable for anticipating future trends and making well-informed decisions. However, the perpetual emergence of new goods, markets, and consumer behaviors introduces a formidable challenge known as concept drift [24]. This phenomenon involves the variation of statistical parameters of the target variable over time in unexpected ways, posing a substantial obstacle to accurate forecasting and optimal decision-making. The patterns derived from historical data may become obsolete when applied to new and evolving datasets. The impact of concept drift extends across data-driven information systems, including decision support and early warning systems, diminishing their overall effectiveness. In the dynamic realm of big data, where data types and distributions are inherently unpredictable, the challenge of concept drift becomes even more pronounced. In response to this challenge, the field introduces a new subject: adaptive data-driven prediction/decision systems.

2.1 Concept Drift Sources

Concept drift, illustrated comprehensively in Fig. 2.2, unfolds through three distinct scenarios. First, (a) portrays a shift in data distribution, signifying changes in the underlying patterns and characteristics of the incoming data. This type of drift challenges the model's ability to adapt to new trends and patterns [25] [26] [27] [28]. Second, (b) showcases a change in function output, leading to the need for adjustments in the position of the class delimiter. Here, the relationship between input features and output classes undergoes transformations, demanding the model to realign its decision boundaries accordingly. Third, (c) presents a scenario involving a dual shift—both in data distribution and function output. This complex manifestation of concept drift requires the model to address simultaneous changes in underlying patterns and output relationships. Effectively navigating these shifts is crucial for maintaining the model's predictive accuracy and decision-making capabilities in dynamic environments. Understanding these nuanced aspects of concept drift is foundational for devising adaptive strategies in machine learning models.

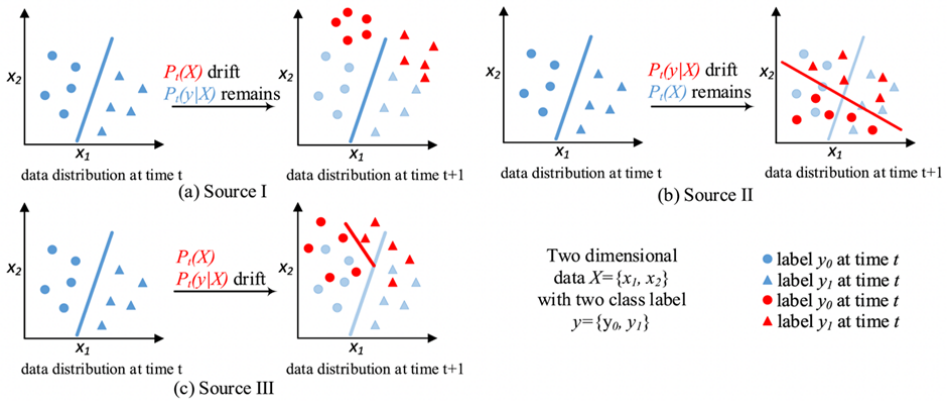


Figure 2.1: Concept .

Figure 2.2: Concept Drift Types.

2.2 Concept Drift Types

Concept drift is commonly categorized into four types, as illustrated in Fig. 2.4. In Types 1-3, the focus of research on concept drift adaptation revolves around

minimizing the drop in accuracy and achieving the fastest recovery rate during the transformation process of concepts. Conversely, Type 4 drift delves into leveraging historical concepts, emphasizing the identification of the best-matched historical concepts within the shortest time frame when a new concept emerges—whether suddenly, incrementally, or gradually. To illuminate the distinctions between these types, [27] introduces the term ”intermediate concept” to describe the transitional phases between concepts. As noted by [29], concept drift may not only occur at a precise timestamp but may also extend over a prolonged period. Consequently, intermediate concepts may emerge during the transformation, representing a blend of the starting and ending concepts in the case of incremental drift or embodying one of the starting or ending concepts, as observed in gradual drift. Understanding these intermediate concepts is essential for comprehending the nuanced dynamics of concept drift during transitions from one state to another.

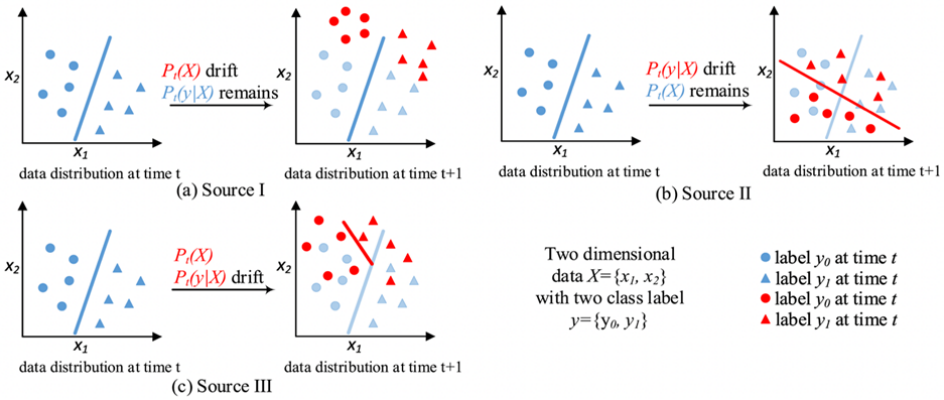


Figure 2.3: Concept .

Figure 2.4: Concept Drift Types.

2.3 Concept Drift Components

Conventional machine learning primarily involves prediction and training/learning. However, learning under the concept drift paradigm introduces three additional critical steps as show in Fig. 2.9: concept drift detection, drift understanding, and drift adaptation. Concept drift detection involves identifying changed points or change

time periods to define and predict drift. Drift understanding delves into crucial aspects such as when the drift starts, how long it lasts, and where it occurs, providing indispensable insights for the subsequent adaptation step. The adaptation step, also referred to as the reaction step, plays a pivotal role in updating current learning models in response to concept drift. Three main approaches address various types of drift: Simple Retraining, Ensemble Retraining, and Model Adjusting. Drift detection employs various tools and algorithms, comparing old and fresh data chunks with statistical models based on data distribution. Techniques vary, with some utilizing a constant chunk length and others employing a variable length. Drift understanding is essential for making well-informed decisions during the adaptation step. This involves calculating the necessary modifications in the trained model to adapt to new changes as shown in Fig. (2.3). The severity region determines whether to generate a completely new model or make minimal adjustments to the existing one.

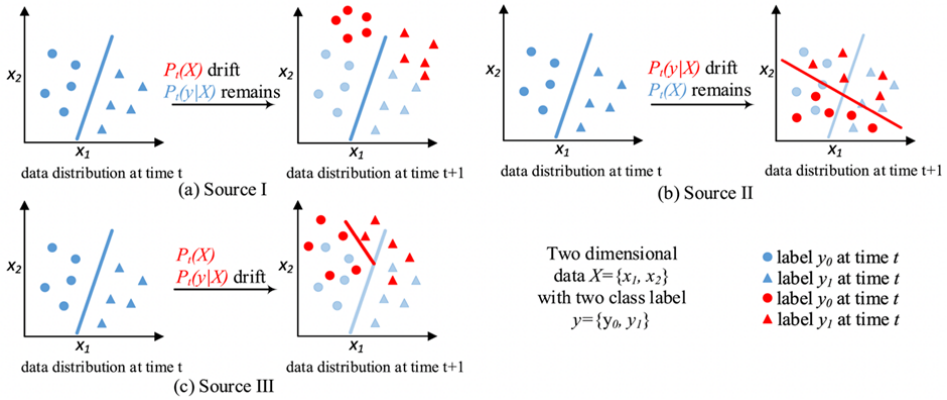


Figure 2.5: Concept .

Figure 2.6: Concept Drift Main Components.

2.3.1 Concept Drift Detection

Drift detection involves techniques and mechanisms to characterize and quantify concept drift by identifying change points or intervals [30]. The general framework for drift detection consists of four stages:

- **Stage 1 (Data Retrieval):** This stage focuses on retrieving data chunks from data streams. Given that a single data instance lacks sufficient information to infer the overall distribution [31], organizing data chunks meaningfully is crucial for effective data stream analysis [32].

- **Stage 2 (Data Modeling):** This optional stage abstracts the retrieved data, extracting key features that contain sensitive information impacting a system in case of drift. This stage may involve dimensionality reduction or sample size reduction to meet storage and online speed requirements [33].

- **Stage 3 (Test Statistics Calculation):** This stage involves measuring dissimilarity or estimating distance to quantify drift severity and generate test statistics for hypothesis testing. Defining an accurate and robust dissimilarity measurement remains a challenging aspect of concept drift detection. Test statistics can also be used for clustering evaluation [34] and to determine dissimilarity between sample sets [35].

- **Stage 4 (Hypothesis Test):** This stage employs a specific hypothesis test to assess the statistical significance of the change observed in Stage 3, such as the p-value. These tests determine drift detection accuracy by establishing statistical bounds for the test statistics from Stage 3. Without Stage 4, the acquired test statistics are meaningless for drift detection, as they cannot establish the drift confidence interval. Commonly used hypothesis tests include estimating the distribution of test statistics [36] [37], bootstrapping [38] [39], the permutation test [31], and Hoeffding's inequality-based bound identification [40].

It is crucial to note that without Stage 1, the concept drift detection problem can be regarded as a two-sample test problem, examining whether the populations of two given sample sets are from the same distribution [41]. In other words, any multivariate two-sample test can be adopted in Stages 2-4 for detecting concept drift [41]. However, in cases where distribution drift may not be included in the target features, the selection of the target feature becomes critical for the overall performance of a learning system and poses a significant challenge in concept drift detection [42].

2.3.2 Understanding Phase

The extent of concept drift severity serves as a valuable criterion for selecting appropriate drift adaptation strategies. As shown in Fig. **??**, in a classification task where the drift's severity is minimal, resulting in only a marginal shift in the decision boundary within the new concept, adjusting the current learner through incremental learning proves sufficient. Conversely, when confronted with high severity in concept drift, wherein the decision boundary undergoes substantial changes, it might be more effective to discard the old learner and opt for retraining a new one rather than incrementally updating the existing learner. It's noteworthy to mention that despite some researchers highlighting the capability to articulate and quantify the severity of detected drift, this information is not yet widely integrated into drift adaptation practices. The adaptation step offers three distinct ways to adapt the model. Simple Retraining involves training a new model using the most recent data, replacing the old model. Model Ensemble, the second approach, entails keeping and reusing existing models, which proves efficient when dealing with repeated instances of concept drift. The third approach, Model Adjusting, constructs a model that adapts flexibly from changed data, allowing partial updates when the original data distribution undergoes significant changes.

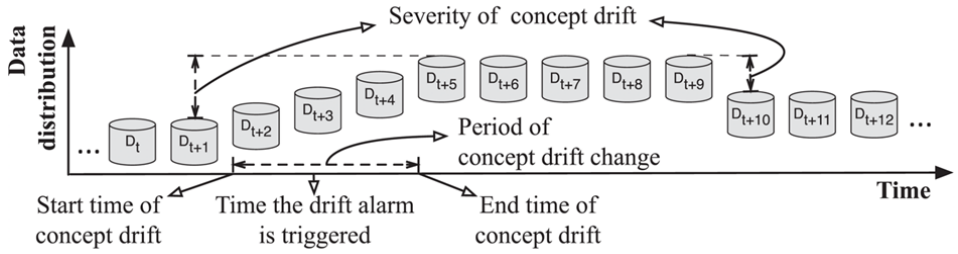


Figure 2.7: Understanding Phase of Concept Drift.

2.3.3 Adaptation Phase

This section delves into strategies for updating existing learning models in response to drift, referred to as drift adaptation or reaction. The three primary categories of drift adaptation methods are simple retraining, ensemble retraining, and model adjusting, each tailored to address specific types of drift.

A. **Simple Retraining**

One way to respond to concept drift is by retraining a new model with the latest data, replacing the outdated model as shown in Fig. (2.5). This approach requires an explicit concept drift detector to determine when to retrain the current model. A window strategy is commonly used, preserving recent data for retraining and/or utilizing old data for distribution change tests. An example of this strategy is Paired Learners [43], which employs two learners: the stable learner and the reactive learner. If the stable learner consistently misclassifies instances correctly classified by the reactive learner, indicating a new concept, the stable learner is replaced with the reactive learner. This method is straightforward, easy to implement, and adaptable at any point in the data stream. However, a trade-off arises when adopting a window-based strategy in determining an appropriate window size. A small window better reflects the latest data distribution, while a large window provides more data for training a new model. To address this challenge, the ADWIN algorithm [44] dynamically adjusts sub window sizes based on the rate of change between two sub-windows, eliminating the need for users to predefine window sizes. Beyond direct model retraining, researchers have explored integrating the drift detection process with the retraining process for specific machine learning algorithms. DELM [45], for instance, extends the traditional ELM algorithm, adapting to concept drift by dynamically adjusting the number of hidden layer nodes in response to increasing classification error rates, indicating a potential concept drift. Similarly, FP-ELM [46] introduces a forgetting parameter to the ELM model to adapt to drift conditions. A parallel version of the ELM-based method [47] has been developed for high-speed classification tasks under concept drift. OS-ELM [48], an online learning ensemble of repressor models, integrates ELM using an ordered aggregation (OA) technique to address the challenge of defining the optimal ensemble size. In the realm of instance-based lazy learners for handling concept drift, the Just-in-Time adaptive classifier [36] follows the ”detect and update model” strategy, extending the traditional CUSUM test [49] to a pdf-free form for drift detection. When a concept drift is identified, old instances (beyond the last T samples) are removed from the case base. Advancements include extending

this algorithm to handle recurrent concepts by considering and comparing the current concept to previously stored concepts [34] [36]. NEFCS [31], another KNN-based adaptive model, employs a competence model-based drift detection algorithm [31] to locate drift instances in the case base and distinguish them from noise instances. The redundancy removal algorithm, Stepwise Redundancy Removal (SRR), is developed to eliminate redundant instances uniformly, ensuring that the reduced case base retains sufficient information for future drift detection.

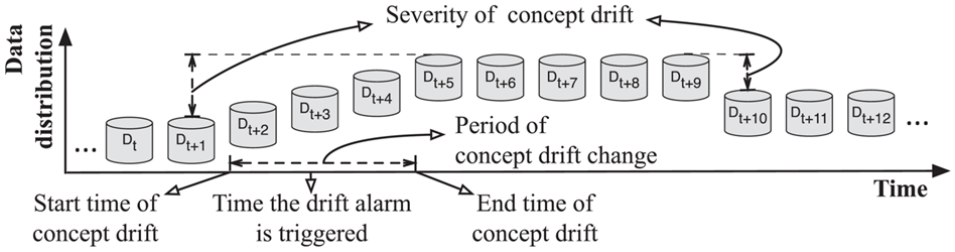


Figure 2.8: Retraining New Model Approach.

B. **Model Ensemble for Recurring Drift**

In recurring concept drift scenarios, preserving and reusing old models can be more efficient than retraining new ones for each recurrence, forming the basis for employing ensemble methods [50]. These methods, a focus in stream data mining research, consist of base classifiers with varied types or parameters. Their outputs, determined by specific voting rules, collectively predict new data. Various adaptive ensemble methods, extending classical ones or introducing adaptive voting rules, address the challenges of concept drift as shown in Fig. **??**. Classical ensemble methods like Bagging, Boosting, and Random Forests have been adapted for streaming data with concept drift. For instance, online Bagging [51] uses each instance once, simulating batch mode bagging. Leveraging Bagging [52] employs ADWIN drift detection to replace the existing classifier with the worst performance when concept drift is detected. Adaptive boosting [53], monitoring prediction accuracy through a hypothesis test, addresses concept drift, assuming classification errors on non-drifting data follow a Gaussian distribution. The

Adaptive Random Forest (ARF) algorithm [54] extends the random forest tree algorithm, incorporating concept drift detection (e.g., ADWIN) to decide when to replace an obsolete tree. A similar approach is seen in [55], using Beyond classical methods, novel ensemble methods with innovative voting techniques tackle concept drift. Dynamic Weighted Majority (DWM) [56] adapts to drifts through weighted voting rules, managing base classifiers based on individual and global ensemble performance. Learn++NSE [57] addresses frequent classifier additions by weighting them based on prediction error rates. Specific types of concept drift are considered in specialized ensemble methods. Accuracy Update Ensemble (AUE2) [58] equally addresses sudden and gradual drift, using a batch mode weighted voting ensemble method. Optimal Weights Adjustment (OWA) [59] achieves a similar goal with weighted instances and classifiers. Special cases, like class evolution, are considered in [60], while recurring concepts are handled by monitoring concept information [61] [62].Another method [63], refines the concept pool for recurring concepts.

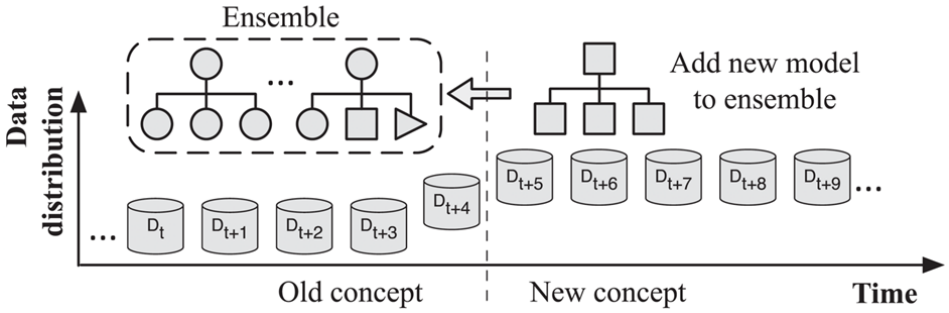


Figure 2.9: Ensemble Approach for Adaptation Phase.

C. **Model Ensemble for Recurring Drift**

As Shown in Fig. (2.7), instead of retraining the entire model, an alternative is to construct a model with adaptive learning capabilities, allowing partial updates in response to changing data distributions [64], as in Fig. 2.10. This is efficient when concept drift occurs in localized regions. Many techniques in this category use the decision tree algorithm, leveraging its ability

to adapt to individual sub-regions. VFDT [65] is a foundational contribution for high-speed data streams, employing the Hoeffding bound for node splitting. VFDT processes each instance once, doesn't store instances, and has minimal maintenance costs. CVFDT [66], an extension, addresses concept drift by maintaining a sliding window of the latest data and replacing the original sub-tree with a better-performing alternative. VFDTc [67] enhances VFDT by handling numerical attributes and adapting to concept drift with node-level detection. Later extensions [68] [69] introduce an adaptive leaf strategy in VFDTc, selecting the best classifier from options like majority voting, Naive Bayes, and Weighted Naive Bayes. Recent studies [70] [71] question VFDT's foundation, the Hoeffding bound, for non-independent variables in information gain. An alternative impurity measure is proposed in a new online decision tree model [72], demonstrating its reflection of concept drift and potential use in CVFDT. IADEM-3 [73] addresses Hoeffding bound concerns by computing the sum of independent random variables for drift detection and pruning.

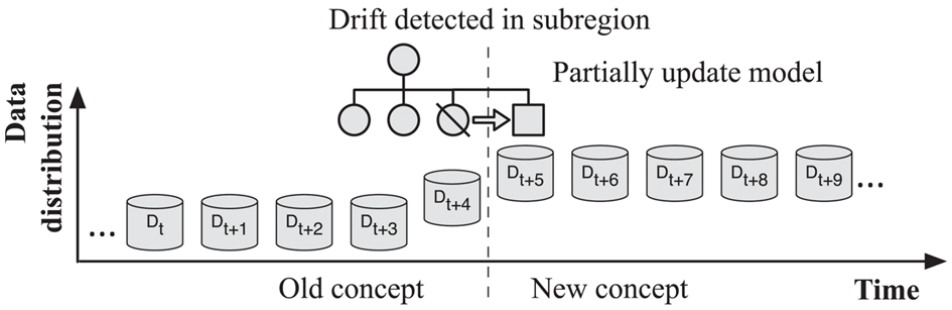


Figure 2.10: Partial Updating Approach for Adaptation Phase.

Nothing in life is to be feared, it is only to be understood. Now is the time to understand more, so that we may fear less.

Marie Curie

CHAPTER

3

State-of-the-art

In this chapter, we provide a comprehensive review of recent advancements in stream classification, addressing several critical challenges that arise in dynamic data environments. The rapid evolution of data streams introduces complexities such as concept drift, imbalanced multiclass scenarios, class overlap, classifier ensemble selection, emergence of new classes, and the incorporation of transfer learning. This introduction outlines the structure of the chapter and highlights the significance of each topic in advancing stream classification methodologies. The first section (Setion 3.1) focuses on concept drift, a phenomenon where the statistical properties of the data change over time. We explore various methodologies developed for real-time detection and adaptation to concept drift in streaming scenarios. This includes a review of techniques that enable systems to identify shifts in data patterns promptly, ensuring sustained classification accuracy. By analyzing the strengths and limitations of these approaches, we highlight the necessity for robust drift detection mechanisms in evolving data streams. Next, in Section 3.2, we delve into classifier ensemble selection in the context of streaming data. As the characteristics of data streams evolve, selecting the most appropriate ensemble of classifiers becomes crucial. We present algorithms designed to dynamically choose the best-performing classifiers based on the current data distribution. This section

emphasizes the importance of adaptive ensemble strategies in enhancing classification performance amidst changing data landscapes. Section 3.3 addresses the challenges posed by imbalanced multiclass scenarios, particularly in the presence of overlapping classes. We discuss specialized oversampling techniques aimed at countering the effects of class imbalance in streaming data. By providing insights into effective strategies for balancing the representation of minority classes, we lay the groundwork for developing robust frameworks that can maintain performance in imbalanced drifted streams. As new classes emerge in streaming systems, traditional classifiers often struggle to adapt effectively. In Section 3.4, we explore methodologies that specifically focus on the integration of new classes within existing frameworks. This section highlights innovative approaches that enhance the adaptability of stream classification systems, ensuring they remain effective as new data patterns emerge. In Section 3.5, we examine the role of transfer learning in stream classification. This section discusses how transfer learning techniques can leverage knowledge from related tasks to improve classification performance in dynamic environments. We explore current methodologies that facilitate the transfer of information across different domains, particularly in situations where labeled data may be scarce. The insights gained from this discussion will be pivotal for understanding how transfer learning can complement other strategies in our proposed framework. In Section 3.6,To further contextualize our discussion, we conduct a comparative analysis of recent works in the field, focusing on the recent works for each challange. We assess their contributions and identify limitations, illuminating specific gaps in the current research landscape that our work aims to address. Finally, Section 3.7 concludes this chapter by identifying key remarks and gaps, which inform the research direction for the subsequent chapters.

3.1 Concept Drift

In machine learning, concept drift refers to the phenomenon where the underlying data distribution changes over time [**? ? ?**] leading to a decrease in the accuracy or relevance of previously trained models. Detecting and responding to concept drift promptly is crucial for maintaining the performance of machine learning models.

To address this challenge, various concept drift detection methods have been proposed in the literature. One widely used method is the Drift Detection Method (DDM)[**?** **?**] which employs a statistical test to compare the error rate of a model on consecutive data sets. By identifying significant performance decreases, DDM signals the presence of concept drift. Another approach is the Early Drift Detection Method (EDDM)[**?** **?**] an extension of DDM that considers the error rate of a moving window of the latest data compared to the previous window. The ADaptive WINdow (ADWIN)[**?** **?**] approach uses a sliding window technique to monitor statistical differences between consecutive windows for drift detection. It dynamically adjusts the window size to adapt to changing drift patterns. The Kolmogorov-Smirnov windowing method (KSWIN) [**?**] calculates the Kolmogorov-Smirnov distance between two sliding windows to detect concept drift. Hoeffding's bounds with moving average test (HDDMA) and its variant HDDMW compute upper and lower bounds for the true mean of the data stream [**?** **?**] . By comparing these bounds, they can detect changes in the data distribution indicative of concept drift. Lastly, the Page-Hinkley [**?**] method measures the cumulative sum of errors and detects drift when the sum exceeds a predefined threshold. These concept drift detection methods are crucial in enabling machine learning models to adapt to the evolving nature of data streams. By continuously monitoring and detecting changes in data distributions, these methods facilitate the necessary adjustments and updates to maintain the model's performance and accuracy over time. Incorporating these methods into machine learning frameworks enhances their robustness and enables them to handle concept drift effectively.

3.2 Classifier Ensemble Selection

This study focuses on the overproduce-and-select approach for classifier ensemble selection methods [23] [88] [20]. The primary objective of classifier ensemble selection is to identify the optimal subset of classifiers from a larger ensemble, considering various criteria such as performance measures, diversity metrics, meta-learning techniques, and performance estimation approaches. This selection process aims to reduce computational complexity, enhance efficiency, and improve

overall ensemble performance, making it highly valuable for real-world applications. By carefully selecting a smaller subset of classifiers, ensemble selection strikes a balance between accuracy and computational resources, adapting to the evolving nature of the data stream. This approach leverages the strengths of different classifiers and adjusts the ensemble composition to handle changing conditions effectively. The goal is to enhance the accuracy, robustness, and overall performance of classification models in dynamic and challenging scenarios. There are two main approaches to the selection process: static and dynamic selection. Static selection assigns classifiers to specific partitions of the feature space, while dynamic selection chooses a classifier specifically for each unknown data sample based on its local competencies. Dynamic Ensemble Selection (DES) is a widely recognized approach that selects the best classifiers for each test instance, considering their competence within the local region of competence. The Randomized Reference Classifier proposed by Woloszynski and Kurzynski [21] stands out among various approaches. This classifier introduces randomness through beta distribution, enhancing adaptability and robustness. By considering the stochastic nature of class supports, the Randomized Reference Classifier can potentially improve classification performance in concept drift scenarios. However, it is important to note that employing diversity measures during the classifier selection process, as demonstrated by Lysiak [22], may lead to smaller ensembles but does not necessarily enhance classification accuracy. Overall, the overproduce-and-select approach for classifier ensemble selection methods offers a comprehensive framework for addressing the challenges associated with concept drift. By dynamically adapting the ensemble composition and leveraging the competencies of individual classifiers, this approach aims to improve classification performance, efficiency, and adaptability in dynamic and challenging scenarios.

3.3 Imbalanced data Streams

In the context of imbalanced data classification, as previously discussed, researchers have categorized three primary methodologies[**?**]. Our study primarily focuses on the first category, which pertains to addressing imbalanced data streams, particularly through sampling methods, specifically generating synthetic instances to rec-

tify the class imbalance. This process is commonly referred to as oversampling[**?**] and aims to balance instance quantities across both classes [**? ? ? ?**] . It's important to note that class imbalance can manifest in two scenarios: binary imbalanced classes, featuring skewed distribution between two classes, and multi-class imbalanced situations. Our study is specifically centered on multi-class oversampling techniques. In addressing class imbalances within multi-class contexts, Multi-Label SMOTE (MLSMOTE) [**?**] has extended the principles of SMOTE to cater to imbalanced multi-class learning scenarios. MLSMOTE significantly enhances classifier performance by generating synthetic examples for each minority class label. This approach thoughtfully considers neighboring examples in the feature space, ensuring that synthetic examples are appropriately assigned to their respective minority classes. Moreover, a recent advancement known as Multi-Label Synthetic Oversampling based on Local label imbalance (MLSOL) [**?**] has emerged as a promising technique. MLSOL systematically combats local imbalances within the domain of multi-class classification by employing distinct sampling strategies for each label. Empirical research confirms MLSOL's superiority by showcasing its capability to outperform existing methods in terms of classification accuracy and other evaluation metrics, particularly in effectively addressing local imbalance. Importantly, MLSOL offers a potentially effective approach to enhancing the performance of multi-class classification models, surpassing MLSMOTE in several aspects. Notably, MLSOL constructs synthetic samples exclusively from minority class instances within a restricted neighborhood, resulting in a more compact synthetic dataset compared to MLSMOTE. This attribute bears potential advantages for computational efficiency and helps mitigate concerns related to overfitting.

3.4 Streams with Emerging New Classes (SENC)

Existing approaches have been proposed to detect and handle the emergence of new classes in streaming data. Clustering-based methods, such as SACCOS [89], ECSMiner [90], and SAND [91], employ clustering techniques to identify new class emergence. However, these methods require access to true labels for either parts or all instances, limiting their practical applicability. Similarly, SENC-MaS [92]

uses matrix sketches for detecting emerging new classes but assumes the availability of true label information for all instances. In contrast, tree-based methods like SENCForest [93] and SEEN [94] utilize anomaly detection techniques to identify new classes, often with limited or no label information. However, these methods often suffer from high false positive rates and runtime inefficiencies. Another approach, SENNE [95], focuses on exploiting local information using the nearest neighbor ensemble for improved detection performance. Nevertheless, the absence of an effective model retirement mechanism in SENNE results in longer runtimes than alternative methods. The k-nearest Neighbor Ensemble-based method (KNNENS) [96] method emerges as a promising solution for the challenges of streaming emerging new class problems. By effectively utilizing a k-nearest neighbor-based hypersphere ensemble and incorporating model updates, the KNNENS approach tackles the issues of new class detection and known class classification within a unified framework. It is worth noting that an explicit limitation of existing methods is their lack of utilization of concept drift techniques for detecting emerging new classes and retraining the classification model. This limitation highlights the need for approaches that can effectively handle concept drift while addressing the emergence of new classes in streaming data.

3.5 Transfer Learning

In recent years, transfer learning has received significant attention due to its growing importance in addressing disparities between source and target domain distributions. Bridging the gap in distribution disparities is vital for optimizing performance in transfer learning, resulting in the development of diverse approaches, which can be broadly categorized into instance re-weighting and feature matching [97]. Instance re-weighting methods focus on aligning domain distributions by adjusting the weights of source instances, enabling the reuse of those source instances that closely align with the target domain. Notably, there's a strong emphasis on estimating these instance weights. For example, Huang et al. [98] introduced the Kernel Mean Matching (KMM) technique, which calculates weights by minimizing the mean differences between instances from the source and target domains within a Reproducing Kernel Hilbert Space (RKHS). Sugiyama et

al. [99] put forth the Kullback-Leibler Importance Estimation Procedure (KLIEP), utilizing Kullback-Leibler distance as a metric for assessing domain distribution dissimilarity, which includes a model selection step. Building on these instance reweighting methods, Sun et al. [100] introduced the 2-Stage Weighting Framework for Multisource Domain Adaptation (2SW-MDA) to address challenges in multisource transfer learning. It simultaneously adjusts the weights of source domains and their instances to reduce both marginal and conditional distribution disparities, akin to KMM, while leveraging the smoothness assumption for domain weighting. TrAdaBoost [101], a variation of the AdaBoost framework [102], operates by iteratively adjusting the weights of training data. In each iteration, it trains a classifier on a mix of source and target data and uses this classifier to make predictions on the training data. If a source instance is incorrectly predicted, its weight is reduced, diminishing its influence on the classifier. Conversely, the weights of misclassified target instances are increased to amplify their impact. An extension of TrAdaBoost, known as Multisource TrAdaBoost (MsTrAdaBoost) [103], is employed to address multi-source transfer learning challenges. MsTrAdaBoost combines each source and target dataset, training a separate classifier for each. Subsequently, it selects the classifier with the least error on the target data to update the instance weights. On a different note, feature matching aims to establish a shared feature representation space between source and target domains, which can be achieved through either symmetric or asymmetric transformations. A typical example of a symmetric transformation is the Transfer Component Analysis (TCA) method by Pan et al. [104], employing Maximum Mean Discrepancy (MMD) [105] [106] [107] to minimize differences in marginal distribution between source and target domains within an RKHS. Expanding on TCA, Joint Distribution Adaptation (JDA) [108] has been introduced to address both marginal and conditional distribution disparities. Recognizing the varying importance of marginal and conditional distribution differences across different problems, Wang et al. [109] [110] introduced the Balanced Distribution Adaptation (BDA) approach, which introduces a balancing factor. Subspace Alignment (SA) [111] focuses on aligning domain distributions in a lower-dimensional subspace, selecting crucial eigenvectors using principal component analysis [112] and learning a linear transformation matrix to minimize differences in eigenvectors between domains. In contrast, the Distribution Alignment

between Two Subspaces (SDA-TS) [113] was proposed to align both bases and distributions. Correlation Alignment (CORAL) [114], [115], an asymmetric transformation approach, is designed to align sub-space bases and employs second-order statistics. CORAL uses a learned transformation matrix to project source instances into the target domain. While there is a wide range of feature matching techniques in transfer learning, it is imperative to prevent the negative transfer, which occurs when transferred knowledge hinders the performance of target tasks. One of the reasons for negative transfer is the inclusion of unrelated or detrimental source samples in the target domain. To mitigate this, transfer joint matching (TJM) [116] introduces sparsity regularization in the feature transformation matrix, aligning features and re-weighting instances simultaneously. Zhong et al. [117] have also developed strategies to mitigate the impact of unrelated source instances and ensure positive transfer. To tackle the challenges posed by partial transfer learning scenarios, where the source domain contains more classes than the target domain, prior works [118] [119] [120] introduce instance-level re-weighting and class-level re-weighting mechanisms. These mechanisms are employed to reduce the influence of outlier classes from the source domains. Additionally, another approach presented in [3] utilizes an adversarial neural network to align domain distributions. In this method, lower weights are assigned to the source samples that are deemed distant from the discriminator. This weighting reflects the perception that such instances have weaker relevance to the target domain. Yang et al. [121] introduce the HE-CDTL approach for Concept Drift Transfer Learning (CDTL). HE-CDTL leverages knowledge from both source domains and historical time steps within the target domain to improve learning performance. Its key advantages include the utilization of the class-wise weighted ensemble for historical knowledge and the implementation of AW-CORAL for knowledge extraction from source domains. The class-wise weighted ensemble empowers individual classes in the current learning process to select historical knowledge independently. AW-CORAL serves to minimize domain disparities between source and target domains while mitigating negative knowledge transfer. Extensive experiments demonstrate that HE-CDTL outperforms baseline methods in addressing transfer learning challenges in the context of concept drift. Melanie addresses the challenge of non-stationary environments by considering an online scenario where data in both source and target domains are

generated. This method employs an online ensemble to learn models from each domain, subsequently combining these models using a weighted-sum approach. The models are trained incrementally, with their weights dynamically adjusted to handle concept drift. Generally, Melanie can be adapted to address CDTL by substituting the online learning ensemble with an ensemble designed for chunk-based concept drift.

3.6 Comparsion

In this section, we undertake a critical comparison of closely related works addressing the challenges of imbalanced multiclass streams 3.6.1, the emergence of new classes 3.6.2, and the integration of transfer learning 3.6.3 within streaming environments. The increasing complexity of real-world data streams necessitates advanced methodologies that can effectively manage the intricacies of these challenges. By examining various approaches in the literature, we aim to highlight their contributions, strengths, and limitations in dealing with imbalanced data distributions, adapting to new class occurrences, and leveraging transfer learning techniques. This comparative analysis not only sheds light on the current state of research but also underscores the specific gaps and unresolved issues that our work seeks to address, ultimately paving the way for more robust and adaptive solutions in the realm of streaming data classification.

3.6.1 Imbalanced Stream

In the context of multi-class classification, addressing class imbalances is crucial. Multi-Label SMOTE (MLSMOTE) extends the principles of SMOTE to generate synthetic examples for each minority class label, thereby enhancing classifier performance by considering neighboring examples in the feature space. Recently, Multi-Label Synthetic Oversampling based on Local label imbalance (MLSOL) has emerged as a more effective technique. MLSOL systematically addresses local imbalances by employing distinct sampling strategies for each label. Empirical research demonstrates that MLSOL outperforms existing methods, including MLSMOTE, in terms of classification accuracy and computational efficiency. By

generating synthetic samples exclusively from minority class instances within a restricted neighborhood, MLSOL produces a more compact and efficient synthetic dataset, mitigating concerns related to overfitting. As illustrated in Fig. 3.1, MLSOL is more likely to select x1 as a seed instance because it is surrounded by more neighbors of the opposite class for l3. MLSMOTE assigns the label vector [0,1,0] to all synthetic instances based on their neighbors. In contrast, MLSOL creates more diverse instances by assigning labels according to their location. Moreover, synthetic instances c2 and c3 generated by MLSMOTE introduce noise, whereas MLSOL copies the labels of the nearest instance to the new examples. In summary, MLSMOTE tends to generate new instances biased toward the dominant class in the local area, whereas MLSOL effectively explores and exploits both the feature and label space.

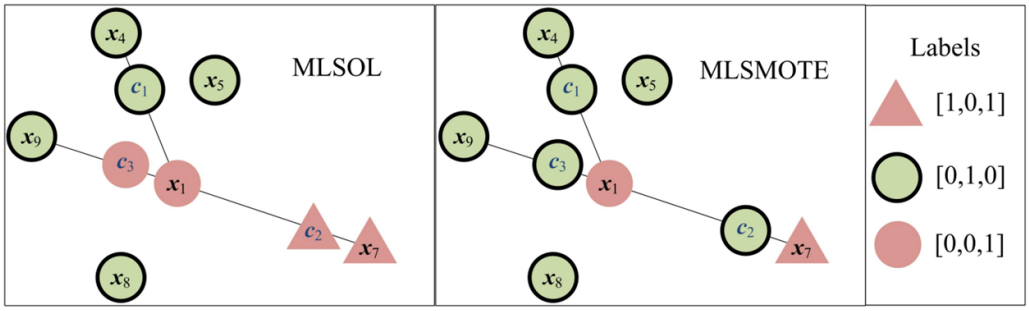


Figure 3.1: Distribution of the prediction accuracy.

Table 3.1 presents a comparison of two methods, MLSMOTE and MLSOL, which are designed to address the issue of imbalanced data in multi-class classification. MLSMOTE enhances classifier performance by generating synthetic examples for each minority class label, thereby balancing the class distribution. Its primary advantage is the generation of these synthetic examples, which helps mitigate the imbalance. However, it has significant limitations: the synthetic samples generated might be related to the majority class, which can blur the distinction between classes, and the method struggles with overlapping classes, leading to potential misclassification. On the other hand, MLSOL systematically combats local imbalances by employing distinct sampling strategies for each label within a restricted

neighborhood. This localized approach allows MLSOL to generate synthetic examples more precisely, addressing local imbalances effectively. Nonetheless, like MLSMOTE, MLSOL faces challenges with overlapping classes, which can result in misclassification in areas where class boundaries are not clear. Despite its advantages in handling local imbalances, the overlapping class issue remains a critical limitation for both methods, affecting their overall effectiveness in classification tasks.

Table 3.1: Comparison of MLSMOTE and MLSOL Methods

Method	Theory	Advantages	Limitations
MLSMOTE	MLSMOTE significantly enhances classifier performance by generating synthetic examples for each minority class label.	Generating synthetic examples for each minority class label.	• Random synthetic samples may be related to the majority class. • Overlapping classes.
MLSOL	MLSOL systematically combats local imbalances within the domain of multi-class classification by employing distinct sampling strategies for each label.	Generating synthetic examples for each minority class label within a restricted neighborhood.	• Overlapping classes.

3.6.2 Emergence of new classes

Effectively detecting and adapting to new classes in streaming data is crucial for maintaining classification accuracy. Tree-based methods like SENCForest and SEEN utilize anomaly detection but face high false positive rates and runtime inefficiencies. SENNE improves detection performance using a nearest neighbor

ensemble but suffers from longer runtimes due to the lack of an effective model retirement mechanism. The k-nearest Neighbor Ensemble-based method (KNNENS) addresses new class detection and known class classification using a k-nearest neighbor-based hypersphere ensemble and dynamic model updates. However, a critical limitation of these methods is their inadequate handling of concept drift, which is essential for detecting new classes and retraining the classification model. These methods represent the best closely related work for our proposal, which aims to build upon them by incorporating robust concept drift techniques for more adaptive and resilient classification systems.

Fig. 3.2 illustrates the SENCForest approach, which divides the space into three regions (normal, outlying, and anomaly) and detects emerging new classes (anomalies) using a calculated threshold path length. Fig. 3.3 depicts the SENNE algorithm, where hyperplanes are drawn in three dimensions (x1, x2, and x3) for each class (Fig. 3.3a). New instances are then classified as emerging or known classes based on the rank of each class (Fig. 3.3b). Fig. 3.4 presents the KNNENS algorithm, which draws hyperplanes for all class samples (Fig. 3.4a), and classifies new instances using a voting mechanism to determine if the instance is an emerging or known class (Fig. 3.4b). These visualizations highlight the operational differences between the SENNE and KNNENS algorithms in handling the classification of emerging and known classes.

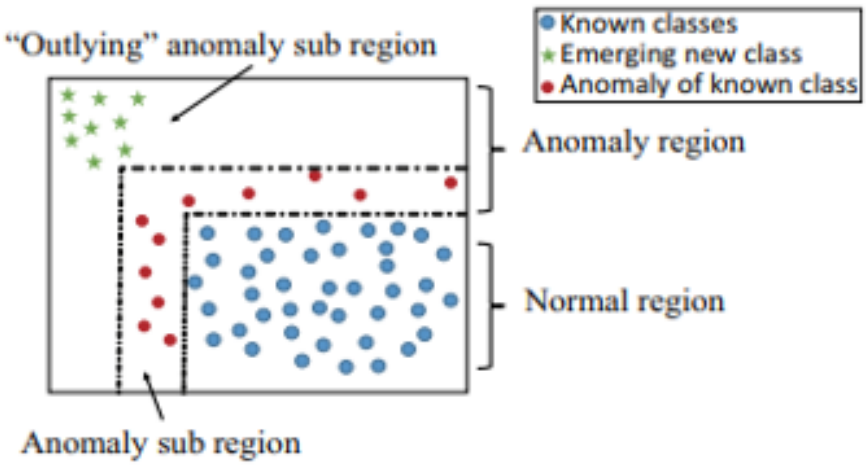


Figure 3.2: Stream Emerging New Class Overview

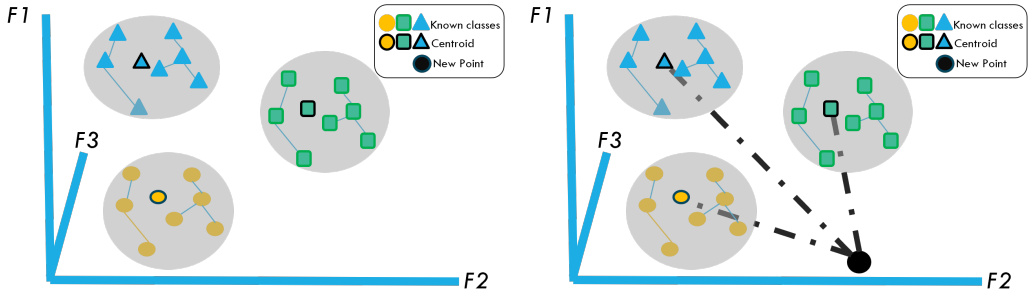


Figure 3.3: Stream Emerging Nearest Neighbor Ensemble (SENNE) Overview.

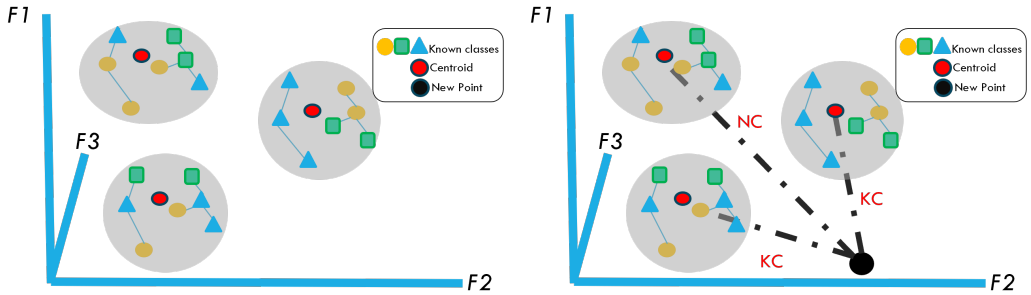


Figure 3.4: k-nearest Neighbor Ensemble-based (KENNE) Overview.

Table3.2 compares three methods for emerging class detection: SENCForest, SENNE, and KNNENS. SENCForest employs the anomaly detection method iForest for new class detection and uses a threshold path to identify anomalies, serving both as an unsupervised anomaly detector and a supervised classifier. However, it has a high potential for false positives and depends on a complex path length threshold. SENNE utilizes a nearest neighbor-based hypersphere of one class ensemble to explore local neighborhood information and sort distances, handling both low and high geometric distances between classes. Its limitations include the assumption that the distribution of known classes remains unchanged and it has lengthy update times. KNNENS employs a nearest neighbor-based hypersphere of all class

ensembles to explore local neighborhood information, reducing false positives for new classes without needing true labels for model updates. However, like SENNE, it assumes that the distribution of known classes remains unchanged.

Table 3.2: Comparison of SENCForst, SENNE and KENNE Methods

Method	Theory	Advantages	Limitations
SENCForst	employs anomaly detection method iForest [16] for a new class detection and then applies threshold path to detect the anomalies.	SENCForest serves as both an unsupervised anomaly detector and a supervised classifier.	• Potential for High False Positives. • Dependency on Path Length Threshold (more complexity).
SENNE	nearest neighbor-based hypersphere of one class ensemble to explore local neighborhood information and sort distance to calculate distance.	SENNE is able to handle both the low and high geometric distance between two classes in the feature space.	• Assumes that the distribution of known classes remains unchanged. • Take long time for update.
KENNE	nearest neighbor-based hypersphere of all class ensemble to explore local neighborhood information.	KNNENS to reduce false positives for the new class. KNNENS does not require true labels to update the model.	• Assumes that the distribution of known classes remains unchanged.

3.6.3 Transfer Learning

In the realm of transfer learning, three prominent methods—CORAL, Melanie, and HE-CDTL—serve as closely related approaches to our proposed method. Correlation Alignment (CORAL) is an asymmetric transformation approach that aligns sub-space bases using second-order statistics. By employing a learned transforma-

tion matrix, CORAL projects source instances into the target domain, thereby minimizing domain discrepancies and reducing negative knowledge transfer. Melanie addresses the challenge of non-stationary environments through an online ensemble learning approach. It incrementally trains models from both source and target domains, dynamically adjusting their weights to handle concept drift, and combines these models via a weighted-sum approach as shown in Fig. 3.5. As Shown in Fig. 3.6 This method can be extended to Concept Drift Transfer Learning (CDTL) by using an ensemble for chunk-based concept drift. HE-CDTL, designed explicitly for CDTL, leverages knowledge from source domains and historical time steps within the target domain to enhance learning performance. It utilizes a class-wise weighted ensemble for historical knowledge and implements AW-CORAL for extracting knowledge from source domains. The class-wise weighted ensemble allows individual classes to select historical knowledge independently, while AW-CORAL minimizes domain disparities and mitigates negative knowledge transfer. Extensive experiments have shown HE-CDTL to outperform baseline methods in addressing transfer learning challenges in the context of concept drift. Together, these methods provide a comprehensive framework for effective transfer learning in dynamic and evolving data environments.

Table3.3 compares three methods: CORAL, Melanie, and HE-CDTL. CORAL (Correlation Alignment) utilizes a learned transformation matrix and Singular Value Decomposition (SVD) to project source instances into the target domain, effectively minimizing domain discrepancy and reducing negative knowledge transfer. However, it faces challenges with non-stationary and heterogeneous data. Melanie (Multi-source Online Transfer learning for Non-stationary Environments) addresses online learning problems where data in source and target domains are generated from non-stationary environments. Its advantages include considering online problems but it too is limited by the complexities of online learning and data heterogeneity. HE-CDTL (Class-wise Weighted and Domain-wise Ensemble) minimizes domain shift by aligning second-order statistics of source and target distributions, leveraging historical knowledge to reduce disparities between domains. Despite its strengths, it relies on the quality of the source domain and also struggles with heterogeneous data.

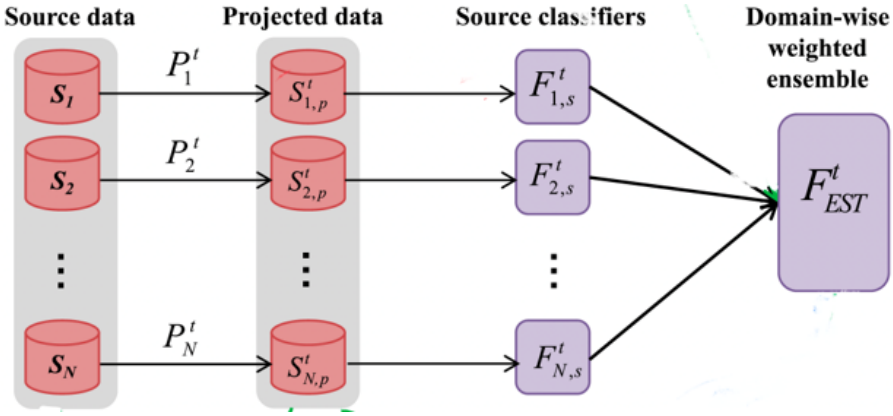


Figure 3.5: Coralation Alignment (CORAL) Overview.

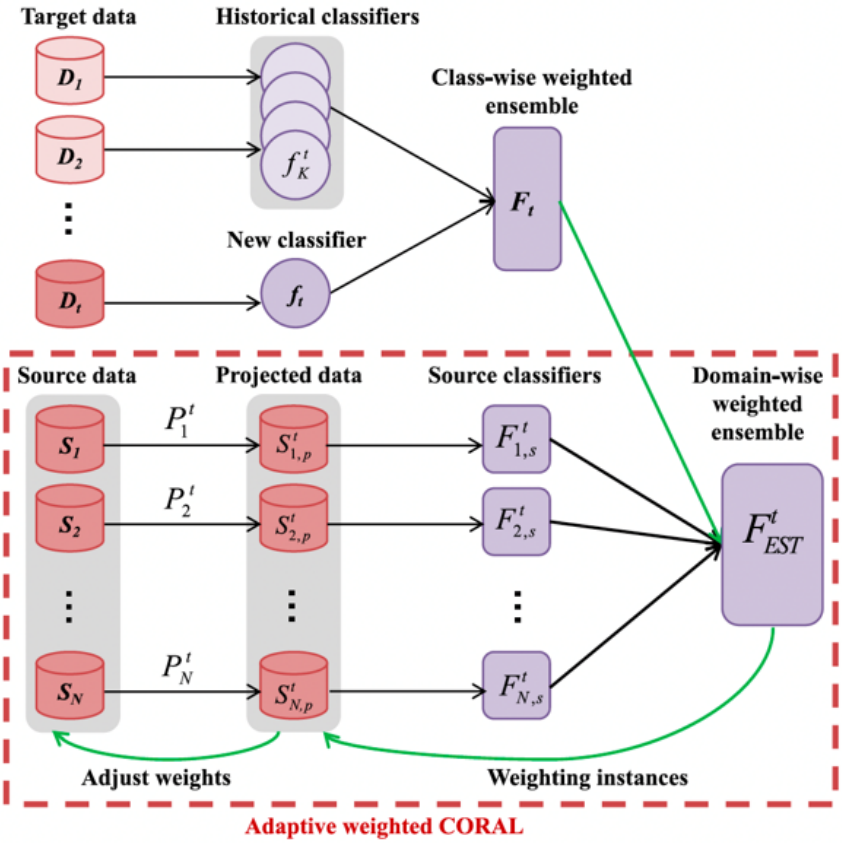


Figure 3.6: Concept Drift Transfer Learning (CDTL) Overview.

Table 3.3: Comparison of CORAL, Malanie and CDTL Methods

Method	Theory	Advantages	Limitations
CORAL	Correlation Alignment (CORAL) uses a learned transformation matrix and Singular Value Decomposition (SVD) to project the source instances into the target domain.	CORAL can minimize domain discrepancy across source and target domains, meanwhile reducing the negative knowledge transfer.	• Non-stationary environments. • Heterogenous multisource.
Melanie	Multi-sourcE onLine TrAnsfer learning for Non-statIonary Environments (Melanie). utilize the class-wise weighted .	It considers an online problem in which the data in source and target domains are generated from non-stationary environments.	• Online Learning based only. • Heterogenous multisource.
MLSOL	HE-CDTL uses the class-wise weighted and domain wise ensemble for historical knowledge and reduce the disparities between the source and target domains .	HE-CDTL minimizes domain shift by aligning the second-order statistics of source and target distributions.	• Depend on Source Domain Quality. • Heterogenous multisource.

3.7 Remarks

By comparing the literature on ensemble learning for classification tasks, the proposals in this thesis differ from other studies in several ways:

- As evident from our literature review on imbalanced streams, most studies have concentrated on generating synthetic samples while ignoring class overlap. *To address this challenge*, we propose an approach to generate non-overlapping classes in imbalanced streams.

- Oversampling techniques often perform inefficiently in the presence of concept drift. *To tackle this issue*, we introduce a methodology that selects the oversampling technique based on the current and historical distribution of the stream chunks.

- Our literature review on non-stationary environments reveals that most works focus on detecting emerging new classes while overlooking distribution changes. *To overcome this challenge*, we propose a combined approach utilizing Dynamic Ensemble Selection (DES) to select the best classifier for each chunk based on stream distribution, k-means clustering, and concept drift to address both emerging new class detection and distribution changes.

- In our literature review on transfer learning, we observed that most studies focus on homogeneous multisource transfer and neglect heterogeneous multisources in non-stationary environments. *To resolve this issue*, we propose a combined approach integrating Dynamic Ensemble Selection (DES), Concept Drift Transfer Learning (CDTL), eigenvector techniques, and concept drift to address heterogeneous transfer learning in non-stationary environments.

You cannot teach a man anything; you can only help him discover it in himself.

Galileo

CHAPTER

4

Dynamic Classification Ensembles for Handling Imbalanced Multiclass Drifted Data Streams

In recent years, the explosion of high-speed data streams has presented new challenges for machine learning models. Three critical issues that have emerged are concept drift, class imbalance, and class overlap. Concept drift refers to the phenomenon where the statistical properties of a data generation process change over time [1,2]. This signifies that the underlying concepts, relationships between variables, or data distribution can change, leading to a fundamental shift in the nature of data. Dealing with concept drift poses a fundamental challenge in machine learning and data mining. This can cause models trained on historical data to become inadequate when applied to new data affected by concept drift, leading to a decline in the model performance [2]. To address this issue, concept drift detectors are used to identify changes in data stream distributions by leveraging information associated with classifier performance or the incoming data items themselves. These signals frequently prompt model updates, retraining, or substitution of an

old model with a new one. In addition, class imbalance [3,4], which is characterized by uneven class distribution, poses a challenge for traditional classifiers [5], particularly in multiclass scenarios where minority class samples are at risk of misclassification owing to their limited representation [6]. Addressing imbalanced data classification requires specialized techniques to ensure accurate minority class classification without compromising the performance of the majority class [7–9]. This challenge becomes more daunting when minority class instances are scattered in unknown configurations, thereby increasing the likelihood of overfitting during the learning process. To address this issue, three primary methods are employed in the context of imbalanced data classification, which are effective in both binary and multiclass imbalance scenarios [10]. The first approach involves sampling methods that address class imbalance by either reducing the number of majority class instances (undersampling) or generating artificial minority class instances (oversampling) [11–13]. The second and third groups encompass adaptive algorithms and hybrid methods, respectively. Adaptive algorithms include one-class and cost-sensitive classification [14]. Hybrid methods merge data preprocessing with classification techniques, often utilizing ensemble classifiers to effectively mitigate class imbalance and enhance classifier performance [15–17]. Class overlap occurs when instances from different classes inhabit the same region in the data space [18,19]. This overlap complicates the task of distinguishing between representative instances of various classes and posing performance challenges for traditional classifiers. This issue is commonly referred to as class overlap. Researchers have proposed class overlap undersampling techniques to address class imbalance problems [20]. These techniques aim to leverage local similarities among minority instances to identify potentially overlapping majority instances. Although these methods have demonstrated promising results in improving model performance on specific datasets, many of them rely heavily on nearest-neighbor approaches to detect overlapping regions around minority instances. This approach often neglects the global similarity within the overlapping domain, which can lead to local optimal values getting stuck during calculations. Furthermore, determining appropriate parameters for these models poses a significant challenge. If the parameter selection is too extensive, it may lead to the exclusion of valuable instances, while conservative parameters may overlook instances with overlap. This issue

can significantly affect classifier performance, particularly when handling streams containing both a minority class and instances with class overlap. Therefore, both class imbalance and class overlap present significant challenges in the realm of data stream analyses. Consequently, addressing class imbalance is crucial in multiclass learning, leading to research efforts that focus on both concept drift and class imbalance challenges. Researchers have explored techniques such as DES and multiclass oversampling to address these issues. Dynamic classifier ensembles offer the unique ability to adapt their composition based on data characteristics, making them valuable in situations with evolving data conditions [21]. The aim of classifier ensemble selection is to identify the optimal subset of classifiers from a larger ensemble. A prominent approach in classifier ensemble selection is the overproduce-and-select strategy. This selection process is guided by diverse criteria, including individual performance measures, diversity metrics, meta-learning techniques, and performance-estimation approaches. Such optimization is particularly crucial in scenarios in which striking a balance between accuracy and computational resource constraints is paramount. There are two distinct approaches: static and dynamic approaches. Static selection involves assigning classifiers to predefined feature partitions, whereas dynamic selection adaptively selects classifiers based on their competency [22]. Dynamic selection offers two choices: Dynamic Classifier Selection (DCS) and Dynamic Ensemble Selection (DES). DCS algorithms enable the selection of the most appropriate classifier for each data point, based on its local competencies. In contrast, DES focuses on selecting the optimal classifiers for each instance based on their competence within localized regions [23–25]. Competency assessment relies on a Dynamic SELection dataset (DSEL) containing labeled samples. Moreover, innovative techniques, such as the randomized reference classifier, introduce randomness into class supports to enhance adaptability in addressing the challenges related to imbalanced data. The main goal of this study is to formulate a precise classification approach that addresses changing conditions. Specifically, the proposed approach aims to address scenarios where there is an uneven distribution among several classes, overlapping instances of classes, and instances where the fundamental concept of data evolves. To address these challenges, we employed dynamic classifier ensembles. These ensembles utilize

oversampling techniques, implemented either on a global or local scale, as a preprocessing step to address class imbalance. Furthermore, we enhanced multiclass learning techniques to counteract class imbalance through method adaptation.

The remainder of this chapter is organized as follows: In Section 4.1, we present the motivations and the contributions. The proposed framework and combination via SI algorithms are discussed in detail in Section **??**. The experimental results and the discussion are presented in Sections 4.3 and **??**, respectively. Finally, the conclusions of this study and future research are discussed in Section 4.4.

4.1 Motivations and Contributions

1. We introduce a classification approach that dynamically adjusts to multiclass imbalanced data, incorporates mechanisms for detecting concept drift, and optimizes classifier ensemble selection. The objective is to enhance the classification accuracy, specifically for multiclass imbalanced nonstationary streams.
2. Additionally, we propose an adaptive method for the class imbalance issue, considering the data distributions and historical instances of class imbalance. This is particularly relevant in cases where class overlap occurs within the multiclass and drifted datan performance by selecting the most suitable oversampling method based on the unique characteristics of the data stream.

4.2 Proposed Methodology

In this section, we present the primary phases of our study, comprising three distinct stages. Following this introduction, we delve into the synthetic data-generator method, providing a comprehensive breakdown of the four essential steps. Our proposal aims to develop a robust approach designed to overcome the challenging domain of multiclass classification for imbalanced and drifting data streams. In pursuit of this goal, our proposal addresses the four primary challenges inherent in constructing our approach.

- **Multiclass Imbalanced Streams:** This study focuses on addressing the widespread problem of imbalanced data streams in the context of multiple classes. We

aim to address this issue using well-known techniques, notably MLSMOTE and MLSOL.

- **Class Overlap:** We also address a critical challenge involving class overlapping, a factor known to substantially affect model performance [23] [24]. To address this issue, we introduce an adaptive method that generates nonoverlapping synthetic instances, thereby enhancing the overall performance of the model.

- **Drifted Streams:** Our proposed approach integrates a concept drift detector to identify shifts in the underlying data distribution. This dynamic detection mechanism enables the model to promptly recognize changes and adjust its classifiers, thereby ensuring its effectiveness in handling drifting stream.

- **Classifier Performance:** To enhance the classifier performance, our approach employs Dynamic Ensemble Selection (DES). This technique creates a pool of classifiers and dynamically selects the most suitable classifier for each incoming data point, further improving classification accuracy and robustness.

4.2.1 Approach Overall Details

Our proposed approach is designed with three distinct phases that work together to improve its performance in managing multiclass imbalanced and drifting data streams.

- **DES Phase (dynamic ensemble selection phase):** The first phase, known as the dynamic ensemble selection (DES) phase, is responsible for selecting the most appropriate classifier for the incoming data. This ensures that the selected classifier is well-suited for the current data chunk.

- **Drift detector phase:** The second phase of our approach is the drift detector phase, which operates in real-time to continuously monitor the data stream. Its primary function is to identify any signs of concept drift, which indicates shifts in the underlying data distribution over time.

- **Synthetic data generator phase:** The final phase of our approach is the synthetic data generator phase, which is dedicated to generating synthetic data for the minority classes. This step is crucial for addressing class imbalance by producing additional samples for underrepresented classes, thereby significantly enhancing the model's ability to accurately classify instances from minority classes.

Specifically, as shown in Fig. 4.1, the DES phase retrieves the current data chunk from the stream and applies the DES technique to select the most suitable classifiers for the received chunk. The selected classifiers were then passed to the second phase, where they were employed to predict the class of each instance within the received data chunk. Simultaneously, detectors like ADWIND or DDM are employed to monitor any occurrence of concept drift. If the discrepancy between the class frequency and standard deviation of the current chunk is significant, as described in reference [37], and the imbalance ratio exceeds the average imbalance ratio, the current chunk is forwarded to the third phase, as indicated by the red rectangle in Fig. 1. In the third phase, our proposed method uses a set of equations 4.14.24.3 to identify minority classes. The first equation calculated the frequency of each class in the current chunk. The second equation determines the optimal frequency for each class based on the size of the chunk and the number of classes in the current chunk. Finally, the third equation designates the classes as minority classes if their frequency differs significantly from the standard deviation of the current chunk. As shown in Fig. 4.2, phase These identified minority classes are then fed into the synthetic data generator phase, which increases the minority class samples to balance any imbalanced chunks. This ensures optimal performance for the new classifiers. Algorithm 2 provides a comprehensive outline of the process of the proposed approach, which is the main contribution of our study and is designed to effectively address multiclass imbalanced and drifting data streams, uses streaming data as input, and systematically executes each step within the approach. The outcome of this process is the classification prediction generated using the proposed approach.

4.2.2 Synthetic Data Generator

Fig. 4.2 presents a comprehensive overview of the synthetic data generator phase, which is an essential component responsible for generating synthetic samples by considering both data distribution and historical chunk behaviors. This phase has several advantages and can perform a wide range of tasks.

- **Similar chunk analysis:** Initially, the phase analyzes the current chunk distribution and identifies a similar chunk from historical data. This analysis forms the basis for generating synthetic samples that align with prevailing distribution patterns.

- **Oversampling method selection:** This phase utilizes the knowledge of the oversampling technique applied to the identified similar chunk. Consequently, it employs an alternative oversampling technique, using MLSMOTE and MLSOL, to create the most effective synthetic data. This step is designed not only to optimize the current classification but also to preemptively address potential drifts in similar future chunks.

- **Class overlap validation:** This step involves generating synthetic samples for the minority classes to effectively address the issue of minority classes and consequently enhance the classifier performance. The process utilizes the K-Nearest Neighbor (KNN) algorithm, as applied in [25], to identify overlaps between newly generated data instances and existing instances. If overlaps are detected, the proposed approach iteratively removes these samples because their presence can potentially diminish the overall classifier performance [23] [24]. Consequently, the proposed approach generates alternative samples to address this challenge and preserve the primary objectives of the synthetic data generator step.

- **Continuous refinement:** This process iterates until it successfully generates high-quality synthetic data that aligns with the data distribution, minimizes overlap, and mitigates potential concept drift. The generated data were subsequently utilized in the training phase to improve classifier accuracy.

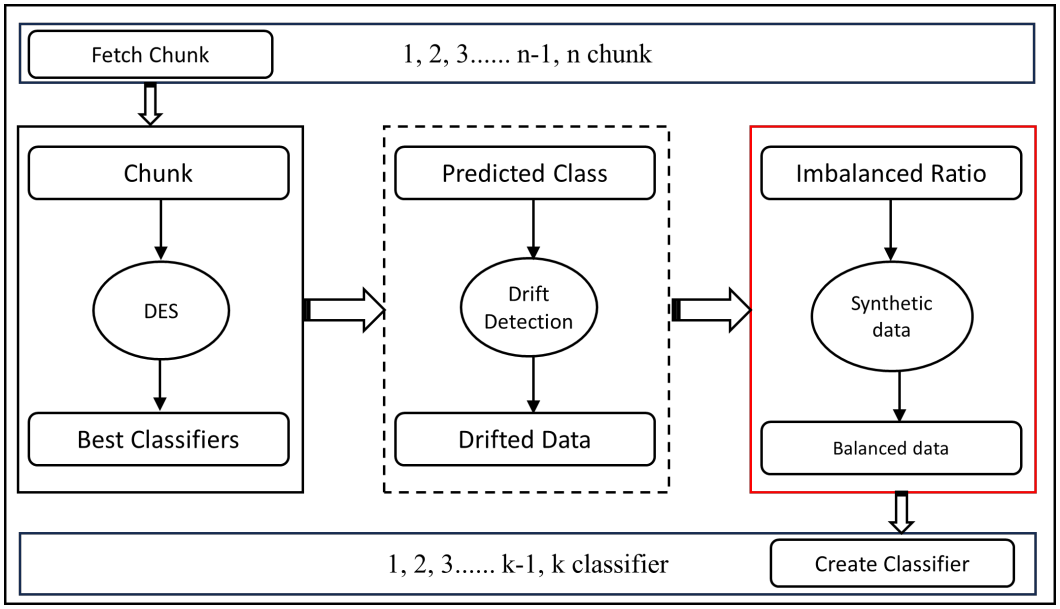


Figure 4.1: Proposed Approch Flow.

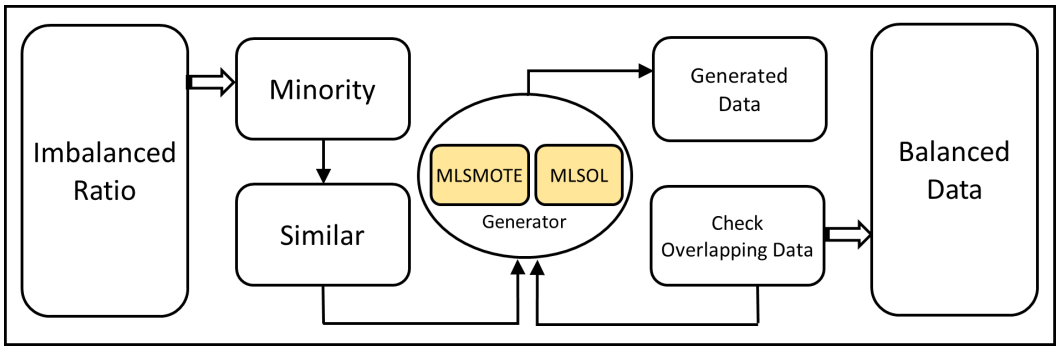


Figure 4.2: Synthetic Data Generator Flow.

$$frq_c = \sum_{i=1}^{\text{chunk size}} \begin{cases} 1, & \text{if } y_i = c \\ 0, & \text{otherwise} \end{cases}, \quad i = 1, 2, 3, \ldots \text{chunk size} \tag{4.1}$$

$$\text{best } freq_n = \frac{|n|}{|C|} \tag{4.2}$$

$$\text{classes type}_{\text{chunk}} = \sum_{c=1}^{C} \begin{cases} \text{Minority}, & \text{if } diff(sd_c - frq_i) > \text{best } freq_{\text{chunk}} \\ \text{Majority}, & \text{otherwise} \end{cases}, \quad c = 1, 2, 3, \ldots C \tag{4.3}$$

Algorithm 1: Proposed Framework Algorithm for Imbalanced Multi-Class Drifted Data Streams

Input: data stream, maximum classifiers pool size κ
Output: Prediction P

$\psi, \Psi, \Omega, \mu \leftarrow \emptyset$;
$\omega \leftarrow 0$;
for *stream have chunk* **do**
  if a *is the First chunk* **then**
    $k \leftarrow$ `trainingNewClassifier`(a);
    $P \leftarrow$ `getPrediction`(a, k);
  else
    $k \leftarrow$ `DES`(a, Ψ);
    $P \leftarrow$ `getPrediction`(a, k);
    $\psi \leftarrow$ `conceptDriftDetector`(P);
    if $\psi > 0$ **then**
      $\Omega \leftarrow$ get classes frequency according to Eq.1;
      $\omega \leftarrow$ best frequency according to Eq.2;
      $\mu \leftarrow$ get minority classes according to Eq.3;
      $b \leftarrow$ utilize a and μ to get the synthetic data according to Algorithm 2;
      trainingData $\leftarrow a + b$;
      $k \leftarrow$ `trainingNewClassifier`(trainingData);
      $\Psi \leftarrow \Psi + k$;
      if $\Psi > \kappa$ **then**
        `removeWorstClassifier`(Ω);
    $P \leftarrow$ `getPrediction`(a, k);
return P

In Algorithm 2, also known as the Synthetic Data Generator, we input three essential elements: the minority class samples, the current data chunk, and the desired size for generating synthetic data. The primary objective of this algorithm is to produce synthetic data samples. To achieve this, we employ the KNN algorithm to identify any overlapping instances within the current chunk (Line 3). This algorithm utilizes two specific techniques, MLSMOTE [37] and MLSOL [33]. MLSMOTE was chosen for its introduction of randomness during instance generation, reducing its reliance on the local characteristics and distribution of the mi-

nority class. This randomization diminishes the likelihood of producing overlapping instances, particularly in cases where minority class instances are situated in overlapping regions. In contrast, MLSOL considers the local behavior of minority classes, resulting in synthetic points that closely resemble the minority class. This approach significantly improves the accuracy of the classifier (lines 4-10). Additionally, in this algorithm, specifically from lines 11 to 17, these lines are dedicated to generating synthetic instances that ensure non-overlap with existing classes. This procedure depends on the selected oversampling method and utilizes the KNN algorithm to guarantee that the generated instances do not overlap with the existing ones.

Algorithm 2: Synthetic data generator

Input: Minority classes μ, current chunk a, sample size η, historical chunks h
Output: Generated data b
$b \leftarrow \emptyset$;
$f \leftarrow$ MLSMSOTE;
$knn \leftarrow$ kNearestNeighbor(a);
$chunk \leftarrow$ similarChunk(a, h);
$f \leftarrow$ similarChunkOverSamplingMethod$(chunk)$;
if $f =$ *MLSMSOTE* **then**
  $f \leftarrow$ MLSOL;
else
  $f \leftarrow$ MLSMSOTE;
while $|b| < \eta$ **do**
  $p \leftarrow$ generateSyntheticPoint(μ, f);
  $similarPointsClass \leftarrow$ KNN.getKneighbor(b);
  if $similarPointsClass = \mu$ **then**
    $b \leftarrow b \cup \{p\}$;
return b;

4.3 Experimental Results

The objective of the experiments conducted in this study was to evaluate the efficacy of multiclass oversampling techniques in enhancing the performance of the

proposed method in imbalanced drifted multiclass classification streams. Our primary goal was to develop a novel approach that combines Dynamic Ensemble Selection (DES) to improve classification accuracy and robustness in such streams. These experiments yielded valuable insights that could further refine the performance of the proposed approach and its ability to effectively handle imbalanced data streams. These findings provide a better understanding of the capabilities of the proposed approach and offer insights into an optimal strategy for tackling minority class issues and concept drift in imbalanced data streams. This study contributes to the advancement of stream mining techniques for generating more accurate and robust classification models in dynamic data stream environments. By addressing the challenges posed by minority classes and concept drift, this study offers valuable insights for improving the performance of the proposed approach and enhancing the overall efficiency of stream mining. The two main questions to be answered are:

- Q_1: What is the impact of reduced and consistent data on the performance of ensemble learning?
- Q_2: Is it possible with the search capability of swarm intelligence to enhance the combination of classifiers?

4.3.1 Experimental setup

The evaluation of the proposed method incorporated the utilization of various metrics such as recall, precision, specificity, f1 score, balanced accuracy score (BAC), and geometric mean score (G-mean) [38]. The experimental protocol utilized for evaluation was the test-then-train approach [39], where the classification classifier was trained on a specific data chunk and subsequently evaluated on the subsequent one. The chunk size was standardized for all utilized data streams to 2,000 instances. We employed four classification classifiers as base estimators: K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Gaussian Naive Bayes (GNB), and Hoeffding Tree (HT), as implemented in scikit-learn [40]. A pool of classifiers was constructed with a maximum size of L = 8, where the DES selected the best classifier for each chunk. If the pool surpassed the set threshold (L), the classifier

with the lowest performance was eliminated. The experiments were conducted using Python programming language, and the source code was publicly available on GitHub . We conducted a comparison between multiclass oversampling techniques (MLSMOTE and MLSOL) and our proposed approach to demonstrate the effectiveness of our contribution. Additionally, we conducted these experiments using two different concept drift detectors, ADWIN [28] and DDM [27], to demonstrate the adaptability and robustness of our proposed approach across varying drift detectors.

4.3.2 Data Streams

In this study, the proposed approach was assessed using various datasets including benchmark datasets, a real application stream dataset, and synthetic data streams. The Stream-learn Python library was used to conduct the evaluations [41]. Table 4.1 illustrates the benchmark dataset employed in this study, which consists of the Covertype dataset containing 40 features, seven classes, and 581,010 instances. For real application stream evaluation, the Sensor stream dataset was used, which consisted of five features, 58 classes, and 392,600 instances. This represents a real-world application scenario and provides valuable insights into the performance of the proposed approach in practical settings. Synthetic datasets were generated using Scikit-learn Python library to evaluate the performance of the proposed approach. The synthetic dataset was designed to simulate data streams and comprised 10 features and four classes divided into 200 chunks of 2,000 instances each. The performance of the proposed approach was systematically evaluated using these datasets and a stream-learn library. These evaluations provided insights into the effectiveness of the proposed approach in handling different types of data streams, including benchmark datasets, real application streams, and synthetic data streams.

Dataset	Number of Features	Number of Classes	Number of Instances
Covertype dataset[2]	40	7	581,010
Sensor Stream dataset[3]	5	58	392,600
Synthetic stream	8	3	200,000

Table 4.1: Characteristics of the datasets used in the experimentation.

4.3.3 Analysis of Experimental Results

The performance of the proposed framework was comprehensively assessed on multiple data streams, considering two distinct concept drift detectors, ADWIN and DDM. To ensure thorough evaluation, six key performance metrics—F1 score, recall, precision, G-mean, specificity, and balanced accuracy—were carefully presented using two visualization diagrams: radar and line. A radar diagram was strategically utilized to provide an overview that effectively depicted the performance of each algorithm across the six metrics. The mean value of each metric was calculated to present the overall performance of each method (MLSMOTE, MLSOL, PA). It is important to note that the PA is represented by red lines.

4.3.3.1 Results on the Benchmark Stream

The results of the mentioned methods (MLSMOTE, MLSOL, PA) applied to the Covertype dataset are presented in Fig. 4.3 , utilizing ADWIN as the drift detector. The radar diagram shows that the metric values were between 0.8 and 1.0. MLSOL exhibited the highest precision, whereas MLSOTE and PA had nearly identical values. However, PA excels in other metrics, whereas MLSMOTE has the lowest values. The line diagram in Fig. 4.3 shows the classification accuracy across 100 data chunks using the specificity metric for the methods in each chunk. Notably, all methods exhibit suboptimal accuracy during the first 20 chunks. However, a noticeable improvement was observed beyond this initial phase. The key factor driving this improvement was the expansion of the classifier pool, which now encompasses a growing number of classifiers. This expansion enables the Dynamic Ensemble Selection (DES) technique to become more proficient in selecting the most suitable classifier for each incoming chunk. Consequently, accuracy experienced a significant boost in later chunks, reflecting the adaptability and effectiveness of the ensemble approach. From chunk 20 to the last chunk, PA achieved the highest accuracy, whereas MLSMOTE recorded the lowest accuracy. PA's superior performance of the PA is credited to its use of historical chunks to generate optimal nonoverlapping samples, thereby effectively training the pool classifiers. In Fig. 4.4 , the same dataset is employed with the DDM functioning as the drift detector. The radar plot illustrates results comparable to those of the prior experiment,

whereas the line diagram underscores PA's supremacy in most chunks. However, it also reveals that all approaches demonstrate diminished performance, in contrast to Fig. 4.3 (ADWIN), suggesting that ADWIN surpasses DDM when utilized in the Covertype data stream.

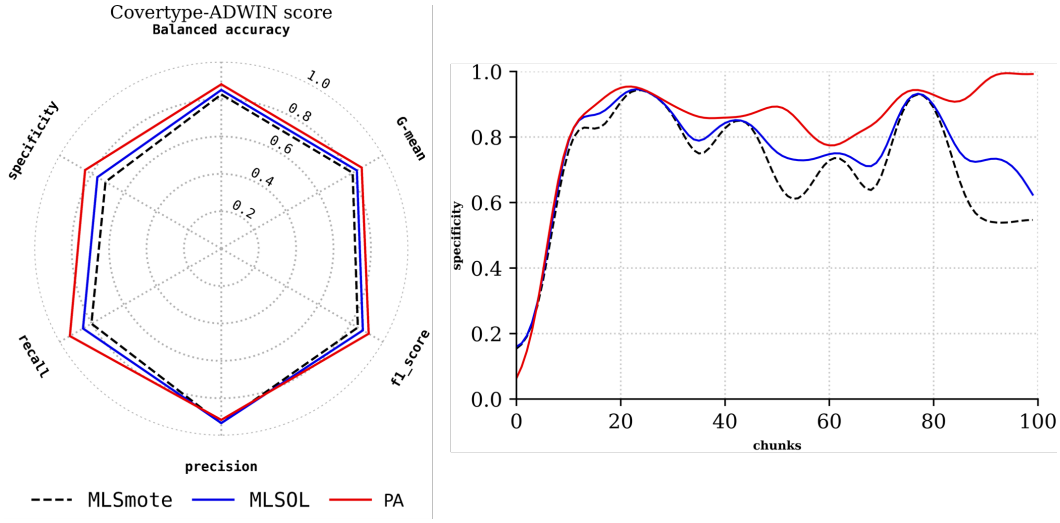


Figure 4.3: Synthetic Data Generator Flow.

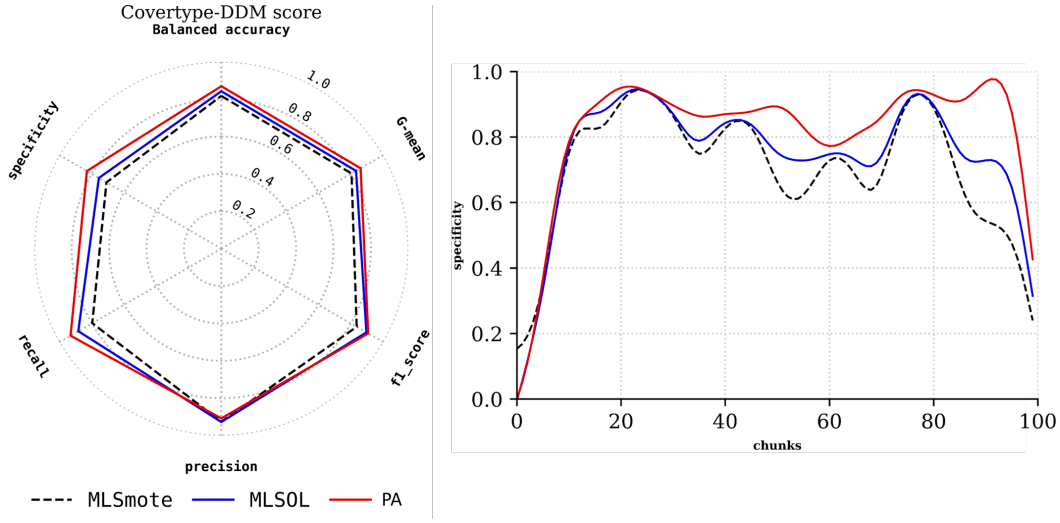


Figure 4.4: Synthetic Data Generator Flow.

4.3.3.2 Results on the Real Application Stream

Fig. 4.5 illustrates the outcomes of employing the three methods on the Sensor data stream using ADWIN as the drift detector. The radar graph depicts metric values ranging from 0.6 to 0.8, which indicate the pronounced drift and imbalance of the Sensor stream. In terms of the precision and recall metrics, the three methods exhibited almost identical values, whereas PA stood out in the other metrics. In contrast, MLSMOTE and MLSOL displayed similar values. By examining the line graph in Fig. 4.5 , it is evident that during the initial 30 chunks, PA's performance of PA might be suboptimal in some instances because of the limited number of classifiers in the pool. However, after the first 30 chunks, the performance significantly improved with the addition of more classifiers. In the same graph, PA consistently achieved the highest performance across all chunks, whereas MLSMOTE exhibited lower performance in certain chunks, and MLSOL exhibited lower performance in the other chunks. In Fig. 4.6, using the same dataset with the DDM as the drift detector, the radar plot metrics indicate lower results compared to the previous experiment. Nonetheless, the line graph underscores PA's dominance of PA in most chunks, although all methods exhibit nearly identical performance across the entire set of chunks. Overall, the performance of all the methods in Fig. 4.6 is inferior to that in Fig. 4.5 (ADWIN), which highlights ADWIN's superiority of ADWIN over DDM when applied to the Sensor data stream.

4.3.3.3 Results on the Synthetic Stream

The results of applying the same methods on the synthetic data stream using ADWIN as the drift detector are presented in Fig. 4.7. The radar diagram indicates metric values ranging from 0.6 to 0.8, suggesting that the synthetic stream is prone to frequent drifts. While MLSOL exhibited the highest precision, MLSOTE exhibited the lowest values. Conversely, PA performed well in other metrics, and MLSMOTE demonstrated the least favorable values. Upon examining the line diagram in Fig. 4.7, it becomes evident that during the initial ten chunks, the PA's performance might be suboptimal because of the limited number of classifiers in the pool. However, after the first ten chunks, the performance improved significantly with the inclusion of more classifiers. In the same diagram, PA consistently

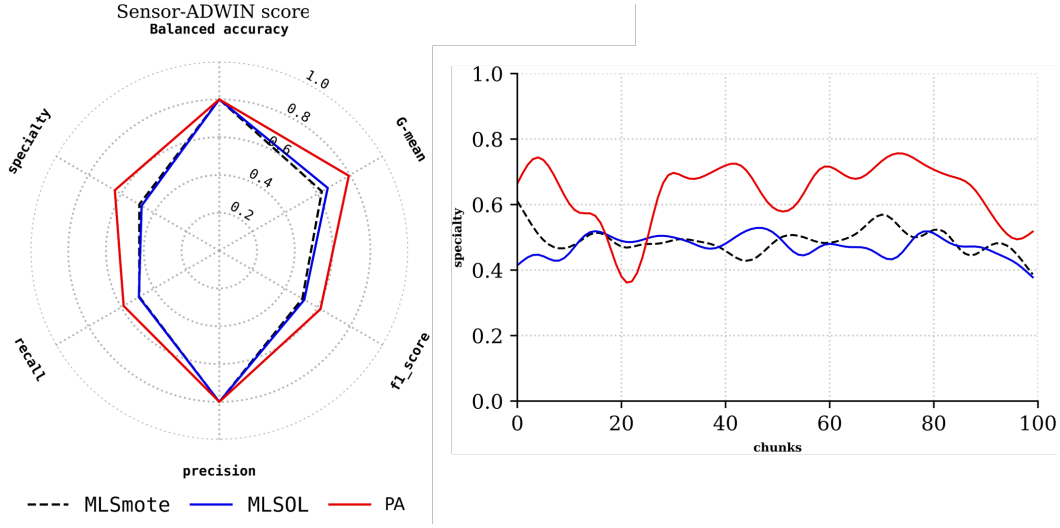


Figure 4.5: Synthetic Data Generator Flow.

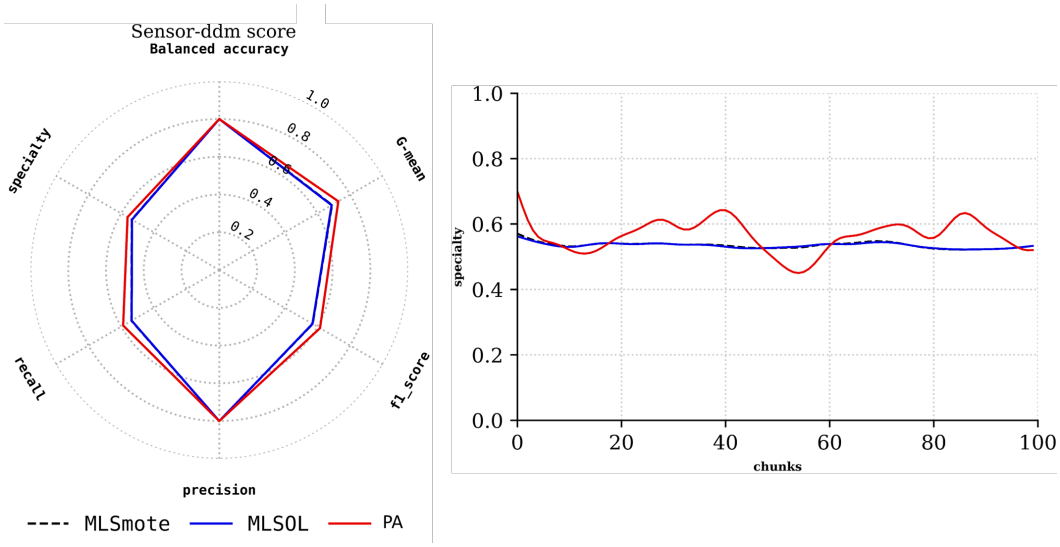


Figure 4.6: Synthetic Data Generator Flow.

achieves the highest performance across all chunks, whereas MLSOL maintains satisfactory performance, and MLSMOTE exhibits lower performance throughout. In Fig. 4.8, using the same synthetic data stream but with the DDM as the drift detector, the radar plot metrics produce similar results to the previous experiment. However, the line diagram emphasizes the same outcomes as those in the previous experiment. Nevertheless, MLSOL and MLSMOTE achieved lower values than

the previous experiment. These findings confirmed ADWIN's superiority of ADWIN over DDM when applied to synthetic data streams. These results highlight the versatility and robustness of the algorithm, demonstrating its effectiveness in handling concept drift across diverse datasets, including the challenging synthetic dataset, Covertype dataset, and Sensor dataset, regardless of whether ADWIN or DDM is used as the concept drift detector.

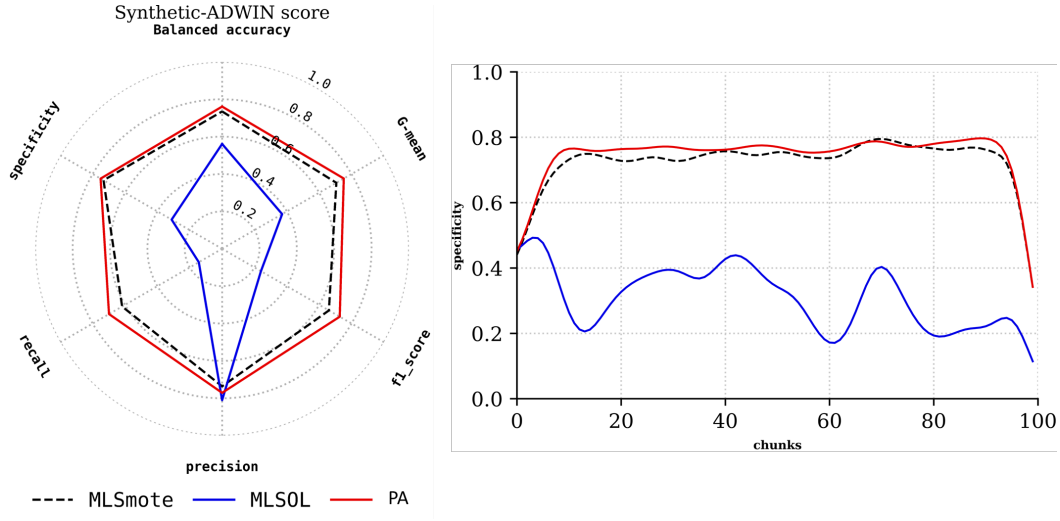


Figure 4.7: Synthetic Data Generator Flow.

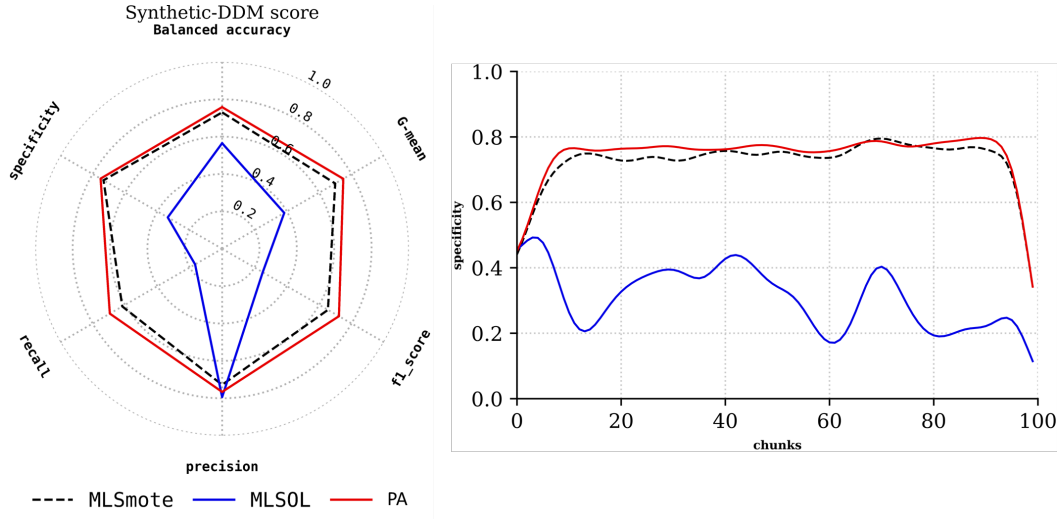


Figure 4.8: Synthetic Data Generator Flow.

4.3.4 Analysis of Class Overlap Factor between Proposed Approach (PA), MLSMOTE, and MLSOL Techniques.

In our experimental study, we aimed to identify the critical factors that influence the selection of an optimal approach for minority classes in imbalanced drifted streams. These factors include various aspects, such as dataset characteristics, the choice of concept drift detectors, and the effectiveness of the algorithm in handling class overlap. To compare the overlapping class behavior of our proposed approach (PA) with MLSMOTE and MLSOL, we systematically designed experiments organized into groups of ten chunks. The results were visualized in bar diagrams, contrasting PA with other methods (MLSMOTE and MLSOL). Figures 4.94.104.11 present ten groups, each with three bars representing MLSOTE, MLSOL, and PA, respectively. Each bar visually represents overlapped samples for ten chunks of several data streams, considering distinct concept drift detectors, specifically ADWIN and DDM. Fig. 4.9 shows two diagrams for the ADWIN and DDM detectors. In the ADWIN diagram, the third bar group (chunks 20-30) lacks overlapping samples because this chunk range does not have drifts, and no synthetic samples are generated for the training step. The sixth and tenth groups have the highest overlapped samples because these groups experience many drifts, leading to the generation of several samples and, consequently, the data indicates that MLSMOTE displays the highest number of overlapping samples across all groups, while PA exhibits very few, except for the last group (90-100), which has a small number of overlapping samples. However, the DDM diagram has more overlapping samples than the ADWIN diagram because the DDM detector detects fewer drifts than ADWIN. Specifically, the last value on the Y-axis of the ADWIN diagram is 3,000 samples, while the last value on the Y-axis of the DDM diagram is 2,000 samples. Fig. 4.10 and Fig. 4.11 display the overlapping samples of the three methods on the Sensor and synthetic data streams, respectively. In Fig. 4.10, the ADWIN diagram demonstrates fewer overlapped samples than in Fig. 4.10, indicating a lower number of drifts in the Sensor stream. The overall diagram indicates that our proposed approach achieves fewer overlapped samples than MLSOL and MLSMOTE, with MLSMOTE having the highest number of overlapped samples. In the DDM diagram of Fig. 10, the number of overlapped samples is lower than in

Fig. 4.9, with our proposed approach consistently achieving the lowest number of overlapped samples across most group bars. In Fig. 4.11, the ADWIN and DDM diagrams exhibit the greatest number of overlapped samples compared to the earlier figures. This suggests that the synthetic stream underwent frequent drifts and was more susceptible to noise. Consequently, the last value on the Y-axis in both the ADWIN and DDM diagrams was recorded for 7,000 samples. Our proposed approach consistently achieves the lowest number of overlapped samples, whereas MLSMOTE consistently attains the highest number of overlapped samples across most group bars in both the ADWIN and DDM diagrams. In conclusion, our thorough examination of overlapping class instances across MLSMOTE, MLSOL, and our proposed approach consistently reveals minimal overlapped samples compared to other methods, regardless of whether DDM or ADWIN is employed as drift detectors

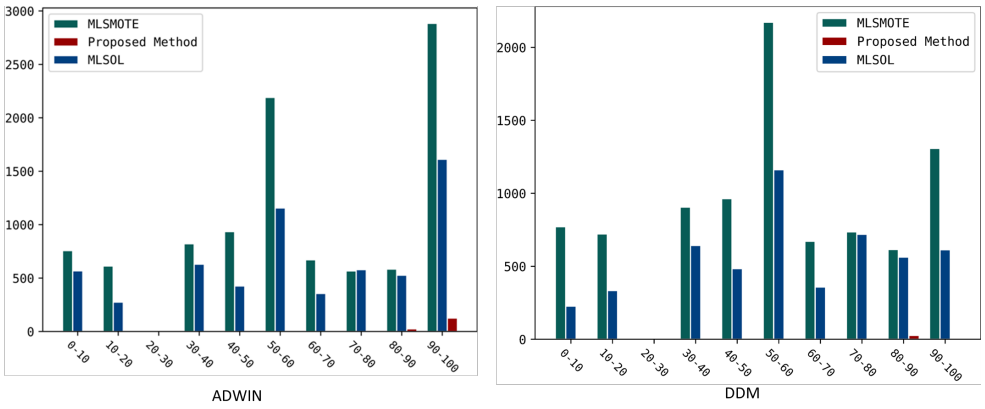


Figure 4.9: Synthetic Data Generator Flow.

4.3.5 Analyzing Runtime Factor Between the Proposed Approach, MLSMOTE, and MLSOL Techniques

The results of our experiments indicate that the choice of the best algorithm for multiclass imbalanced streams depends on various factors, including dataset characteristics, the concept drift detector used, the presence of an overlapping class problem, and the algorithm's runtime demands. To investigate the runtimes of MLSMOTE, MLSOL, and our proposed approach, we conducted experiments, as

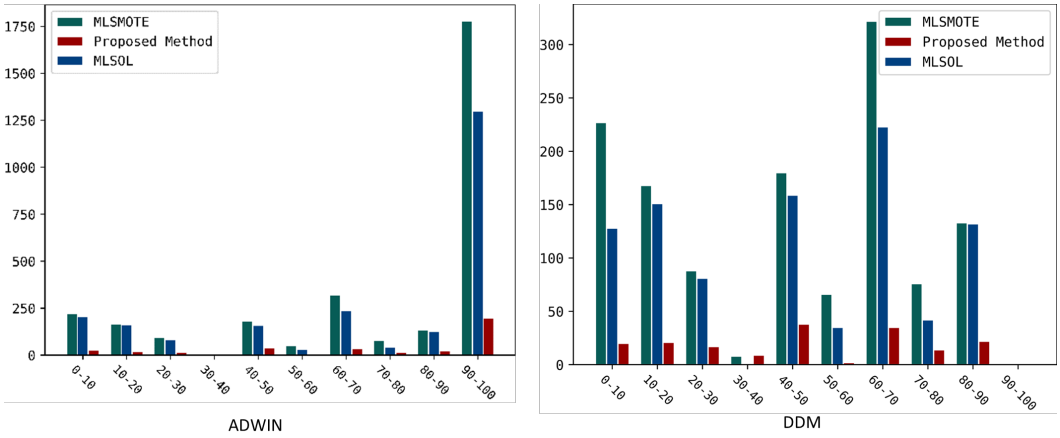


Figure 4.10: Synthetic Data Generator Flow.

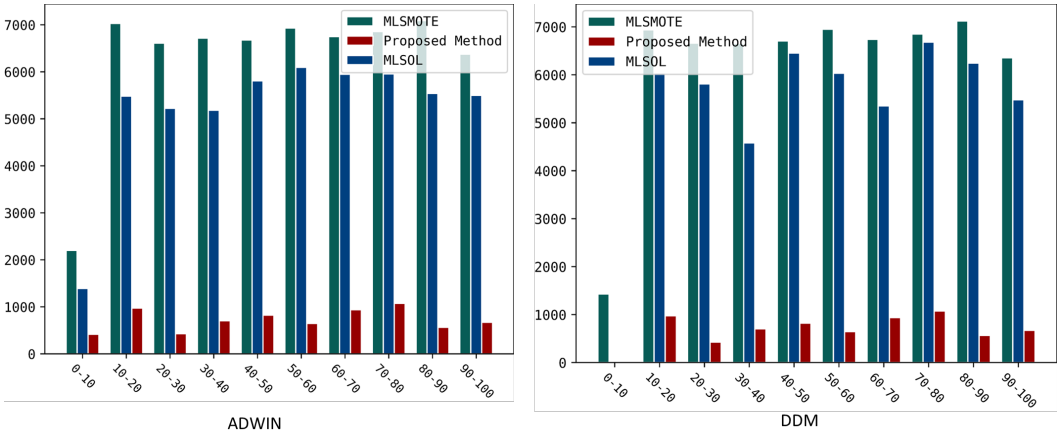


Figure 4.11: Synthetic Data Generator Flow.

shown in Table 4.2. Our findings reveal that our proposed approach is highly efficient, regardless of whether ADWIN or DDM is used as the concept drift detector. Specifically, when ADWIN was used, the proposed approach algorithm took 8344 s to train and predict the Covertype stream for ensemble classifiers, whereas it took 8019 s when using the DDM detector. In contrast, MLSMOTE and MLSOL require more time for the same task. Notably, our proposed approach maintains efficiency, even when using the DDM concept detector. Additionally, our proposed approach demonstrates shorter processing times in the Sensor stream and synthetic data compared with other methods. Consistently, the DDM detector achieved less time across all experiments owing to its lower detection of drifts compared to the

ADWIN detector. This is because there are fewer instances that trigger training for pool classifiers when using the DDM. The bold highlighting in Table 4.2 further emphasizes the efficiency of our proposed approach with both the ADWIN and DDM detectors across all dataset streams. Because our proposal generates fewer overlapped samples, leading to an overall decrease in the running time.

Stream	Concept Drift Detector	MLSMSOTE	MLSOL	PA
Benchmark	ADWIN	9559	8655	**8344**
	DDM	8388	8031	**8019**
Real Application	ADWIN	1291	1310	**1102**
	DDM	585	607	**521**
Synthetic	ADWIN	12870	4866	**4834**
	DDM	12397	4958	**4687**

Table 4.2: Runtime of MLSMOTE, MLSOLE, and Proposed Approach (PA)

4.3.6 Analyzing Non-parametric Tests between the Proposed Approach, MLSMOTE, and MLSOL Techniques

We conducted a thorough series of statistical analyses encompassing 12 comparisons across three diverse datasets, three methods, and two drift detectors. These analyses were rigorously evaluated using a non-parametric test, specifically the Kruskal-Wallis test. The results of this test were striking, revealing substantial variations in the G-mean measurements across most experiments. Importantly, these differences were not due to random chance [42]. Upon closer examination of the assessments for the three methods, PA, MLSMOTE, and MLSOTE, as detailed in Table 4.3, we found compelling evidence supporting the acceptance of the null hypothesis (H0). This implies that the expected and observed data exhibited statistically significant disparities. H0 is rejected when significant differences are not observed and H0 is accepted if the P-value falls below the critical value. Underlining the significance of our analysis, the Kruskal-Wallis test was conducted with a 95% confidence level, and the P-value was rounded to the first three digits after

the decimal point. Nevertheless, it is important to note that a similarity in the performances of the methods emerged in the third and fourth experiments, resulting in the rejection of H0. This similarity can be attributed to the fact that the DDM detected fewer drifts in these experiments.

Table 4.3: Kruskal-Wallis test between MLSMOTE, MLSOLE, and Proposed Approach (PA)

Dataset	Drift detector	Comparison	P-value	Critical value	H0
Covertype stream	ADWIN	PA - MLSSMOTE	0.001	0.05	Accept
		PA - MLSOL	0.014	0.05	Accept
	DDM	PA - MLSSMOTE	0.361	0.05	Rejected
		PA - MLSOL	0.401	0.05	Rejected
Sensor stream	ADWIN	PA - MLSSMOTE	0.001	0.05	Accept
		PA - MLSOL	0.001	0.05	Accept
	DDM	PA - MLSSMOTE	0.001	0.05	Accept
		PA - MLSOL	0.001	0.05	Accept
Synthetic stream	ADWIN	PA - MLSSMOTE	0.001	0.05	Accept
		PA - MLSOL	0.001	0.05	Accept
	DDM	PA - MLSSMOTE	0.001	0.05	Accept
		PA - MLSOL	0.001	0.05	Accept

4.4 Conclusion and Future Works

Our study in this chapter presents a comprehensive methodology designed to facilitate incremental learning in drifted streams, with a particular focus on addressing the challenges associated with imbalanced data streams including minority and overlapping classes. The proposed methodology integrates our proposed oversampling method, concept drift detection strategies, and Dynamic Ensemble Selection (DES) to select the most suitable ensemble classifier. Extensive experimentation across various datasets, including benchmark datasets, real application streams, and synthetic data, validated the effectiveness of our contribution. The critical role of

the concept drift detector in our approach lies in its capacity to promptly detect concept drifts, allowing our methodology to adapt by training new base classifiers to maintain relevance and accuracy in real-time scenarios. Oversampling techniques were utilized to mitigate the minority class problem, and the KNN algorithm was employed to prevent the generation of overlapped class instances. The DES technique was utilized to intelligently select the best base classifiers to ensure optimal performance. Our evaluation approach employs various performance measures, showcasing its proficiency in addressing multiclass imbalanced stream problems, particularly its exceptional accuracy in classifying data streams with evolving class distributions. In addition to its accuracy and performance, our approach has several advantages, including adaptability, efficiency, and scalability. Dynamic model updates driven by incoming data instances enable continuous adaptation to changing data distributions, thereby ensuring reliability and relevance in real-time scenarios. However, it is essential to recognize the specific limitations of the proposed approach. One limitation is the extended time required to generate nonoverlapping synthetic instances. Moreover, the efficacy of our contribution depends on the accuracy of the MLSMOTE and MLSOL techniques. Consequently, future research efforts should be directed toward enhancing our contributions. Potential avenues for improvement include the development of advanced oversampling techniques to avoid generating overlapping synthetic instances. Additionally, meta-learning methods have been explored to calculate imbalanced multiclass ratios and minority classes.

Everything is theoretically impossible, until it is done.

Robert A. Heinlein

CHAPTER

5

Conclusions and Future Work

In this chapter, we summarize the main contributions, limitations, and future lines of research resulted from this thesis. First, a consolidated view of results and contributions is presented in Section 5.1. Afterward, the limitations of this work are identified in Section 5.2. Finally, the future research lines from this thesis are exposed in Section 5.3.

5.1 Summary of Contributions

Machine learning is an artificial intelligence technology that gives systems the ability to learn and develop from experience automatically without being programmed specifically. The primary aim is to help computers learn automatically without human intervention or assistance. This field still receiving great interest due to the growth of data, computing power, and statistical algorithms. While data mining is a closely related topic, aims to discover useful, valid, understandable, and unexpected patterns from data. However, each learning algorithm discovers the pattern from a different perspective. Regarding that, ensemble data mining has been proved as an optimal strategy to aggregate and combine several learning algorithms

together. For the classification task, the aggregated models are well-known with the name of multiple classifier systems (MCS). The main objective of the thesis is to enhance the generation and the integration of MCS via soft computing techniques.

In the context discussed above, MCS are hybrid intelligent systems with the potentiality to cope with ambiguity, uncertainty, and complex problems. While soft computing methods support intelligent control, nonlinear programming, optimization, and decision making. Soft computing methods exploit the tolerance for imprecision, partial truth, and uncertainty to achieve tractability, robustness, low cost solution. With the human mind as a role model, soft computing enables solutions for problems that may be either unsolvable or just too time-consuming. This could be particularly helpful to optimize the design and the integration of the complex MCS.

The research developed in this dissertation has contributed to the enhancement of MCS. The experiments have been conducted on popular benchmark datasets. In chapter 2, we presented a nice taxonomy for MCS. Afterward, we have discussed the importance of diversity and how we can measure it through pairwise and non-pairwise metrics. Furthermore, the concept of error diversity is presented to alleviate the consolidated error. Finally, the importance of soft computing is highlighted within the area of MCS. In chapter 3, we have reviewed the state-of-the-art classifier ensembles, this included; bagging-like ensembles, boosting, and gradient boosting ensembles. The presented algorithms were compared together in terms of how to promote diversity, and the type of the base model. Afterward, due to the complexity of MCS, metaheuristic algorithms (MA) were specifically applied to better integrate or prune the classifiers set. Finally, from the conducted revision, we identified the gaps and the research questions that we answered in this thesis. Among the chief contributions of this thesis:

Chapter **??**, is dedicated to the first objective: *"To build more diverse and highly accurate MCS, only from a reduced portion of the available data"*. The complexity of the classifiers ensemble increases positively with the size of training data. To reduce this complexity, IS techniques could be used as a preliminary step to reduce the training data-size. First, the border noisy samples are removed via intelligent data sampling. Hence, pure, reduced, and informative data samples can be obtained to train individual classifiers quickly and correctly. Second, the proposed

MCS, Section **??**, considers two strategies to promote diversity; data manipulation and algorithm manipulation techniques. For data manipulation, the bagging and the random feature selection are applied to the reduced dataset from phase one. While, for algorithmic manipulation, the diversity is promoted via generating heterogeneous classifiers. Finally, SI algorithms were incorporated to enhance MCS predictivity. Where, a weighted voting schema is optimized via MFO, GWO, and WOA algorithms to enhance the fusion of multiple decisions. The novelty of this contribution is the intersection between three computational intelligence paradigms: instance selection, ensemble learning, and swarm intelligence. The effect of IS and SI on the generalization performance of MCS has been analyzed and discussed in detail in Sections **??**, and **??**, respectively. With the conclusion, IS proved its effectiveness to reduce the training data-size of 17 datasets by more than 25%. AllKNN, as IS technique, did not prove its capability to capture the integrity from the whole data, where RFSM outperform RFCOM in only 4 out of 25 datasets. The mistake of IS to capture informative samples has been compensated via our proposal. The proposed MCS with a simple combination function, majority voting, outperforms RFCOM in 8 out of 25 datasets. While, a great improvement has been achieved via SI, weighted voting strategy, to outperform RFCOM in 14 out of 25 datasets. Concluding, the predetermined first objective is partly achieved due to the limited performance of the IS method.

Chapter **??**, is dedicated to the second objective: *"Increasing the efficiency of MCS and going beyond what can be achieved from ensemble pruning methods"*. A framework has been proposed to benefit from the power of instance selection and ensemble selection simultaneously. In relation to that, IS methods can be applied as a kind of data preprocessing to clean and eliminate inconsistent data. While ensemble pruning is the strategy by which a small-size ensemble can be selected without affecting the general performance of the original ensemble. Hence, the computational resources can be saved, and the testing time can be accelerated by depending on some classifiers instead of all. Ensemble pruning is the second component, that was integrated into the framework, Section **??**, to elevate its performance. A guided search-based MCS pruning has been proposed to consider both the classifier's accuracy and ensemble diversity. First, two ordering-based pruning metrics have been applied, where each of them returns a set of classifiers. The suggested

classifier set from each metric is merged together to form a new subensemble to be further searched via metaheuristic methods. The proposal successfully solved the parameter tuning challenges, the alpha parameter in Equation **??**, which is part of a novel and recent pruning metric, MDEP [1]. The limited accuracy of MCS pruning metrics has been maximized via the proposed guided search-based pruning. This has been proved and clarified as in Figure **??**, and according to the statistical analysis, Section **??**. In this work, small-size ensembles with training on fewer samples could outperform significantly the large-size ensembles which use the whole available training data. Concluding, the predetermined second objective is totally achieved.

Chapter **??**, is dedicated to the third objective: *"Grouping and analyzing fast and accurate heuristic metrics for MCS pruning"*. Ensemble pruning strategies are one of the hot topics to gain efficient and effective ensembles. The efficiency could be reached via forming small-size ensembles with their impact; to consume short memory space, reduce the communication cost of the distributed models, and to accelerate the prediction time. While the efficacy could be achieved via building trustable models with a high level of prediction accuracy. We applied a fast and accurate strategy to work with bagging ensembles via ranking the ensemble members. This pruning strategy is known as "ordering-based pruning" to identify the best subset of classifiers for early aggregation. The identification of this subset is mainly controlled through a heuristic measurement to optimize the augmented subensemble. In this chapter, greedy search methods and group-based ranking have been considered to reorder the pool of classifiers. Since the great analysis by Martínez-Muñoz et al. [2] in 2009, several heuristic metrics belonging to this category were proposed [1, 3–5] with no existence comparison between them. Regarding that, our proposal was conducted to solve that gap. The conclusions from this analysis proved that the efficacy of those metrics is affected by the original ensemble size, the required subensemble size, the kind of individual classifiers, and the number of classes. Furthermore, the investigated metrics realize robust and stable predictions, this is analyzed via the range of their prediction accuracy around the median as shown in Figure **??**. Finally, the computational cost of some metrics is linear with the initial pool size T, while other metrics have a larger computational cost, which is quadratic in T. Concluding, the predetermined third objective is totally achieved.

5.2 Limitations

In relation to the contributions summarised above, this thesis also presents limitations that need to be mentioned. In this section, we highlight some aspects that are a barrier to better improvement.

- In Chapter **??**, the proposed MCS uses reduced data for training. However, we cannot guarantee whether or not the reduced data via IS could keep the integrity from the original data. In some cases, the reduced data miss informative samples which leads to bad training. This has been analyzed and clarified via the performance on D_{14}, D_{19} from Table **??**.

- In Chapter **??**, the proposed approach is characterized by a relatively high computational complexity. Therefore, it is not suitable for *online* learning, e.g., in the case of nonstationary data classification streams, namely, when the *concept drift* phenomenon can occur. Instead, it can be a suitable alternative if the training time is not a critical parameter from the application perspective.

- In Chapter **??**, the guided search-based pruning identifies a subensemble with higher predictive performance, but the other pruning metrics (EPIC, UMEP, MDEP) identify a thinner/small-size subensemble. This is clarified by the statistical analysis in Table **??**.

- The presented work in this thesis has not been scaled up to prove its suitability for big datasets. In addition, the investigated work did not consider problems with certain properties, like imbalanced class labels.

5.3 Future works

As the limitations of our study indicate, there are numerous ways to beneficially extend this line of research in the future. They range from improving the training phase of MCS to pruning and combining MCS efficiently.

1. The proposed MCS is heterogeneous, containing different models, with the aim to promote diversity. While the identification of classifiers to be included in classifier ensembles remains a key issue for predictive performance. In [6], the classifiers types to be consolidated are selected manually via the majority voting error and forward search. Experimentally, the set of classifiers solving a task could not be effective for another task [7]. The inclusion of weak classifiers could degrade the general performance unless if they create complement decisions for particular samples, this is too complex and need to be further discovered and analyzed via tailored metrics to determine the individual classifier type.

2. In Chapters **??** and **??**, AllKNN [8], as a training set selection, has been applied due to its reasonable selection time, reduction rate, and limited-accuracy over test [9]. An interesting research line would be to evaluate the performance of other training set selection algorithms. Particularly, how each reduction method could add to the performance of MCS. In addition, the concept of instance selection had been widely used with the support vector machine classifier [10, 11] to alleviate its computational difficulties when dealing with huge amounts of data. In connection with that, a Pareto-based SVM ensemble has been proposed in [12] to optimize the size of the training set and the classification performance attained by the selection of the instances. Thus, the consideration of those modeling approaches could potentially lead to new promising versions for both homogeneous and heterogeneous MCS.

3. In Chapter **??**, We have identified that both the classifier's accuracy and the ensemble diversity are crucial for MCS pruning. Our proposed schema is so simple and innovative. The computational cost of our solution is light, due to the usage of ordering metrics, in comparison to evolutionary algorithms. While multi-objective optimization could be further applied to tune the calibration between the subensemble size and the subensemble accuracy. Examples of recent works in this area [13].

4. The concept of uncorrelated error, Section **??**, is so important to alleviate the consolidated error. Regarding that, the weights of Equation (**??**) can be optimized by evolutionary algorithms. Furthermore, the subensemble could be better located if the concept of uncorrelated error can be incorporated in the investigated metrics of Chapter **??**.

5. In Chapter **??**, the SI-based weighted voting schema considers the abstract predictions from the individual classifiers. In contrast, the class probability distribution carries information that should not be ignored [14]. For example, if we take the votes according to the class labels, then the incorrect outputs of the mistaken members will be amplified into wrong votes. A future research line could be interesting to apply SI for the class probability distribution. Furthermore, transforming the outputs of ensemble members into class labels before considering their soft votes, leads to destroying the real distribution and losing useful information [15]. Besides, the usage of soft voting could be more efficient for ensemble pruning via decreasing the probability that a tie occurs, i.e in majority voting.

6. In [16], the authors recommended using the classification confidence and the weights of the base classifiers to define the ensemble margin. They used a homogeneous ensemble of SVM, and the classification confidence for each classifier is calculated in terms of the distance between sample $\mathbf{x}_i$ and the hyperplane. However, it is not clear how to measure the classification confidence for different classifier types, an interesting research line could be to extend their proposal to adapt to a heterogeneous ensemble. In addition, in the same article, the weighted voting of the pruned ensemble proved superior results, in terms of accuracy, over the majority voting of the pruned ensemble. Regarding that, the investigated metrics of Chapters **??** and **??** could be further enhanced.

By taking these issues into account, the predictive performance of MCS scheme could be further enhanced.

Bibliography

[1] H. Guo, H. Liu, R. Li, C. Wu, Y. Guo, and M. Xu, "Margin & diversity based ordering ensemble pruning," *Neurocomputing*, vol. 275, pp. 237–246, 2018.

[2] G. Martínez-Muñoz, D. Hernández-Lobato, and A. Suárez, "An analysis of ensemble pruning techniques based on ordered aggregation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 245–259, 2008.

[3] Z. Lu, X. Wu, X. Zhu, and J. Bongard, "Ensemble pruning via individual contribution ordering," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 871–880.

[4] L. Guo and S. Boukir, "Margin-based ordered aggregation for ensemble pruning," *Pattern Recognition Letters*, vol. 34, no. 6, pp. 603–609, 2013.

[5] J. Cao, W. Li, C. Ma, and Z. Tao, "Optimizing multi-sensor deployment via ensemble pruning for wearable activity recognition," *Information Fusion*, vol. 41, pp. 68–79, 2018.

[6] A. Onan, S. Korukoğlu, and H. Bulut, "A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification," *Expert Systems with Applications*, vol. 62, pp. 1–16, 2016.

[7] D. H. Wolpert, "The supervised learning no-free-lunch theorems," in *Soft computing and industry*. Springer, 2002, pp. 25–42.

[8] I. Tomek, "An experiment with the edited nearest-neighbor rule," *IEEE Transactions on systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 448–452, 1976.

[9] S. Garcia, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 3, pp. 417–435, 2011.

[10] C. Liu, W. Wang, M. Wang, F. Lv, and M. Konan, "An efficient instance selection algorithm to reconstruct training set for support vector machine," *Knowledge-Based Systems*, vol. 116, pp. 58–73, 2017.

[11] J. Chen, C. Zhang, X. Xue, and C.-L. Liu, "Fast instance selection for speeding up support vector machines," *Knowledge-Based Systems*, vol. 45, pp. 1–7, 2013.

[12] A. Rosales-Pérez, S. García, J. A. Gonzalez, C. A. C. Coello, and F. Herrera, "An evolutionary multiobjective model and instance selection for support vector machines with pareto-based ensembles," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 6, pp. 863–877, 2017.

[13] S. Fletcher, B. Verma, and M. Zhang, "A non-specialized ensemble classifier using multi-objective optimization," *Neurocomputing*, 2020.

[14] R. Sikora *et al.*, "A modified stacking ensemble machine learning algorithm using genetic algorithms," in *Handbook of Research on Organizational Transformations through Big Data Analytics*, 2015, pp. 43–53.

[15] Q. Dai, T. Zhang, and N. Liu, "A new reverse reduce-error ensemble pruning algorithm," *Applied Soft Computing*, vol. 28, pp. 237–249, 2015.

[16] L. Li, Q. Hu, X. Wu, and D. Yu, "Exploration of classification confidence in ensemble learning," *Pattern recognition*, vol. 47, no. 9, pp. 3120–3131, 2014.