DYNAMIC ENSEMBLE DIVERSIFICATION AND HASH-BASED UNDERSAMPLING FOR THE CLASSIFICATION OF MULTI-CLASS IMBALANCED DATA STREAMS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BİLKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

By

Soheil Abadifard

July 2024

Dynamic Ensemble Diversification and Hash-Based Undersampling for the Classification of Multi-Class Imbalanced Data Streams
By Soheil Abadifard
July 2024

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Fazlı Can(Advisor)

Özgür Ulusoy

Çağrı Toraman

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

DYNAMIC ENSEMBLE DIVERSIFICATION AND HASH-BASED UNDERSAMPLING FOR THE CLASSIFICATION OF MULTI-CLASS IMBALANCED DATA STREAMS

Soheil Abadifard
M.S. in Computer Engineering
Advisor: Fazlı Can
July 2024

The classification of imbalanced data streams, which have unequal class distributions, is a key difficulty in machine learning, especially when dealing with multiple classes and concept drift. While binary imbalanced data stream classification tasks have received considerable attention, only a few studies have focused on multi-class imbalanced data streams. Additionally, dealing with the dynamic imbalance ratio is of great importance. This study introduces a novel, robust, and resilient approach to address these challenges by integrating Locality Sensitive Hashing with Random Hyperplane Projections (LSH-RHP) into the Dynamic Ensemble Diversification (DynED) framework. To the best of our knowledge, we present the first application of LSH-RHP for undersampling in the context of imbalanced non-stationary data streams. The proposed method, undersamples majority classes by utilizing LSH-RHP, provides a balanced training set, and improves the ensemble's prediction accuracy. We conduct comprehensive experiments on 23 real-world and ten semi-synthetic datasets and compare LSH-DynED with 15 state-of-the-art methods. The results reveal that LSH-DynED outperforms other approaches in terms of both Kappa and mG-Mean effectiveness measures, demonstrating its capability in dealing with multi-class imbalanced non-stationary data streams. Notably, LSH-DynED performs well in large-scale, high-dimensional datasets with considerable class imbalances and demonstrates adaptation and robustness in real-world circumstances. For the reproducibility of our results, we have made our implementation available on GitHub.

Keywords: Data stream, Classification, Class imbalance, Concept drift, Ensemble learning, Locality sensitive hashing.

ÖZET

ÇOK SINIFLI DENGESİZ VERİ AKIŞLARININ SINIFLANDIRILMASI İÇİN DİNAMİK TOPLULUK ÇEŞİTLENDİRME VE KARGAŞA-TABANLI AZ ÖRNEKLEME

Soheil Abadifard
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Fazlı Can
Temmuz 2024

Eşit olmayan sınıf dağılımlarına sahip dengesiz veri akışlarının sınıflandırılması, özellikle çoklu-sınıf ve kavram kayması ile uğraşırken makine öğreniminde çözülmesi gereken önemli problemlerden biridir. İkili dengesiz veri akışı sınıflandırma işlemi literatürde büyük ilgi görmüşken, ancak birkaç çalışma çok sınıflı dengesiz veri akışlarına odaklanmıştır. Ayrıca, dinamik dengesizlik oranıyla başa çıkmak büyük önem taşımaktadır. Bu çalışma, Dinamik Topluluk Çeşitlendirme (DynED) çerçevesine Rastgele Hiper Düzlem Projeksiyonları ile Yerelliğe Duyarlı Kargaşa'yı (LSH-RHP) entegre ederek bu zorlukları ele almak için yeni, sağlam ve esnek bir yaklaşım sunmaktadır. Bildiğimiz kadarıyla, dengesiz ve durağan olmayan veri akışları bağlamında az örnekleme için LSH-RHP'nin ilk uygulamasını sunuyoruz. Önerilen yöntem, LSH-RHP kullanarak çoğunluk sınıflarını az örneklemekte, dengeli bir eğitim seti sağlamakta ve topluluğun tahmin doğruluğunu artırmaktadır. Bu çalışmada gerçek dünyadan 23 ve yarı sentetik 10 veri kümesi üzerinde kapsamlı deneyler gerçekleştirilmiş ve LSH-DynED 15 önde gelen yöntemle karşılaştırılmıştır. Sonuçlar, LSH-DynED'in hem Kappa hem de mG-Mean etkinlik ölçütleri açısından diğer yaklaşımlardan daha başarılı olduğunu ve çok sınıflı dengesiz, durağan olmayan veri akışlarıyla başa çıkma konusunda daha etkin olduğunu göstermektedir. Özetle, LSH-DynED önemli sınıf dengesizliklerine sahip büyük ölçekli, yüksek boyutlu veri kümelerinde iyi performans sergilemekte ve gerçek dünya koşullarına uyumluluk ve sağlamlık sergilemektedir. Sonuçlarımızın yeniden üretilebilirliği için uygulamamız GitHub'da kullanıma sunulmuştur.

Anahtar sözcükler: Veri akışı, Sınıflandırma, Sınıf dengesizliği, Kavram kayması, Topluluk öğrenmesi, Yerelliğe duyarlı karma.

Acknowledgement

Words cannot express my gratitude to my advisor, Prof. Fazlı Can, for his unwavering support, invaluable guidance, and continuous encouragement throughout my Master's journey. His expertise and insightful feedback have been instrumental in successfully completing this thesis and my previous research.

I am deeply honored and humbled to extend my sincere thanks to the members of my thesis committee, Prof. Özgür Ulusoy and Prof. Çağrı Toraman. Their valuable insights and thoughtful suggestions have greatly enhanced the quality of my research.

I am profoundly grateful to the Computer Engineering Department for its financial support and the provision of excellent resources and facilities. The department's commitment to fostering a conducive learning and research environment has been crucial to my academic and personal growth.

Special thanks to Sepehr Bakhshi, who has been a constant source of help and inspiration. I have learned much from him, and his assistance has been invaluable throughout my study.

To my beloved spouse, I offer my most profound and heartfelt thanks. Your unwavering support and boundless patience have been the very foundation upon which this journey was built. Without you, this dream would have remained just that – a dream.

I also extend my gratitude to my family; their belief in me has kept my spirits and motivation high during my studies.

Finally, I am filled with immense gratitude towards Bilkent University for providing the fertile ground in which friendships have blossomed and bonds have been forged. The relationships cultivated here have enriched my life beyond measure, and for this, I am eternally thankful.

Contents

List of Figures

List of Tables

List of Publications

This thesis includes content from following publications:

1. **Abadifard, S.**, Bakhshi, S., Gheibuni, S., and Can, F. Dyned: Dynamic ensemble diversification in data stream classification. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (New York, NY, USA, 2023), CIKM '23, Association for Computing Machinery, p. 3707–3711.

2. **Abadifard, S.**, and Can, F. Dynamic Ensemble Diversification and Hash-Based Undersampling for the Classification of Multi-Class Imbalanced Data Streams. Knowledge-Based Systems (To be Submitted).

Chapter 1

Introduction

In this chapter, we start by introducing the data stream and explaining its basic concepts in Section 1.1. Then, in Section 1.2, we discuss why we conducted this study and what we hope to achieve with it.

1.1 Fundamental Concepts

Data Stream: Recently, there has been an explosive increase in the volume of data generated by various sources, a continuous data with high velocities, called data stream: "A data stream is a sequence of data elements made available over time" [90] and need to be processed and assessed in real-time [76]. Two main approaches are presented in the literature to process a data stream in real-time fashion: Online and Chunk-based methods [32]. Data streams are characterized by their continuous flow, unbounded nature, high velocity, one-pass processing, and dynamic changes in content over time [8, 58, 90]. These properties are different from static data, which have all data available while processing. This trend poses new hurdles for traditional machine learning techniques designed initially to handle static data. In the context of data streams, data samples are processed

incrementally using methods appropriate for real-time or nearly real-time analysis, which is crucial in industries such as banking, healthcare, and transportation. Additionally, some of these data may also come from sensors, social media feeds, or Internet of Things devices, all of which produce big data streams since they produce large amounts of data not collected based on predetermined data characteristics. For instance, IoT devices are expected to grow from 26 billion connected units in 2019 to nearly 80 billion by 2025 [6]. This necessitates using specialized algorithms that can manage the high throughput and adjust to the dynamic nature of the data stream. Consequently, this task has become a focus of research, developing algorithms capable of performing real-time tasks such as incremental clustering and classification. Current research in unsupervised and supervised learning within data streams explores methods like density-micro clustering and classification algorithms [52], respectively. This study focuses on multi-class data streams where each data sample is associated with a single label, as opposed to multi-label data streams, in which each data sample is linked to a combination of labels [7, 75].

Imbalanced data streams: One of the most difficult challenges in the data stream classification task is effectively handling class imbalance in data. *Imbalanced data streams* refer to situations when there is a significant difference in the number of instances between various classes [4, 62]. These streams exhibit a notable difference in the representation of different classes, with some classes being referred to as minority classes and others as majority classes. The difference in the distribution of classes poses significant difficulties for any machine learning algorithm, as they may display a bias towards the dominant class and often overlook the less common but crucial information included within the minority class [61]. This difference might result in poor predictive accuracy, particularly when identifying instances from the minority classes. The ever-changing characteristics of data streams result in a fluctuating imbalance ratio, which can reverse the roles of majority and minority classes as the stream progresses.

Concept Drift: In addition to the challenge of imbalance, another significant obstacle in machine learning is to deal with how data streams constantly change, a phenomenon called concept drift [42, 43, 79]. As data distributions change

over time, it is crucial to have a learning system that can adapt. Concept drift happens when the model's statistical properties face an unexpected change in the target variable. This can significantly lower the accuracy of models, making it hard to keep their performance high. To handle concept drift effectively, one needs methods to detect it and adjust the learning process, such as updating the model with new data [8], using algorithms to detect drift [5], or resetting the model periodically. The combination of these challenges, along with overlapping class boundaries, generates complicated classification tasks [4].

Therefore, a combination of concept drift and class imbalance introduces complex dynamics where class roles may shift, new classes may emerge, or existing classes might disappear, requiring continual adaptation of the classifiers. Handling these challenges effectively demands specialized techniques and algorithms, such as ensemble methods, feature selection, resampling techniques, and cost-sensitive learning. These approaches either alter the underlying dataset to balance the class distribution or modify the training approaches to make classifiers robust against these imbalances [4]. Hence, effectively handling an imbalanced data stream necessitates employing advanced tactics to guarantee equitable and precise model performance across all classes while acknowledging the dynamic and intricate characteristics of the data at hand.

Recently, the integration of resampling techniques with online ensemble learning to handle imbalanced data streams has attracted significant interest. Resampling is a typical technique for dealing with class imbalance. This technique involves either increasing instances in the minority class (oversampling) or decreasing instances in the majority class (undersampling) [4, 71] to manage imbalanced data distribution. Although numerous sampling algorithms have been created so far, many of these methods are influenced by the traditional SMOTE algorithm [92]. It lowers class imbalance by employing linear interpolation to generate new instances of the minority class from existing minority class samples. Although this technique effectively reduces class imbalance, it may provide noisy and false instances that do not truly belong to the minority class, potentially leading to over-generalizing the minority class into the majority class region and lowering the overall prediction performance [35, 80].

The imbalance issue in the data stream classification task can also be addressed by utilizing ensemble classifiers [9, 19, 39], which combine several weak classifiers to form a stronger one. By combining multiple classifiers, the ensemble utilizes the advantage of each one's unique capabilities to achieve better performance. Furthermore, resampling can be used in these techniques to balance class distribution during training, provide equal representation of all classes, and enhance overall accuracy and minority class handling [61].

1.2 Contributions

The primary motivation behind this research is that the classification of imbalanced binary data streams has been extensively studied; however, multi-class imbalanced data streams have received less attention [12, 44, 53, 78, 89]. Techniques such as VFC-SMOTE and REBALANCESTREAM have addressed binary class imbalances by adapting to evolving data streams, and concept drifts [10, 11]. Additionally, SMOTE-OB has demonstrated the effectiveness of combining synthetic minority oversampling with online bagging to continuously rebalance class distributions in evolving streams [13]. The Adaptive Chunk-Based Dynamic Weighted Majority (ACDWM) method further enriches this landscape by using an ensemble of classifiers dynamically weighted based on their recent error rates to handle both concept drift and class imbalance, demonstrating significant advancements in the field of online learning with imbalanced data streams [67]. These binary classification advancements underline the urgent need for robust solutions in multi-class settings, particularly when faced with non-stationary environments where class distributions and relationships dynamically change. Furthermore, the problem becomes more complicated when class proportions fluctuate dynamically [4] and may even reverse, particularly when concept drift is involved. This can have a negative influence on classification model performance. These necessitate a more robust and resilient method toward the above-mentioned challenges [85]. While resilience is about recovering and adapting after disruptions, robustness is about withstanding and resisting them [86]. This study presents a new approach to

address the issues of multi-class imbalance in non-stationary data streams and aims to introduce a robust and resilient approach.

The contributions of this study are detailed below. We

- Propose a novel undersampling method using randomly generated hyperplanes and hash-codes,
- Extend and modify the DynED [1], the most recent data stream classification method to overcome multi-class imbalanced data stream tasks, especially dynamic imbalance ratio,
- Present comprehensive experiments over approaches proposed for this task over the last decade on multiple real-world and semi-real-world datasets,
- Provide our implementation with all experimental details to make our approach open to new improvements in future research and available it as a baseline.

This study is organized as follows: in chapter 2, we provide an in-depth review of relevant literature; in chapter 3, we present our proposed method; and in chapter 4, we present our experiments and provide a complete and comprehensive discussion of the findings. Lastly, we conclude with chapter 5.

Chapter 2

Related Works

In this chapter, we start by briefly reviewing current methods used in data stream classification in Section 2.1. Then, we continue with a short overview of sampling methods in Section 2.2.

2.1 Data Stream Classification Methods

Data stream classification methods can generally be categorized into two primary groups: General Purpose Methods (GPM) and Imbalance-Specific Methods (ISM). This chapter analyzes several methods developed for the classification of non-stationary data streams and evaluates their strengths and weaknesses. The summary of various approaches can be seen in Table 2.1. We explain them in the following.

2.1.1 General Purpose Methods (GPM)

Most algorithms created for identifying data streams perform poorly in imbalanced scenarios. There is a scarcity of algorithms capable of handling both balanced and unbalanced settings. In this section, we look at the handful that manage both conditions effectively, providing a full study of their techniques and actual application.

OzaBagAdwin (OBA) [16] is a strategy that combines the online bagging approach of Oza and Russell [72] with the ADWIN (ADaptive WINdowing) algorithm [14] for data streams with non-stationary underlying distribution. This technique can be used to generate an ensemble of classifiers that can adjust over time to variations in data distribution, addressing a common challenge in real-world applications. Utilizing a Poisson distribution, each example is handled as a sample taken from the original distribution as part of the approach for simulating bootstrap samples. By keeping a sliding window of current data that is dynamically sized, ADWIN also aids in the simultaneous detection of distributional changes. OBA's key features include the ability to adapt to changing concepts without requiring a fixed window size, the efficient use of memory via ADWIN's automatic window size adjustment, and the robust ensemble approach that enables rapid adaptation by swapping out less accurate classifiers when changes are detected. Because sustaining strong predictive performance despite data evolution is critical, OBA is especially beneficial in financial markets, sensor networks, and real-time prediction systems.

Even though OBA works well with non-stationary data streams, its adaptability, and accuracy may be limited in some situations as it predominantly concentrates on boosting misclassified instances and might not provide enough variation and randomization in training data. As a result, a higher level of randomness has been incorporated into several aspects of the ensemble process to solve these constraints using **Leveraging Bagging** (LB) [15]. This approach responds more quickly to data stream changes and simultaneously improves prediction accuracy. Introducing randomization into classifier algorithms and input data is a feature

of LB. The training examples are more diverse since they use a Poisson distribution to assign weights to them. Random output codes are also used to lower the correlation across classifiers. To ensure that the ensemble closely fits the current data distributions, ADWIN is included for effective change detection. LB is now more resource-efficient and capable of managing data drift than OBA, making it ideal for real-time analytics scenarios in which data evolves quickly.

LB enhances the adaptability of ensembles through increased randomness, yet it might still struggle with diverse types of concept drift. In contrast, the **Adaptive Random Forest** (ARF) algorithm [40] is specifically designed to overcome these constraints by adapting the classical Random Forest to the challenges of evolving data streams. ARF is designed to withstand many forms of concept drifts. It features a clever resampling technique based on online bagging, which improves its adaptability without requiring extensive modifications for various datasets. It includes adaptive operators that alter the learning models in response to shifts in the data stream, as well as parallel processing, which allows each ensemble component to function independently while maintaining classification accuracy. Furthermore, each tree in the ARF ensemble has a drift detection mechanism, which not only detects drifts but also prepares background trees to replace the principal models if a drift is confirmed. This ensures that the ensemble remains relevant and accurate throughout time. ARF's comprehensive approach to drift adaptation and its capability to handle abrupt, gradual, and incremental drifts sets a new standard in data stream classification, addressing gaps left by previous adaptations of random forest algorithms and other ensemble methods.

Although ARF has made significant progress in managing concept drifts in data streams, it could be better at maximizing resource efficiency across massive amounts of data or striking a balance between stability and diversity. Combining online bagging and random subspaces techniques into a single ensemble stream classification method is what **Streaming Random Patches** (SRP) [41] aims to do in order to fill these particular gaps. SRP seeks to process data effectively with limited computer resources and improves adaptation to concept drift. The ensemble technique in SRP promotes diversity by constructing many base models and training them on various subsets of attributes and occurrences. It replicates

instance weighting by using a Poisson distribution and a global technique to select random subsets of features for each learner. The global feature selection method of SRP gives a more complete approach to ensemble diversity than ARF, which employs a local randomization scheme. While OBA focuses on responding to changes in data distribution rather than feature diversity, SRP adds active drift detection and recovery algorithms that provide a more dynamic response to changing environments. SRP also strives for good performance, whereas Leveraging Bagging seeks stability and variation within the ensemble. Because these approaches are not available in other models, SRP is an excellent choice for retaining model performance and accuracy in dynamic and changing data environments.

The SRP provides a well-rounded strategy for managing dynamic changes in data streams by utilizing a variety of ensemble techniques. However, the **Kappa Updated Ensemble** (KUE) [22] addresses the need for greater precision and adaptability by introducing a method that adjusts ensemble components based on performance metrics using the Kappa statistic. This method improves the balance between stability and responsiveness. KUE employs the Kappa statistic to dynamically allocate weights and select base classifiers, ensuring that only those that have a positive impact on the ensemble's performance are retained. KUE demonstrates the capability to consistently and efficiently adapt to evolving data streams, hence guaranteeing robustness against concept drift. The application of KUE entails employing diversification approaches for base learners, where each classifier is trained on a unique subset of data. This enhances the ensemble's capacity to draw generalizations and mitigates overfitting. Furthermore, this method is employed wherein classifiers have the ability to abstain from voting when they lack confidence, hence enhancing the accuracy and reliability of the ensemble.

Despite prior approaches that have made substantial progress in handling concept drift in data streams, they frequently encounter difficulties regarding inefficiency and instability when working with big or very small data chunks. This can result in either overlooking concept drifts or being overly responsive. **Broad Ensemble Learning System** (BELS) [8] overcomes these constraints by developing an ensemble learning system that uses small chunks for more stable and

efficient model updates. BELS aims to address past models' inefficiencies by employing a novel updating process that maximizes accuracy and stability while avoiding the restrictions of learning data individually or utilizing too large data chunks. The BELS system utilizes a unified feature mapping and enhancement layer for all the data, and the ensemble is composed of instances of the output layer. Each instance of the output layer is composed of many output nodes. By employing this method, the model functions optimally as the feature mapping and enhancement layers require only a single update at each timestep, rendering an ensemble of these layers superfluous. Each instance's accuracy performance determines the addition or removal of each instance in the ensemble's output layer. Every instance that is deleted is stored in a pool and has the potential to be reintroduced into the learning process if its accuracy exceeds a certain level. This technique not only preserves a high level of accuracy but also rapidly adjusts to changes in concepts. This approach exhibits substantial enhancements in managing idea drift, especially in situations involving imbalanced data streams, highlighting its robustness against noise and drifts.

Lastly, the prior techniques such as BELS and SRP have made substantial progress in dealing with idea drift in data stream environments, they often face challenges in effectively managing the trade-off between the diversity and accuracy of ensemble components. They also may be unable to react rapidly to sudden changes in data distribution. **Dynamic Ensemble Diversification** (DynED) [1] tackles these problems by employing a unique approach that prioritizes enhancing prediction accuracy through the dynamic selection of the most pertinent ensemble components, taking into account their performance and diversity. This guarantees that the classifiers maintain a high level of accuracy even when exposed to concept drift. DynED's ability to delete unproductive or extraneous components, using the Maximal Marginal Relevance (MMR) criterion for component selection, ensures that the ensemble's efficiency and effectiveness continue. It dynamically modifies the diversity value in response to the intensity of changes in accuracy.

After reviewing the GPM methods, we now look at ways to deal with class imbalances in non-stationary data streams. These Imbalance-Specific Methods

(ISM) use novel approaches to maintain effectiveness and accuracy across class distributions.

Table 2.1: Summary of data stream methods used in this study for the classification of data streams (Related to Ch. 2, and Ch. 4).

Method	Proposed Year	Domain	Imbalanced Approach	Learning Method	Concept Drift
OzaBagAdwin [16]	2009	GPM	-	Online	Explicit
LB [15]	2010	GPM	-	Online	Explicit
ARF [40]	2017	GPM	-	Online	Explicit
SRP [41]	2019	GPM	-	Online	Explicit
KUE [22]	2020	GPM	-	Chunk	Implicit
BELS [8]	2023	GPM	-	Chunk	Implicit
DynED [1]	2023	GPM	-	Online	Explicit
HDVFDT [28]	2008	ISM	Hellinger Distance Splitting Criteria	Online	Implicit
GHVFDT [68]	2014	ISM	Gaussian Hellinger Splitting Criteria	Online	Implicit
MUOB [88]	2016	ISM	Undersampling	Online	Implicit
MOOB [88]	2016	ISM	Oversampling	Online	Implicit
ARFR [18]	2019	ISM	Resampling	Online	Explicit
CSARF [66]	2020	ISM	Misclassification Cost	Online	Explicit
CALMID [63]	2021	ISM	Sample Weight & Uncertainty Strategy	Online	Explicit
ROSE [23]	2022	ISM	Self-adjusting Bagging & Per Class Sliding Window	Online	Explicit
MicFoal [64]	2023	ISM	Sample Weight & Uncertain Label Request Strategy	Hybrid	Implicit
LSH-DynED [This Study]	2024	ISM	Locality Sensitive Undersampling	Online	Explicit

GPM: General Purpose Methods, ISM: Imbalance-Specific Methods, Hybrid: Online + Chunk.

2.1.2 Imbalance-Specific Methods (ISM)

The ISM methods are specifically developed to address class imbalance in data streams, using a range of strategies, unlike the GPM methods. In the following, we thoroughly examine those equipped with these techniques or built explicitly for this purpose.

The **Hellinger Distance Very Fast Decision Tree** (HDVFDT) method [28] is specifically developed to tackle the issue of learning from imbalanced data streams, a prevalent difficulty in various supervised learning applications. In such cases, conventional algorithms favor larger and more dominant classes. This paper presents and assesses the Hellinger distance [48, 74] as a new, robust splitting criterion for decision tree learning unaffected by skewness. The Hellinger distance

is a reliable metric for quantifying the divergence between feature value distributions. Unlike traditional criteria such as information gain [73] and the Gini index [20], it is unaffected by any skewness in the class distribution. This technique aims to enhance the precision and equity of decision tree models, offering a resilient solution to the difficulties presented by imbalanced data streams.

The HDVFDT algorithm greatly enhances the management of imbalanced data streams, namely by improving the recall rates for the minority class. However, it has difficulties regarding memory efficiency and computing performance while handling large-scale data streams. These constraints can impede its practical implementation in real-time applications where fast processing is crucial. The **Gaussian Hellinger Very Fast Decision Tree** (GHVFDT) [68] solves restrictions by integrating an enhanced variation of the Hellinger distance, which measures the dissimilarity between normal distributions, resulting in better computational efficiency. The GHVFDT is ideal for high-speed data streams in which memory consumption and processing performance are critical. The GHVFDT, similar to its previous version, prioritizes enhancing recall rates for the minority class while minimizing false positives. The algorithm successfully balances sensitivity towards the minority class and overall accuracy by utilizing statistical metrics of data distribution to make educated decisions at each node in the tree. Furthermore, the GHVFDT algorithm improves the management of imbalanced class distributions and consistently achieves high G-Mean values in diverse levels of class imbalance, demonstrating its resilience in various streaming settings. This enhancement in memory efficiency does not undermine the classifier's capacity to handle the difficulties presented by imbalanced learning in data streams.

Previous methods in imbalanced streams, particularly for multi-class scenarios, have struggled with issues like dependency on initial samples for class decomposition and a lack of adaptivity to dynamic class distributions. This often leads to less effective handling of skewed class distributions in real-time online learning environments. Two new ensemble learning methods, **Multi-class Oversampling-based Online Bagging** (MOOB) and **Multi-class Undersampling-based Online Bagging** (MUOB) [88], are introduced to overcome these issues. Both

approaches are intended to process multi-class data streams directly and adaptively without the use of class decomposition or initial samples. To successfully balance class distributions, they employ adaptive resampling techniques [88], whereby MUOB applies undersampling, and MOOB applies oversampling.

MOOB and MUOB differ in their approach to managing class imbalances: MOOB focuses on generating additional instances of the minority classes, whereas MUOB reduces the presence of majority class instances. This enables both strategies to maintain updated and balanced class representations while adjusting to changes in class distribution as they occur. They also provide a time-decayed class size metric that represents the current state of class imbalances, allowing for adaptive sampling rates. In comparison, MOOB performs better and more consistently across numerous dynamic circumstances, particularly in sustaining stronger recall for minority classes without affecting overall accuracy. This is most likely owing to its strategy of increasing the presence of minority classes rather than just diminishing the majority class, which can be vital in contexts where minority class samples are especially valuable or predictive of critical outcomes.

The **Adaptive Random Forest with Resampling** (ARFR) [18] is a method developed to address the constraints of imbalanced streams by improving upon the conventional ARF [40] approach. ARFR presents a resampling strategy that modifies instances' weights during training according to the current class distribution. By utilizing adaptive resampling, the importance of the minority class is enhanced, resulting in improved visibility and the classifier's enhanced accuracy in predicting it. Each instance in the data stream is dynamically weighted, increasing the representation of minority classes and ensuring continual adaptation to changes in class distribution. Moreover, ARFR has strategies to manage concept drift, essential for preserving model relevance in dynamic situations. The approach also demonstrates enhanced computational efficiency, substantially reducing CPU time compared to classical ARF. Efficiency is crucial for applications that necessitate real-time data processing and decision-making.

Although the ARF and ARFR adapt to data streams, they usually struggle

to adequately handle class imbalances, particularly in circumstances where misclassification of minority classes could have serious consequences. ARF prioritizes responding to changes in data streams over expressly considering class imbalance. At the same time, ARFR tries to address imbalance issues by employing resampling algorithms that do not directly incorporate the cost of misclassification into the learning process.

The **Cost-sensitive Adaptive Random Forest** (CSARF) [66] addresses these constraints by introducing misclassification costs into the learning process, with a specific focus on the difficulties posed by imbalanced streams. CSARF improves on the standard ARF framework [40] by using the MCC (Matthews Correlation Coefficient) [27] to provide weight to trees. This approach offers a more refined evaluation of classifier performance, which is particularly advantageous in situations when there is an imbalance in the stream. Furthermore, CSARF has a sliding window method to consistently monitor changes in class distribution, guaranteeing that the training process is adaptable to the ever-changing nature of data streams and the inclusion of minority class samples.

Previous techniques, such as KUE [22], have made considerable progress in dealing with concept drift in data streams. However, they often struggle to handle non-stationary class imbalances effectively. KUE prioritizes the utilization of the Kappa statistic for weighting and selecting classifiers. Although this strategy is effective at adjusting to changes in data drift, it may not always provide the flexibility required to respond swiftly to sudden shifts in class distribution and imbalance. **Robust Online Self-Adjusting Ensemble** (ROSE) [23] is a novel online ensemble classifier designed to address imbalanced and drifting data streams, attempting to overcome these limitations. ROSE seeks to address the deficiencies of prior approaches by improving adaptation and resilience to changes in concepts and class distribution imbalances, all without human intervention. The system includes innovative technical improvements, such as variable-size random feature subspaces for each classifier, which increase diversity and resilience to drift. Furthermore, ROSE utilizes a background ensemble to facilitate rapid adaptation by monitoring and replacing underperforming classifiers based on their response to drift.

Furthermore, ROSE implements a distinctive self-adjusting bagging method that adapts its strategy according to the existing class distribution, specifically enhancing the representation and classification precision of underrepresented classes. This is aided by using per-class sliding windows, which store a current selection of cases for each class. This guarantees that the classifier is constantly updated with the most relevant data. ROSE has a collection of capabilities that allow it to handle fast changes in data streams properly. It is able to retain strong performance even when faced with limited availability of labels and significant imbalances between classes.

A new approach, **Comprehensive Active Learning Method for Multi-class Imbalanced Data Streams with Concept Drift** (CALMID) [63], has been developed to address the limitations mentioned earlier. CALMID uses a resilient framework to improve the management of imbalances in multi-class scenarios while reducing concept drift. The sample weight formula used in this method considers the dynamic class distribution and the specific difficulties associated with each class. In addition, CALMID employs an asymmetric margin threshold matrix for its uncertainty strategy, enabling more accurate and adaptable management of classification uncertainties across various classes. In addition, CALMID implements a novel initialization procedure for newly introduced basic classifiers, guaranteeing their enhanced readiness to handle the dynamic data streams right from the beginning. This strategy enhances the resilience of the learning process against changes in data patterns and also maximizes performance within limited resources for labeling.

Although methods such as CALMID and ARFR are creative in tackling class imbalance and concept drift in data streams, they have limits, especially in situations where the dynamics are very unpredictable. CALMID, for example, may not adequately adjust to situations when there is a sudden change in class dominance or the emergence of new classes. ARFR primarily addresses imbalanced data by utilizing resampling techniques. However, it may not effectively respond to quick changes in class features or the development of new classes. The **MicFoal** framework [64] is designed to overcome these limitations by offering a reliable

solution for imbalanced stream classification that efficiently handles multiclass imbalance and concept drift. MicFoal is a system specifically designed to manage the ever-changing nature of data streams. It is capable of adapting to situations where previous classes may no longer exist, and new ones may form. MicFoal includes a configurable supervised learner and uses a hybrid label request method, which provides flexibility in the process of active learning. This technique is backed by an adaptive mechanism that modifies the budget allocated for label costs according to current performance evaluations, guaranteeing the most efficient application of resources.

In addition, MicFoal presents a new and uncertain approach to requesting labels, utilizing a changeable least confidence threshold vector. This strategy is particularly effective in dealing with the fluctuating number of classes over time. MicFoal can adapt to changes in the data stream, ensuring that it stays accurate and efficient. MicFoal has greater flexibility and adaptability than CALMID and ARFR. It allows for easy modification and tuning of its components to accommodate the complex nature of real-world data streams effectively.

According to the literature review we provided here, while Imbalance-Specific Methods (ISM) have been shown to be effective in managing class imbalances in data streams, they frequently suffer responsiveness to rapid changes in class distribution. Our study introduces a novel undersampling technique that uses randomly generated hyperplanes and hash codes to enhance classification accuracy in multi-class imbalanced data.

2.2 Sampling Methods

Sampling techniques try to balance data samples before using them to train classifiers. Several sampling techniques have been proposed to either oversample or increase data samples in the minority class(es) or undersample to eliminate some data samples from the majority class(es).

2.2.1 Oversampling Techniques

The most straightforward oversampling technique randomly reproduces data items from the minority class. A more complex technique, **Synthetic Minority Oversampling Technique** (SMOTE) [25], aims to generate synthetic examples rather than simple replication of minority class instances by interpolating between the two samples, ensuring the minority class decision region is more generalized. **SMOTEBoost** [26] is proposed to further improve prediction for the minority class in imbalanced datasets by integrating SMOTE with the boosting algorithm to create synthetic minority class examples during each boosting iteration. **Borderline-SMOTE** [45] is proposed to improve the predictive performance for the minority class by only oversampling the minority class samples near the decision boundary. **Geometric SMOTE** [33] improves the oversampling by generating synthetic samples within a geometric region around each selected minority sample, allowing the region to be a hyper-sphere, hyper-spheroid, or line segment.

2.2.2 Undersampling Techniques

The simplest undersampling technique is to eliminate samples from the majority classes randomly. **One-Sided Selection** (OSS) [55], a more complex technique, removes noisy and borderline majority class samples using Tomek links [82] to perform undersampling tasks. The **Neighborhood Cleaning Rule** (NCL) [59] is a technique designed to enhance the identification of small and difficult-to-classify classes within imbalanced datasets. NCL emphasizes data cleaning over data reduction by incorporating Wilson's **Edited Nearest Neighbor** (ENN) [81] rule to remove noisy and borderline instances from the majority class. **Cluster-based Undersampling** [91] clusters the entire dataset and selectively undersamples the majority class based on cluster characteristics, ensuring a more representative subset of the data. **Topics Oriented Directed Undersampling** (TODUS) [77] leverages topic modeling to estimate the latent data distribution and generate a balanced training set based on their estimated importance, directing the

undersampling process to minimize information loss.

In summary, oversampling can lead to overfitting problems, noisy and false instances that do not really belong to the minority class, and, ultimately, over-generalization of the minority class into the majority class region, thereby lowering overall prediction performance, whereas undersampling can cause information loss [35, 70, 80].

Chapter 3

Proposed Approach: LSH-DynED

In this chapter, we introduce a novel method, LSH-DynED, designed to address the challenges posed by multi-class imbalanced and non-stationary data streams. We begin by outlining the DynED method in Section 3.2. Next, we introduce the concept of Locality Sensitive Hashing with Random Hyperplane Projections (LSH-RHP) in Section 3.3. Finally, we present our proposed approach, the LSH-DynED method, in Section 3.4.

3.1 Benefits of the Approach

This integration aims to further extend the previous method's ability to manage class imbalances, make it resilient to dynamic class ratios, and protect it from concept drifts. By combining LSH-RHP in DynED, we can better manage the ensemble components, introducing extra diversity among them while considering the features of the real-time data stream. This adaptation consists of two major parts. First, it chooses dynamic components that strike a balance between accuracy and diversity. Second, it constantly updates the ensemble's training data to

ensure that the components are trained on a balanced yet diverse set of data.

The key feature of our approach is the targeted undersampling of the majority classes using the LSH-RHP technique. By dividing the data into hash-based partitions, we can target and reduce the representation of over-represented classes without losing critical information, resulting in a balanced training dataset. Additionally, repeatedly performing LSH-RHP provides a diverse set of samples from majority classes, leading to higher component diversity. This targeted undersampling is critical for ensuring that minority classes are adequately represented and playing a crucial role in robustness to class imbalance challenges, resilience to dynamic imbalance ratios, and improving the ensemble's overall predictive accuracy.

3.2 DynED Overview

The DynED [1] is a dynamic ensemble method designed to mitigate the challenges posed by concept drift in data stream classification. It operates in three stages: (1) Prediction and Training, (2) Drift Detection and Adaptation, and (3) Component Selection.

Prediction and Training: At this stage, DynED employs a subset of its ensemble components, known as the "selected components," to predict the label of the incoming data sample. This prediction is made via a majority voting. Following the prediction, the "selected components" are trained on the new data instance.

Drift Detection and Adaptation: This stage focuses on detecting and adapting to concept drift. The drift detector, ADWIN [14], is constantly updated with the system's correct/incorrect predictions. If a drift is detected, a new component is added and trained on the most recent data samples stored in a sliding window. This new component is then added to a pool of "reserved components." If it is added or the number of processed samples exceeds the θ threshold, then a parameter λ, which controls the balance between diversity and accuracy in the

ensemble, is dynamically adjusted based on the intensity of changes in the accuracy of the ensemble over time and stage 3 is activated to update the "selected components."

Component Selection: The purpose of this stage is to update the ensemble's components to maintain a balance between diversity and accuracy. The selected components from the previous stage are combined with the reserved components. This combined pool is then sorted based on component accuracy, and any extra components that exceed a predetermined pool size are deleted. The remaining components are clustered according to their prediction errors, and a predetermined number of high-performing components are chosen from each cluster. Finally, a modified version of the MMR (Maximal Marginal Relevance) algorithm [1] is used in the final selection of components.

This technique chooses a new set of components by considering both accuracy and diversity and ensures that the ensemble remains accurate while adapting to changes in the data stream. Newly picked components are designated as "selected components" to replace the active components, resuming the prediction stage, while the remaining components are returned to the "reserved pool." Table 3.1 shows the size limitations for these pools.

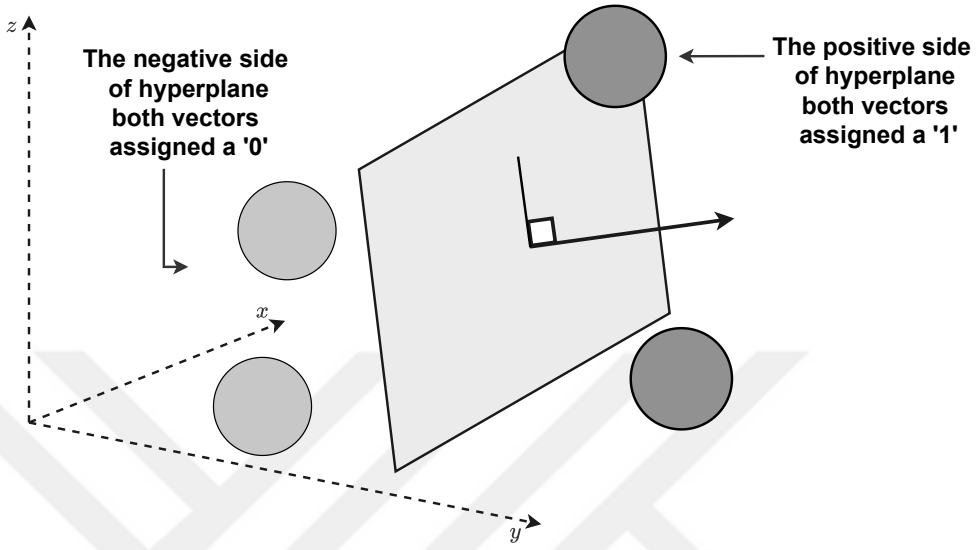


Figure 3.1: LSH-RHP: If the data sample is on the positive side of the hyperplane, the sign operator assigns a '1', and if the data sample is on the negative side, the sign operator assigns a '0' to it. (Related to Sec. 3.3)

Table 3.1: LSH-DynED notation and applicable default hyperparameter values. (Related to Sec. 3.2, Sec. 4.1.3, and Sec. 4.1.5).

Symbol	Meaning	value	Symbol	Meaning	value
ξ_{slc}	Set of selected components	N/A	θ	Threshold for the count of processed samples	1000*
cls_{rst}	Set of not selected components based on clustering	N/A	Δ_λ	Variation in λ parameter	0.1*
Id_c	Class identifier	N/A	S_p	Maximum size of the component pool	500*
σ	A data sample	N/A	S_{cls}	# of components to select from each cluster	10*
S_i	Sample of indice i	N/A	n_{train}	# of train samples	20
D	Data stream	N/A	λ	Controlling parameter of similarity and accuracy	0.6*
MW	Multi-sliding window	N/A	S_w	Window size	1000
DD	Drift detector	N/A	n_v	Number of vectors	5
cls_{slc}	Set of selected components based on clustering	N/A	S_{slc}	# of active components for classification	10
S_c	Sample of each class	N/A	n_{test}	# of test samples	50
D_n	A set of data samples from all classes	N/A			
i	Indice of a sample in sliding window	N/A			

* These values are provided by DynED [1] based on grid search. S_w is set to 1000 [4], with the aim of conducting a fair comparison across all methods.

3.3 Locality Sensitive Hashing with Random Hyperplane Projections (LSH-RHP)

Locality Sensitive Hashing (LSH) [65] with Random Hyperplane Projections (RHP) is an effective method for approximate nearest-neighbor search in high-dimensional data analysis. The method uses a series of hash functions known as the Multilinear Hash Function (MH), which calculates binary codes using multiple linear projection vectors [65]. Combining these binary codes results in a hash code referred to as buckets, in which data points are distributed across them. The data points with similar characteristics tend to have similar hash codes and be placed in the same bucket. One can think of these buckets as locally dense areas [24]. The fundamental principle entails the process of projecting data points into a space with fewer dimensions by utilizing these vectors, which are defined as follows

$$\mathbf{h}_m(\mathbf{x}) = \text{sgn}(\mathbf{u}_1^T\mathbf{x}, \ldots, \mathbf{u}_m^T\mathbf{x}) \tag{3.1}$$
$$\begin{cases} \mathbf{u}_1^T\mathbf{x} > 0 & \text{sgn}(\mathbf{u}_1^T\mathbf{x}) = 1 \\ \mathbf{u}_1^T\mathbf{x} < 0 & \text{sgn}(\mathbf{u}_1^T\mathbf{x}) = 0 \end{cases}$$

where each $\mathbf{u}_l$ is a vector sampled from a normal distribution $\mathbf{u}_l \sim N(0, I_{d\times d})$ and $\mathbf{m}$ indicates the number of these vectors and $\mathbf{d}$ stands for the dimensionality of the data samples indicating each component of the vector $\mathbf{u}_l$ is independently and identically distributed with a variance of 1 with no covariance between components. This method effectively reduces the computational complexity and dimensionality by preserving the angle distance between points [65]. Figure 3.1 visually illustrates the equation described in 3.1.

Although the applications of approximate nearest-neighbor searches, including LSH-RHP, are in different domains, such as large-scale visual search, recommendation, and similar query search [51, 65], it is possible to utilize this technique

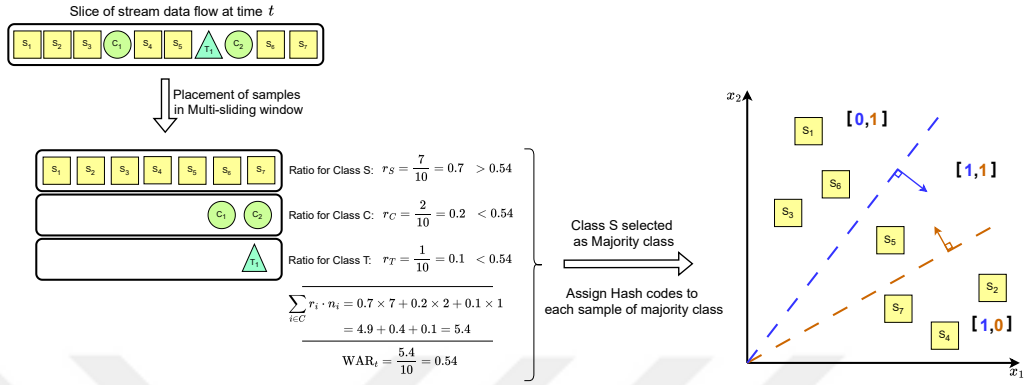


Figure 3.2: Simple example of performing Locality Sensitive Hashing with Random Hyperplane Projection. (Related to Sec. 3.3).
We assumed this stream contains two features, x_1 and x_2, and three classes: Square (S), Circle (C), and Triangle (T). The Square class is the majority class; thus, the LSH-RHP is performed on it. In the case of multiple majority classes, this process is applied to all of them

to divide the data space into smaller subspaces [47, 60]. Each subspace contains samples with similar properties but different from others. This division allows for targeted resampling, especially for the majority classes in multi-class data streams. LSH functions as a filter that groups comparable data points in buckets. Consequently, we are able to select representative samples from each bucket, which substantially reduces the size of the majority classes and preserves feature space information. Furthermore, the randomized initialization of the vectors in the hash function results in added randomization while building buckets, identifying different dense areas, and preventing the selection of the same items repeatedly, as opposed to clustering approaches. As a result, this method is a novel solution for overcoming the obstacles presented by imbalanced data streams in multi-class classification tasks, where real-time adaptation is essential for preserving the accuracy and performance of the model. Figure 3.2 presents a simple example of this technique.

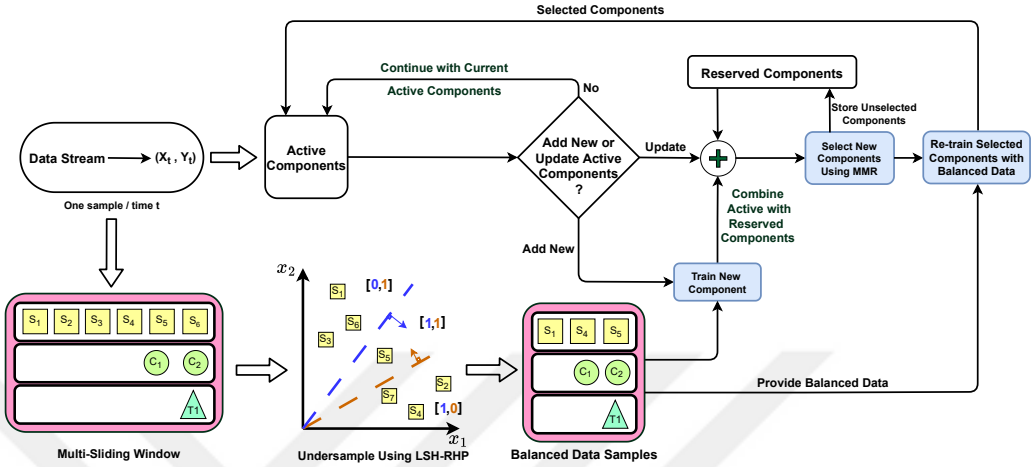


Figure 3.3: LSH-DynED: Working Principle of Integrating Locality Sensitive Hashing (LSH) with Random Hyperplane projection (RHP) into DynED. (Related to Sec. 3.4).
Assumed that data stream contains two features, x_1 and x_2, and three classes: Square (S), Circle (C), and Triangle (T). Active components predict the label of data samples at time t, while multi-sliding windows retain recent class-specific samples. Upon ensemble expansion, new components are trained on balanced datasets generated through undersampling. Component selection employs the third stage of the Dyned approach. Selected components undergo retraining on balanced datasets before replacing active components.

3.4 LSH-DynED

After introducing DynED and LSH-RHP, which serve as the basis of our study, we now explain our proposed approach.

To effectively address the challenges posed by imbalanced data streams, DynED's process must be extended, and the advantages of LSH-RHP must be incorporated to make implementing a more informed undersampling approach for majority classes feasible. It is worth noting that DynED is a general purpose method (GPM) that does not have any specific capabilities for handling imbalanced data steam. This strategy aims for the ensemble's training to include evenly distributed samples, effectively reducing the bias towards the majority classes in the stream. This aims to leverage the strengths of both methods, thereby creating a more robust and scalable classification method.

Current methods in the literature often utilize a single sliding window as a buffer, containing a fixed-size sample for training the components of an ensemble. However, due to inherent imbalances in the data stream, this approach inadvertently introduces a bias towards the majority classes, as highlighted by Cano et al. [23]. The first adaptation of our approach is the use of multi-sliding windows described in algorithm 1, which is required by the LSH-RHP technique. Specifically, each class allocates its own sliding window to buffer the most recent data samples of a fixed size (Class 1, lines 6-16).

This adaptation addresses the challenge of availability; classes with fewer samples will see their respective windows refreshed less frequently compared to the majority classes. As a result, some data samples from minority classes are consistently available. To correctly choose the majority classes, we continuously calculate the weighted average of class ratios (Class 1, line 26) using the following equation:

$$\textbf{W}\text{eighted }\textbf{A}\text{verage of }\textbf{R}\text{atios } (WAR_t) = \frac{\sum_{i \in C} r_i^t \cdot n_i^t}{\sum_{i \in C} n_i^t} \tag{3.2}$$

Where C represents the set of all classes in the stream, with r_i^t denoting the ratio and n_i^t the sample count for class i up to time t and they are subject to change dynamically over time t. The WAR_t can range in value from 0 to 1. A class whose ratio exceeds the WAR_t is considered a majority class and a candidate for undersampling. As shown in Figure 4.1, the WAR_t tends to converge and approach a steady state. This trend suggests that initial fluctuations in class distribution become less volatile as more data is processed and ensures our undersampling technique is dynamically adjusted to prevent minority classes from being underrepresented, improving model performance. After determining the majority classes, we set up the hash function with a certain number of vectors n_v. Next, the buckets are created. The samples are then drawn randomly from each bucket based on the number of samples in each. However, the total number of samples for each majority class does not surpass a certain preset limit n_{train}. By selectively undersampling the majority classes (Algorithm 1, line 27-32), we focus on a subset of samples that encompasses the feature space of those classes, resulting in more balanced training data for the ensemble components.

In the next step, we incorporate the previously introduced undersampling method into the second stage—Drift Detection and Adaptation. Thus, this stage functions as follows:

Initially, as detailed in Section 3.2, the drift detector is updated (Algorithm 1, line 6). Upon detecting drift, a new component is created and trained with a balanced and fixed sample size (Algorithm 1, lines 8-10), using the method outlined in Section 3.3. This procedure ensures the new component is trained on balanced samples, eliminating bias towards any class. Besides, in the case of several consecutive runs, this technique generates an entirely different set of balanced training data to help increase diversity among the ensemble's components. Following this, the component is added to a pool of "reserved components."

If a new component is trained or if the number of processed samples surpasses the θ threshold, then the parameter λ is dynamically adjusted (Algorithm 1, lines 13). This adjustment is based on changes in the ensemble's accuracy over time, leading to the activation of the third stage—Component Selection.

Class 1 LSH-DynED: MultiSlidingWindow-Undersampling Class.

Class: Multi-SlidingWindow-Undersampling
Properties:
    window_size: Integer ▷ Maximum S_w per class
    class_list: List ▷ Identifiers Id_c for each class
    window: Dictionary ▷ Stores S_c for each class
method INITIALIZE (w_{size}, Id_c)
    window_size $\leftarrow S_w$
    class_list $\leftarrow Id_c$
    window $\leftarrow$ Initialize dictionary with keys in Id_c
end method

method ADDTOWINDOW (σ, *class_id*)
    if window[*class_id*] is $> S_w$ **then**
        Remove oldest sample from window[*class_id*]
    end if
    Add σ to window[*class_id*]
end method

method GETDATAFORPREDICTION (N_{text}) ▷ Retrieve test samples
    Prepare $\mathcal{D}_N$ by selecting the last N_{text} samples from each class
    Shuffle and return $\mathcal{D}_N$ and corresponding labels
end method

method FETCHDATAPOINTS (*class_id*, *i*)
    Retrieve specific samples S_i based on indices i from window[*class_id*]
    return S_i
end method

method UNDERSAMPLEDATAFORTRAINING (N_{train}) ▷ Retrieve balanced train samples
    Identify the majority classes (*class_id*) by calculating the WAR_t using Eq. 3.2
    for Each Majority class **do**
        Initialize the n_v vectors of the hashing function
        Get the buckets using the LSH-RHP strategy
        Specify N_{train} indices i from buckets randomly
        Fetch the data byFETCHDATAPOINTS method and put into D_N
    end for
    Shuffle and return D_N the selected data and labels
end method

Algorithm 1 LSH-DynED: Extended main flow.

Require: D, MW, DD, S_slc, S_w, S_p, N_{train}, Δ_λ
Ensure: $\hat{y_k}$: Prediction of each sample at time t

1: $\xi_{slc} \leftarrow$ Initialize slc_s components
2: **while** D has more samples **do**
3: $sample \leftarrow$ Get next sample from D
4: $\hat{y_k} \leftarrow$ Use ξ_{slc} to predict using majority voting
5: MW.AddToWindow($sample$) ▷ Add sample to the sliding window
6: $DD \leftarrow$ Update drift detector
7: Train ξ_{slc} on $sample$
8: **if** DD detects drift **then**
9: $Data, Labels \leftarrow MW$.UndersampleDataForTraining(N_{train})
10: $\xi \leftarrow$ Generate and train a new component on $Data, Labels$, add to the pool
11: **end if**
12: **if** passed samples count $\geq \theta$ or new component is added **then**
13: Update λ with Δ_λ step size
14: $\xi_{slc}, \xi \leftarrow$ Update ξ_{slc} and ξ using Algorithm 2
15: **end if**
16: **end while**

In the updated third stage, we introduced another adaptation, whereas DynED traditionally utilizes accuracy and double-fault (df) diversity measures [56, 57, 83] to select components. To address the imbalance problem more effectively, we replaced the previously used diversity measure with the kappa statistic [84] (Algorithm 2, lines 9-10). Additionally, suppose the component count exceeds the predefined limit within the pool. In that case, they are sorted by their kappa statistic values, and the least performing component is removed (Algorithm 2, line 2).

A crucial aspect of the DynED workflow is the online training of "selected components" for incoming samples. Given the imbalanced nature of the data stream, this training typically introduces a bias towards the majority class. To mitigate this bias, we enhanced this stage by incorporating additional training sessions for newly "selected components" using balanced data samples obtained through our undersampling method, an element absent in the original DynED framework (Algorithm 2, lines 12-13). Figure 3.3 provides an abstract representation of the proposed method's workflow.

Algorithm 2 LSH-DynED: Extended component selection.

Require: MW, ξ, ξ_{slc}, S_{slc}, S_p, S_cls, N_{train}, N_{test}

Ensure: ξ, ξ_{slc}

$P \leftarrow \xi + \xi_{slc}$

Sort P by Kappa statistic and limit to S_p

$Data, Labels \leftarrow MW.\text{GetDataForPrediction}(N_{test})$

$PE \leftarrow$ Calculate prediction error for each component using $Data$ and $Labels$

Perform KMeans clustering on PE with two clusters

$cls \leftarrow$ Labels from KMeans clustering

$cls_{slc} \leftarrow$ Select top S_{cls} components from each cluster based on highest accuracy

$cls_{rst} \leftarrow P$ excluding cls_{slc}

$Diversity \leftarrow$ Calculate kappa metric for components in cls_{slc}

ξ_{slc}, $\xi_{rst} \leftarrow$ Apply MMR on cls_{slc} to select S_{slc} based on diversity and accuracy

$\xi \leftarrow cls_{rst} + \xi_{rst}$ ▷ All other components remain in reserve

$Data, Labels \leftarrow MW.\text{UndersampleDataForTraining}(N_{train})$

$\xi_{slc} \leftarrow$ Re-train selected components on $Data$ and $Labels$

return ξ, ξ_{slc}

3.5 Time Complexity Analysis

As noted in [1], the time complexity of the DynED is approximated as $\mathcal{O}(n^2)$, where the n stands for the number of components in the ensemble. The time complexity of the LSH-RHP involves the following components: we only utilize LSH-RHP to build buckets and divide the feature space of each class into portions, and we skip the nearest neighbor finding step. Thus, the time complexity of generating a hyperplane is $\mathcal{O}(d)$ if the dataset includes d-dimentional feature space and it repeats this process k times to generate k hyperplanes. So far, the time complexity has become $\mathcal{O}(kd)$. For the worst-case scenario, we assume that the majority class's sliding window is full and contains v samples. Therefore, the time complexity of obtaining the hash code for samples of that class is $\mathcal{O}(vkd)$. This step is repeated for c times, where c represents the number of majority classes determined by WAR_t (Eq. 3.2) and the time complexity becomes $\mathcal{O}(cvkd)$. Thus, the overall time complexity of the method is $\mathcal{O}(n^2 \times cvkd)$.

Chapter 4

Experimental Evaluation

In order to demonstrate the effectiveness of our proposed method and compare it, we conducted a thorough experimental evaluation, considering many criteria. In the following, we first describe the experimental environment (Sec. 4.1) and then provide our results on effectiveness (Sec. 4.2.1), statistical analysis (Sec. 4.2.2), impact of dataset characteristics (Sec. 4.3.1), impact of dynamic imbalance ratios (Sec. 4.3.2), ablation analysis (Sec. 4.3.3), and overall evaluation of experimental observations (Sec. 4.4).

4.1 Experimental Environment

This section introduces the experimental environment of our study, which encompasses the baselines, datasets, experimental settings, and the effectiveness measures employed for comparison.

4.1.1 Baselines

The experiments use 15 classification methods as baselines capable of handling multi-class non-stationary data streams. Six of them are high-performing general purpose methods (GPM), while the remaining nine are specifically designed to address the issues of imbalanced data streams (ISM). The reason for including GPM methods in our evaluation is that although they are not explicitly equipped with imbalance handling strategies, they demonstrate robust performance in such environments. The characteristics of these methods are detailed in Table 2.1. We employed the publicly accessible source code on GitHub[1], which is created by Aguiar et al. [4] on top of the MOA framework [17] to run our experiments. Additionally, the Python implementation of BELS [8] is available publicly on the author's GitHub[2]. Our proposed method is implemented in Python 3.11.7 using the River library 0.21.1 [69], and the Faiss library 1.7.4 [34], with a Hoeffding Tree as the base classifier. The source code for our method is also publicly available on GitHub[3] with all necessary details to ensure the reproducibility of our experiments.

4.1.2 Datasets

To assess the performance, robustness, and resilience of the proposed method relative to established baselines, we selected 33 datasets, detailed in Table 4.1, which include 23 real datasets commonly used in the data stream domain and publicly accessible from UCI repository [49], KEEL repository [31] and Kaggle competitions and ten semi-synthetic data streams as described by Korycki et al. [54]. For some datasets, the presence of concept drift is explicitly noted, whereas those marked as "Unknown" should not be assumed to be stationary or devoid of concept drift. The semi-synthetic streams are specifically designed for scenarios where class ratios fluctuate considerably, as depicted in Figure 4.1, making them ideal for testing methods resilience toward dynamic imbalance ratios

[1]Imbalanced-Streams
[2]BELS
[3]LSH-DynED

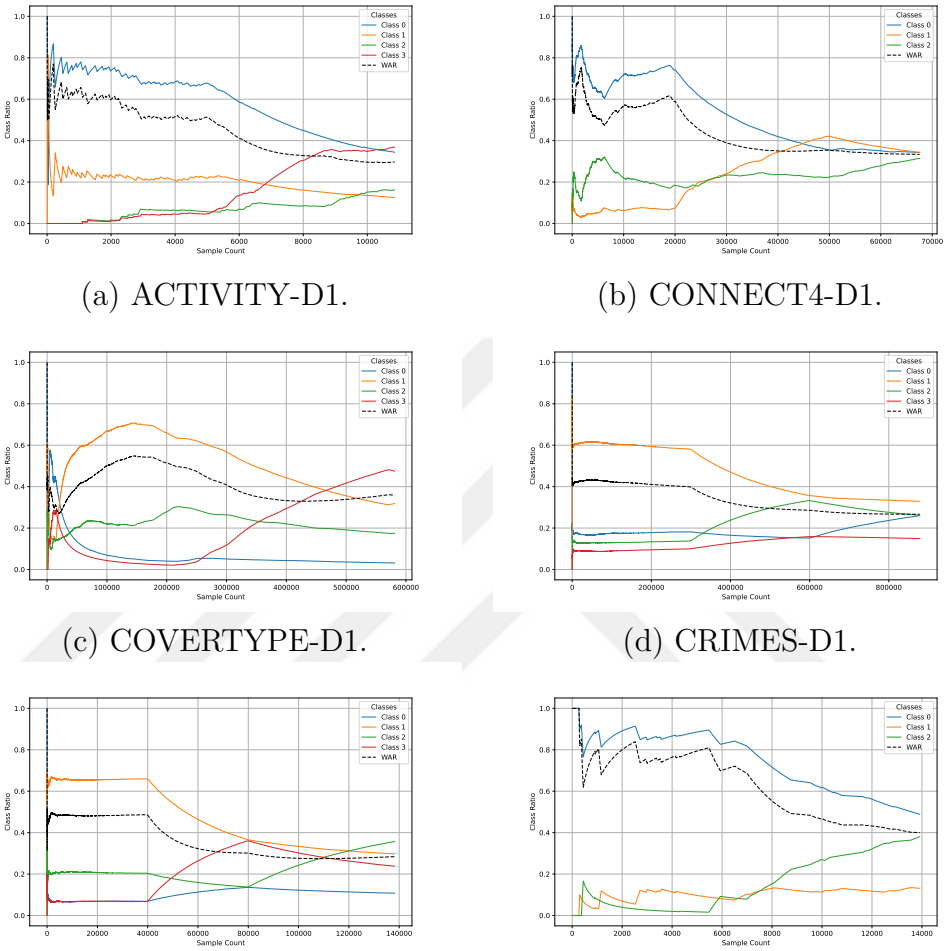
in uneven multi-class streams. We chose to examine both real-world and semi-synthetic datasets because they introduce new and challenging situations, such as confusing inter-class interactions. These conditions are crucial for understanding classifier performance under challenging scenarios. Furthermore, these datasets are curated to replicate specific behaviors and lack the apparent probabilistic features commonly found in stream generators [3].

Table 4.1: Datasets and their characteristics.(Related to Sec. 4.1.2, Sec. 4.3.1, Sec. 4.3.2, and Sec. 4.3.3)

Dataset	Instances	Features	Classes	Type	Max Ratio	Min Ratio	Dynamic Ratio	Drift
Activity	5,418	45	6	real	0.384090	0.045404	—	✓
Balance	625	4	3	real	0.460800	0.078400	✓	Unknown
Connect-4	67,557	42	3	real	0.658303	0.095460	—	Unknown
Contraceptive	1,473	9	3	real	0.427020	0.226069	✓	Unknown
Covtype	581,012	54	7	real	0.487599	0.004728	—	Unknown
Crimes	878,049	3	39	real	0.199192	0.000007	—	Unknown
Ecoli	336	7	8	real	0.425595	0.005952	✓	Unknown
Fars	100,968	29	8	real	0.417122	0.000089	—	Unknown
Gas	13,910	128	6	real	0.216319	0.118973	—	✓
Glass	214	9	6	real	0.355140	0.042056	✓	Unknown
Hayes-roth	132	4	3	real	0.386364	0.227273	✓	Unknown
Kr-vs-k	28,056	6	18	real	0.162283	0.000962	✓	✓
New-thyroid	215	5	3	real	0.697674	0.139535	—	Unknown
Olympic	271,116	7	4	real	0.853262	0.048378	—	Unknown
Pageblocks	548	10	5	real	0.897810	0.005474	—	Unknown
Poker	829,201	10	10	real	0.501116	0.000002	—	✓
Shuttle	57,999	9	7	real	0.785979	0.000172	—	✓
Tags	164,860	4	11	real	0.330462	0.008377	—	Unknown
Thyroid-s	720	21	3	real	0.925000	0.023611	—	Unknown
Thyroid-l	7,200	21	3	real	0.925833	0.023056	—	Unknown
Wine	178	13	3	real	0.398876	0.269663	✓	Unknown
Yeast	1,484	8	10	real	0.311995	0.003369	—	Unknown
Zoo	1,000,000	17	7	real	0.396504	0.042792	—	✓
ACTIVITY-D1	10,853	43	4	semi-syn	0.368193	0.125495	✓	✓
CONNECT4-D1	67,557	42	3	semi-syn	0.343340	0.313409	✓	✓
COVERTYPE-D1	581,012	54	4	semi-syn	0.474603	0.031383	✓	✓
CRIMES-D1	878,049	3	4	semi-syn	0.329192	0.149404	✓	✓
DJ30-D1	138,166	7	4	semi-syn	0.357078	0.107537	✓	✓
GAS-D1	13,910	128	3	semi-syn	0.488497	0.131416	✓	✓
OLYMPIC-D1	271,116	7	3	semi-syn	0.353432	0.321840	✓	✓
POKER-D1	829,201	10	4	semi-syn	0.340320	0.024806	✓	✓
SENSOR-D1	2,219,802	5	4	semi-syn	0.372768	0.090036	✓	✓
TAGS-D1	164,860	4	4	semi-syn	0.349945	0.092327	✓	✓

4.1.3 Experimental Setting

All methods are evaluated online using the interleaved-test-then-train approach [38], which is critical for determining the adaptability of models to new data as it arrives. In machine learning, choosing the right hyperparameters is critical since



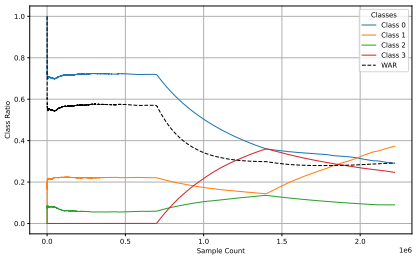
(a) ACTIVITY-D1. (b) CONNECT4-D1.

(c) COVERTYPE-D1. (d) CRIMES-D1.

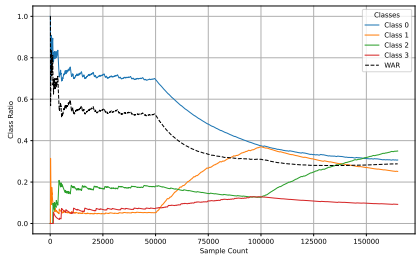
(e) DJ30-D1. (f) GAS-D1.

Figure 4.1: Graphical presentation of datasets with dynamic imbalance ratio. (First part)

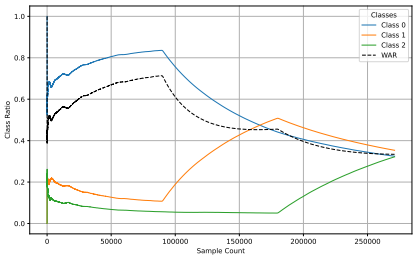
they significantly impact model performance. To guarantee a fair comparison across all methods, we used the suggested hyperparameters for each model as specified by the original authors. We hard-coded the chunk size to one in the BELS model to replicate the behavior of online models. We determined values after conducting tests with various hyperparameters based on the grid search method as described in Sec. 4.1.5. Table 3.1 details the hyperparameters for our proposed technique, which are not tuned to any specific dataset to preserve its broad applicability. The experiments are carried out on a server with an Intel(R) Xeon(R) Gold 5118 CPU @ 2.30 GHz, 128 GB RAM, and Ubuntu 18.04.4 LTS.



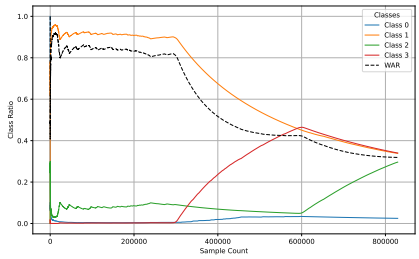
(i) SENSOR-D1.



(j) TAGS-D1.



(g) OLYMPIC-D1.



(h) POKER-D1.

Figure 4.1: Graphical presentation of datasets with dynamic imbalance ratio. (Continued)

Due to the fundamental differences between Java and Python, a direct comparison of computational performance between our implementation and the baseline methods described in Section 4.1.1 would be unreliable. The framework for baseline experiments is Java-based, while our proposed method is implemented in Python. This disparity in programming languages prevents meaningful comparisons of execution time and memory usage [2, 50].

4.1.4 Effectiveness Measures

To evaluate the methods, we utilized the Kappa [4, 21, 46] and mG-Mean (multi-class G-Mean) [27, 36, 37, 87, 88] metrics. They are both introduced to measure the classification effectiveness of multi-class classification methods in imbalanced environments.

The Kappa measures inter-rater reliability and is adjusted for chance agreement. It is calculated as shown in Eq. 4.1:

$$\text{Kappa} = \frac{n \sum_{i=1}^{c} x_{ii} - \sum_{i=1}^{c} x_{i.} x_{.i}}{n^2 - \sum_{i=1}^{c} x_{i.} x_{.i}} \tag{4.1}$$

Here, n is the total number of observations, and x_{ii} represents the instances where both raters agree on category i. The terms $x_{.i}$ and $x_{i.}$ denote the columns and row total classifications each rater makes into category i, respectively. The product of these terms provides the expected frequency of agreements by chance for each category. The numerator of Kappa calculates the excess of observed agreements over those expected by chance, indicating the actual agreement adjusted for randomness. The denominator represents the maximum possible excess of agreements over chance. The resulting value ranges from -1 to 1, where 1 indicates perfect agreement, 0 suggests chance-level agreement and negative values indicate less agreement than expected by chance. The Kappa is beneficial in imbalanced settings as it penalizes uniform predictions, offering insights into shifts in class distributions within multi-class imbalanced data streams.

Additionally, we report the mG-Mean (multi-class G-Mean), which is an essential metric for evaluating classification model performance over imbalanced class distributions:

$$\text{mG-mean} = \sqrt[n]{\prod_{i=1}^{n} \text{sensitivity}_i} \tag{4.2}$$

$$\text{sensitivity}_i = \text{recall}_i = \frac{TP_i}{TP_i + FN_i} \tag{4.3}$$

Where sensitivity_i indicates the class-wise sensitivity. These measures give a complete picture of model performance and predictive abilities across different class distributions. All of these metrics are calculated across a 500-instance window.

4.1.5 Hyperparameter Selection

In our evaluation of the proposed method, we focus on four key parameters to assess how they impact overall performance using the Kappa metric. To investigate different values for each parameter, we conduct this analysis over four semi-synthetic datasets that have a sufficient amount of samples and a variety of features: 'ACTIVITY-D1', 'DJ30-D1', 'GAS-D1', and 'TAGS-D1'. The parameters we examine are:

- S_{slc}: The number of active components for classification, three values: 5, 10, and 15.
- n_{train}: The number of samples used for training, three values: 20, 50, and 100.
- n_{test}: The number of samples used for testing purposes, three values: 50, 100, and 200.
- n_v: The number of hyperplanes used in the undersampling process, five values: 2, 3, 4, 5, and 6.

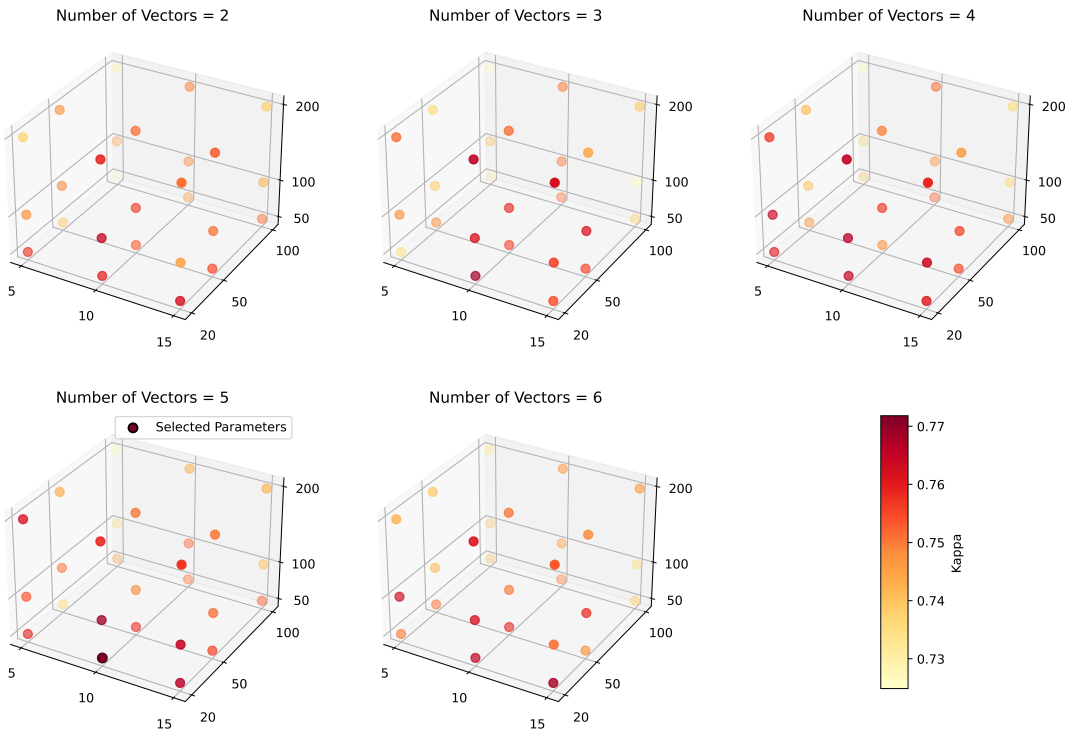


Figure 4.2: Hyperparameter Selection. (Related to Sec. 4.1.5).

x-axis [20-100]: Number of Train Samples (n_{train}), y-axis [5-15]: Number of Active Components (S_{slc}), and z-axis [50-200]: Number of Test Samples (n_{test}). Selected Default Parameters: $n_{train} = 20$, $n_v = 5$, $S_{slc} = 10$, and $n_{test} = 50$.

By adjusting these parameters, we aim to understand their influence on the method's performance and identify optimal configurations. These values produced 135 possible combinations of parameters for each dataset, totaling 540 results across four datasets.

The evaluation results are presented in Figure 4.2, and full results for each set of combinations are available on GitHub [4]. By closely analyzing the results, we determined that the optimal values for our key parameters are as follows: Our proposed method tends to yield a higher Kappa score with ten active components, 20 training samples, 50 test samples, and five hyperplanes, marked as 'Selected Parameters' in Figure 4.2 for the undersampling part with LSH-RHP. These values are also presented in Table 3.1.

[4]LSH-DynED

Table 4.2: Kappa and mG-Mean metrics comparison, and the Ranks of the methods for each dataset. (Related to Sec. 4.2).

Dataset		LSH-DynED	ARF	ARFR	BELS	CALMID	CSARF	GHVFDT	HDVFDT	KUE	LB	MicFoal	OBA	OOB	ROSE	SRP	UOB
Activity	*Kappa*	**77.49**	59.94	61.37	0.21	35.09	58.92	51.12	51.12	32.66	48.05	30.27	54.49	52.52	25.34	59.93	0.00
	mG − Mean	**88.64**	83.44	82.46	0.00	66.84	80.34	57.96	57.96	57.77	77.14	60.96	52.13	60.23	41.45	84.30	85.07
Connect-4	*Kappa*	26.19	32.23	35.46	23.39	34.79	36.98	26.08	21.10	19.69	33.68	30.53	25.84	37.90	**41.70**	33.44	20.02
	mG − Mean	51.05	61.47	53.55	22.84	52.45	47.02	41.69	49.43	50.41	55.72	51.44	56.32	47.44	48.90	**63.75**	36.22
Covtype	*Kappa*	**91.13**	51.98	52.28	78.69	77.96	47.69	22.83	12.03	34.99	44.10	61.44	42.56	37.89	40.57	52.65	14.02
	mG − Mean	**95.24**	85.82	84.59	82.04	82.94	62.22	57.99	70.94	75.39	79.85	85.18	76.53	66.75	80.19	85.81	4.78
Crimes	*Kappa*	**10.88**	0.36	0.36	3.54	1.29	0.14	1.14	1.82	0.99	2.74	0.02	2.27	4.35	1.52	4.40	0.00
	mG − Mean	34.64	80.74	80.00	0.00	34.78	80.57	51.20	43.40	43.41	47.21	83.26	53.03	27.94	41.24	16.63	**92.42**
Fars	*Kappa*	69.41	65.60	**70.39**	20.05	66.51	48.73	45.57	49.33	63.89	66.70	61.04	63.19	50.83	42.81	69.13	0.00
	mG − Mean	**84.01**	49.48	46.95	10.84	41.95	1.31	28.73	15.55	57.43	38.93	46.58	18.72	46.61	41.85	55.96	81.50
Gas	*Kappa*	**92.03**	69.70	62.52	84.34	55.25	72.34	40.55	40.55	30.42	70.68	79.39	60.21	39.12	43.97	82.98	0.00
	mG − Mean	**96.74**	36.65	27.48	87.24	16.35	36.55	9.71	9.71	22.72	27.10	37.87	29.95	8.41	19.93	39.11	51.83
Kr-vs-k	*Kappa*	5.64	10.00	9.97	**94.83**	10.08	10.19	8.13	8.13	8.73	10.21	9.85	10.23	8.28	9.74	10.13	8.77
	mG − Mean	13.22	10.48	10.48	**98.18**	10.48	10.48	10.48	10.48	10.48	10.48	10.48	10.48	10.48	10.48	10.48	10.48
Olympic	*Kappa*	9.89	32.27	35.57	7.35	30.31	26.16	18.24	7.11	26.05	34.93	30.37	24.17	27.03	**37.29**	7.07	10.74
	mG − Mean	22.68	**61.94**	45.89	3.70	44.89	27.80	31.56	35.33	40.52	53.39	44.78	44.42	26.06	42.47	58.32	13.16
Poker	*Kappa*	**95.07**	0.64	0.72	49.38	51.27	0.41	0.19	0.66	2.55	1.10	1.40	0.12	5.26	9.36	0.29	0.00
	mG − Mean	**97.36**	82.29	82.34	29.35	83.43	80.16	81.20	81.23	81.90	81.97	81.71	82.14	81.42	80.83	82.21	82.35
Shuttle	*Kappa*	**98.82**	2.72	12.88	55.31	72.78	-3.80	56.89	37.72	28.20	0.43	16.10	15.26	18.78	0.86	37.39	0.00
	mG − Mean	**99.59**	95.83	92.62	15.08	64.49	7.16	20.17	29.99	92.82	91.46	95.98	78.62	70.58	84.58	96.50	96.61
Tags	*Kappa*	47.99	1.94	1.90	**85.83**	51.70	1.81	22.09	23.86	23.99	18.09	35.04	16.32	29.70	10.62	7.79	0.00
	mG − Mean	73.11	55.71	56.43	**96.33**	11.65	54.91	2.52	2.22	2.08	41.33	36.84	43.54	2.10	45.01	56.93	62.61
Thyroid-l	*Kappa*	**87.73**	86.20	78.83	1.51	79.48	73.92	60.58	61.44	76.93	87.91	78.84	79.34	83.83	85.09	87.81	58.59
	mG − Mean	**95.53**	87.85	82.00	0.00	86.24	76.82	79.55	82.63	82.64	90.38	87.49	87.21	85.07	86.06	92.49	64.68
Zoo	*Kappa*	**93.63**	90.76	90.58	50.10	85.50	88.89	89.81	90.05	90.55	84.82	89.28	90.12	89.64	68.97	93.58	87.50
	mG − Mean	**97.19**	86.41	85.91	25.72	76.90	81.48	83.92	84.12	85.62	75.66	83.61	84.38	83.01	80.15	91.41	80.20
ACTIVITY-D1	*Kappa*	**75.80**	64.58	60.52	0.10	41.22	64.64	36.42	38.15	16.90	55.72	45.60	47.33	48.00	62.53	67.90	3.00
	mG − Mean	**87.95**	69.36	70.87	1.58	64.13	70.46	57.14	53.34	46.43	65.10	47.39	64.91	60.41	66.19	76.03	55.51
CONNECT4-D1	*Kappa*	24.01	32.12	30.75	23.88	33.14	37.68	22.59	22.05	15.72	33.64	29.85	25.83	31.56	**41.45**	33.61	16.14
	mG − Mean	52.62	**63.54**	61.62	27.27	50.51	51.83	35.48	43.54	47.74	54.24	50.55	57.82	42.48	49.51	61.70	33.93
COVERTYPE-D1	*Kappa*	**88.56**	57.12	56.23	78.03	75.23	45.28	37.07	36.09	51.84	54.52	63.23	45.25	59.33	57.82	60.49	8.43
	mG − Mean	**91.69**	84.30	83.73	84.36	85.24	62.49	53.36	47.77	77.92	82.03	80.86	76.76	69.60	80.47	83.80	8.03
CRIMES-D1	*Kappa*	6.91	2.21	2.48	2.84	1.85	11.58	1.41	0.27	4.22	0.59	5.32	0.01	4.61	10.87	**14.31**	3.81
	mG − Mean	32.01	65.55	51.63	11.19	33.82	28.53	48.79	48.49	41.58	71.28	29.10	**82.64**	25.86	30.64	39.98	22.29
DJ30-D1	*Kappa*	98.23	73.48	71.92	4.71	58.69	71.25	47.58	52.86	68.10	39.67	93.22	65.52	55.66	41.06	**98.68**	24.84
	mG − Mean	99.06	88.66	87.66	5.82	82.76	86.76	61.68	55.04	83.82	78.17	98.07	87.93	65.90	78.51	**99.03**	41.63
GAS-D1	*Kappa*	77.61	**87.07**	66.19	80.45	60.06	73.63	37.19	40.59	33.79	64.50	70.90	56.79	43.81	66.63	85.21	26.17
	mG − Mean	81.61	74.42	56.01	**88.07**	43.69	52.80	53.74	63.08	44.08	54.89	57.49	53.60	66.53	60.19	74.13	51.46
OLYMPIC-D1	*Kappa*	6.00	29.52	27.28	6.09	26.69	25.24	8.51	4.13	22.89	27.91	29.76	20.77	14.16	**33.75**	10.04	9.13
	mG − Mean	24.78	61.18	55.92	13.55	50.62	33.62	30.70	25.00	49.75	54.63	56.61	57.64	27.22	47.30	**62.07**	21.46
POKER-D1	*Kappa*	**57.15**	22.73	16.71	29.45	56.92	15.66	12.18	16.81	37.14	28.25	49.94	10.94	40.64	36.48	28.74	2.04
	mG − Mean	64.37	85.10	82.23	38.85	91.71	17.48	46.85	50.10	84.99	88.80	86.34	**93.79**	50.63	63.69	86.39	31.63
SENSOR-D1	*Kappa*	**92.35**	56.64	57.15	31.00	59.55	52.49	28.51	28.66	45.80	55.17	60.62	28.72	56.46	50.39	55.79	1.84
	mG − Mean	**94.92**	79.65	81.73	44.60	90.53	70.84	55.41	55.54	83.67	90.56	72.34	89.12	78.61	78.97	68.05	35.63
TAGS-D1	*Kappa*	47.21	42.28	42.65	**84.11**	42.99	39.86	11.74	12.84	9.87	39.34	57.14	28.48	12.01	33.88	46.36	2.37
	mG − Mean	70.50	56.28	55.13	**92.77**	40.36	48.42	7.27	7.96	20.65	60.39	49.71	62.68	23.03	47.02	59.54	27.21
Balance	*Kappa*	38.09	61.69	64.01	47.07	68.76	48.12	**69.57**	**69.57**	0.00	68.80	65.56	68.67	50.34	61.92	64.75	28.26
	mG − Mean	71.03	0.00	0.00	19.87	23.17	34.08	21.34	21.34	**79.31**	23.17	0.00	0.00	11.65	16.01	0.00	33.85
Contraceptive	*Kappa*	37.52	27.30	24.17	**93.38**	0.00	28.81	7.03	7.03	6.09	22.71	11.86	18.98	9.38	23.73	42.34	0.79
	mG − Mean	71.72	43.69	38.36	**96.74**	73.45	44.37	52.75	52.75	67.44	61.27	50.39	59.50	54.59	43.54	72.48	73.60
Ecoli	*Kappa*	46.38	45.66	46.08	**94.03**	44.49	54.93	39.37	39.37	0.00	44.52	50.29	42.05	43.56	40.66	45.21	0.00
	mG − Mean	86.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**89.87**	0.00	0.00	0.00	0.00	0.00	0.00	89.87
Glass	*Kappa*	34.05	25.46	21.18	**89.43**	21.87	26.46	21.55	21.55	0.00	21.87	35.66	21.70	25.00	30.56	29.45	0.00
	mG − Mean	64.52	0.00	0.00	**83.40**	0.00	47.39	0.00	0.00	85.25	0.00	61.26	0.00	0.00	47.78	58.48	85.25
Hayes-roth	*Kappa*	10.44	18.80	23.65	6.30	37.16	25.19	**40.59**	**40.59**	0.00	37.16	33.85	35.31	35.67	35.78	36.81	33.37
	mG − Mean	61.30	48.46	53.12	27.26	59.27	50.59	61.27	61.27	**72.83**	59.27	60.53	58.85	58.44	58.13	60.28	56.42
New-thyroid	*Kappa*	75.58	49.43	49.43	**85.97**	47.70	54.41	47.70	47.70	0.00	47.70	48.18	47.70	47.70	54.41	54.41	0.00
	mG − Mean	78.50	0.00	0.00	**95.97**	92.26	0.00	92.26	92.26	88.69	92.26	0.00	92.26	92.26	0.00	0.00	88.69
Pageblocks	*Kappa*	21.00	31.37	25.92	**48.48**	29.07	28.91	28.14	28.14	0.00	29.07	13.53	28.72	28.74	28.16	32.71	12.68
	mG − Mean	80.92	0.00	27.91	0.00	0.00	19.60	0.00	0.00	**98.00**	0.00	0.00	0.00	0.00	0.00	0.00	98.14
Thyroid-s	*Kappa*	**66.76**	41.07	29.62	-2.95	18.21	42.15	-0.26	0.49	0.00	16.39	64.35	-0.26	28.98	54.87	38.93	34.43
	mG − Mean	**91.82**	63.77	27.01	0.00	36.94	45.44	48.73	24.37	28.76	36.57	75.13	48.73	58.76	68.08	83.28	48.05
Wine	*Kappa*	46.37	35.94	39.46	**95.15**	42.96	41.62	42.96	42.96	0.00	42.96	38.57	39.40	37.61	42.08	38.01	-0.23
	mG − Mean	79.24	57.06	59.87	**96.38**	66.82	70.78	66.82	66.82	69.21	66.82	58.58	60.21	57.72	64.90	58.12	0.00
Yeast	*Kappa*	36.66	25.31	26.68	33.64	20.63	31.28	24.20	24.20	7.91	29.29	27.07	29.94	29.59	29.05	37.06	0.00
	mG − Mean	66.34	39.69	54.39	0.00	42.82	0.00	42.05	42.05	56.94	43.42	39.60	42.88	63.05	44.16	43.78	**82.63**
Avg. Mean	*Kappa*	**54.32**	40.43	39.24	45.03	43.61	38.84	30.52	29.67	23.03	38.28	42.97	34.73	36.00	37.56	44.47	12.32
	mG − Mean	**72.74**	56.33	53.87	39.36	51.56	44.92	42.19	41.93	61.21	56.17	53.94	54.75	46.15	49.95	58.21	52.95
Rank	*Kappa*	**4.70**	7.17	7.74	7.61	7.08	7.67	11.26	11.09	11.83	7.56	6.88	9.70	8.38	7.58	5.33	14.44
	mG − Mean	**3.88**	6.65	7.59	10.62	8.30	10.38	10.98	10.91	7.97	7.48	8.15	7.76	10.68	9.79	5.76	9.09

4.2 Effectiveness Analysis

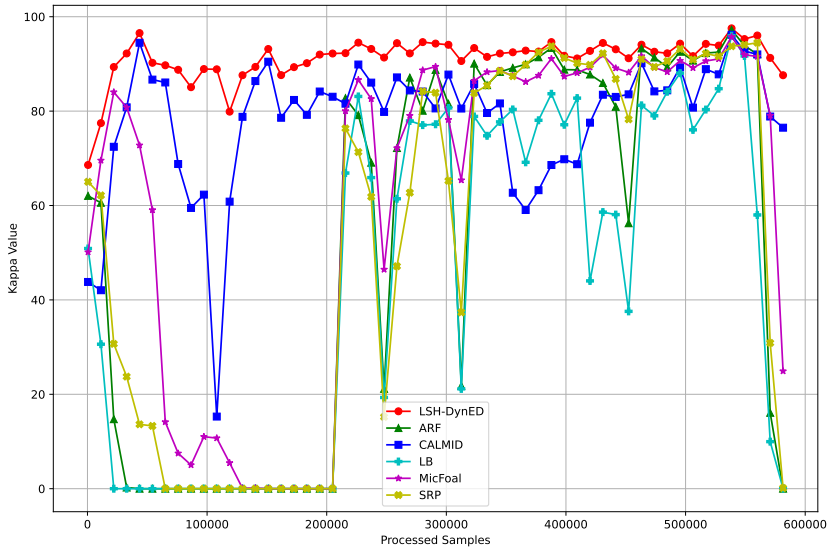
Table 4.2 and Figure 4.3 present the experimental evaluation results of various data stream classification models, focusing on their capability to effectively handle imbalanced multi-class non-stationary data streams. This included adapting to concept drift and maintaining accuracy across diverse class distributions. Compared to the other methods, the proposed approach has a better average performance in both the Kappa and mG-Mean metrics and achieves better rank.

In the sections that follow, we first compare the methods based on Kappa and mG-Mean scores in Section 4.2.1. Then, we statistically analyze the effectiveness of these methods in Section 4.2.2.

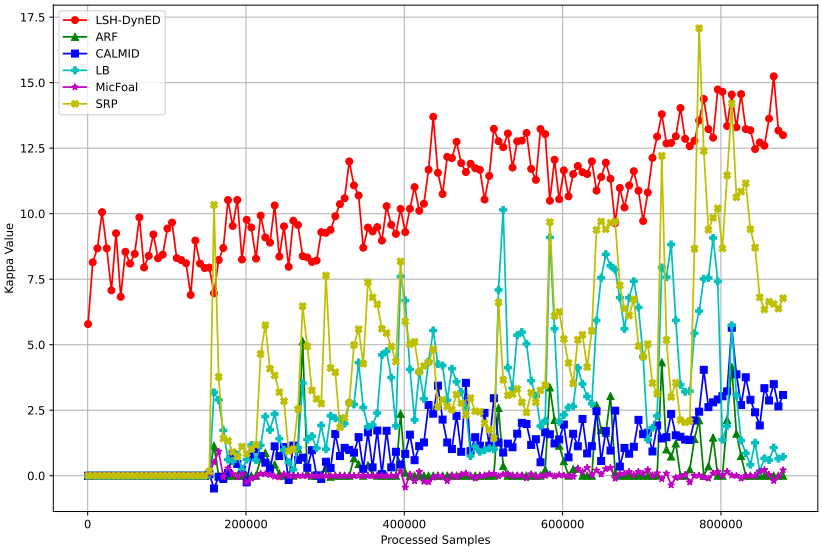
4.2.1 Effectiveness Comparison

General Purpose Methods (GPM): General Purpose Methods (GPM), presented in Table 2.1, are designed to handle balanced scenarios but are not explicitly tailored for imbalance. These models show varying degrees of success across different datasets. For example, in the 'Poker' dataset, OBA and LB struggle with high imbalance ratios, reflected in their lower Kappa (OBA: 0.12, LB: 1.10) and mG-Mean scores. This variability highlights GPM methods' challenges in highly imbalanced environments compared to specialized approaches. On the other hand, approaches such as BELS and SRP perform effectively on the Kappa metric and KUE on the mG-mean when compared to other GPM approaches and, in certain circumstances, outperform ISM methods.

Imbalance-Specific Methods (ISM): Imbalance-Specific Methods (ISM) are explicitly designed to address class imbalances. LSH-DynED, a proposed method within this category, demonstrates superior performance across multiple datasets. This is particularly noticeable in its ability to manage multi-class imbalances and adapt to concept drift, as seen in its high average Kappa (58.23) and mG-Mean (72.78) scores. Its robust performance is evident in challenging datasets

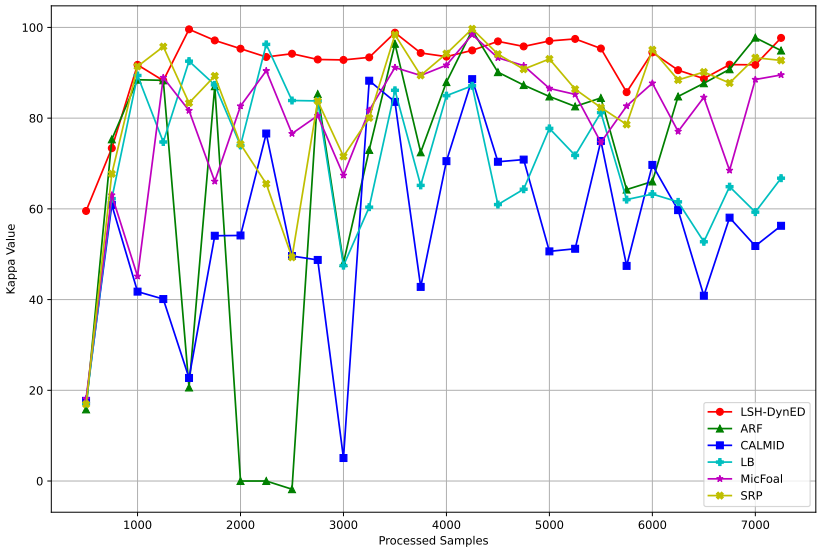


(a) Covtype.

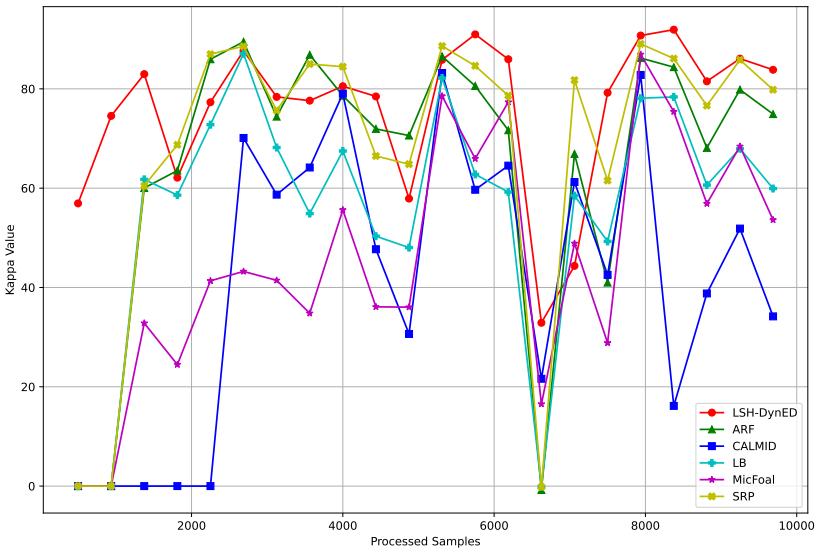


(b) Crimes.

Figure 4.3: Kappa metric results of three real and three semi-synthetic datasets. The results are plotted among the top 5 ranked methods in the Kappa metric: LSH-DynED, ARF, CALMID, LB, MicFoal, and SRP. (Related to Sec. 4.2).

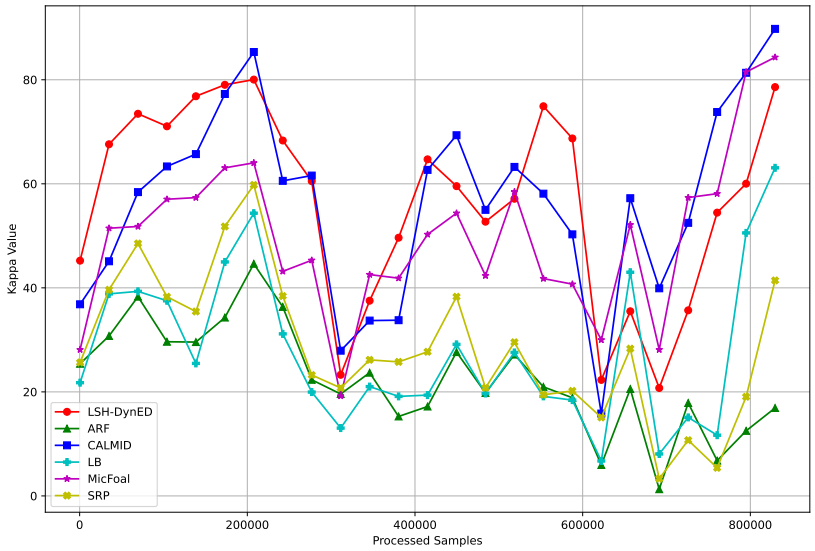


(c) Gas.

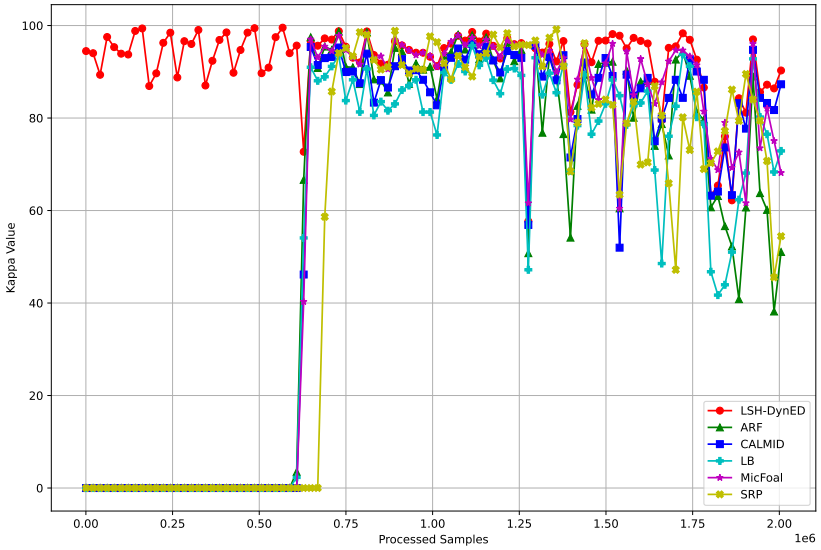


(d) ACTIVITY-D1.

Figure 4.3: Continuation of Kappa metric results.



(e) POKER-D1.



(f) SENSOR-D1.

Figure 4.3: Continuation of Kappa metric results.

like 'Poker' (Kappa: 95.11) and 'Shuttle' (Kappa: 99.32), and it shows adaptability in datasets with concept drift such as 'ACTIVITY-D1' (Kappa: 77.68), 'COVERTYPE-D1' (Kappa: 89.40), and 'SENSOR-D1' (Kappa: 92.26). Furthermore, it is evident how well LSH-DynED performs on datasets like "Activity," "Crimes," and "Fars" that have high imbalance ratios, indicating its resilience in such challenging environments. Thus, the integration of LSH-RHP in the proposed method helps it effectively undersample majority classes and maintain a balanced training set, enhancing its performance.

4.2.2 Statistical Analysis of Effectiveness Performances

We use the Friedman Test ($\alpha = 0.05$) to dismiss the null hypothesis, followed by implementing the Nemenyi posthoc test (critical distance (CD) = 2.30)[30]. The diagrams in Figure 4.4 indicate that the proposed method obtains the better ranking in all two measures of effectiveness.

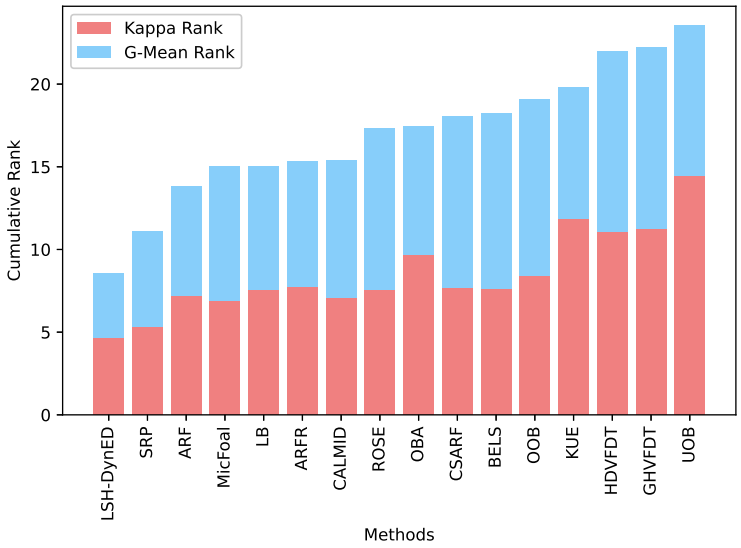
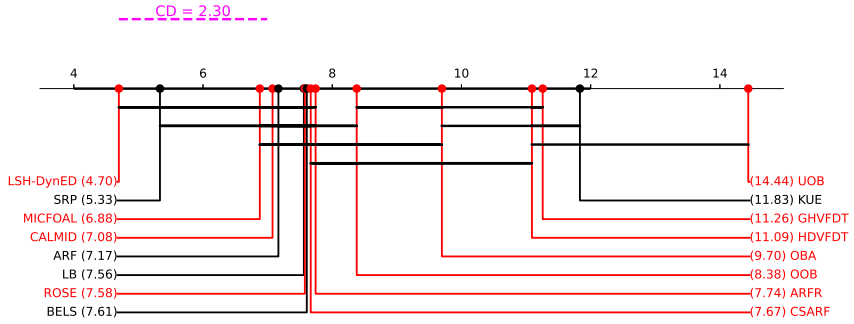
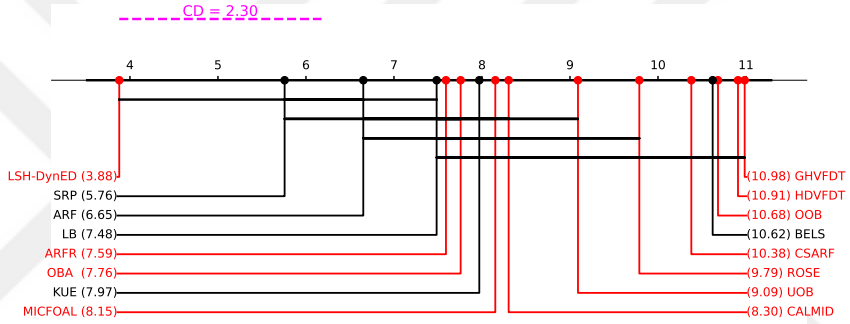


Figure 4.5: Cumulative ranking across two metrics (lower is better), models sorted in ascending order. (Related to Sec. 4.2.2).

Regarding the Kappa metric, LSH-DynED is in the same critical distance



(a) Kappa.



(b) mG-Mean.

Figure 4.4: Critical distance (CD) diagram based on the ranks of the methods (Table 4.2) for each evaluation metric (CD = 2.30). The color red is for ISM methods, and black is for GPM methods. (Related to Sec. 4.2.2).

as SRP and MicFoal and statistically significantly outperforms the rest of the methods. In the mG-Mean metric, LSH-DynED is in the same group as SRP, ARF, and LB and demonstrates statistically significant superiority over all other methods. Figure 4.5 shows the cumulative ranking of methods, with LSH-DynED having a better overall ranking.

4.3 Impact of Other Aspects on Effectiveness and Ablation Analysis

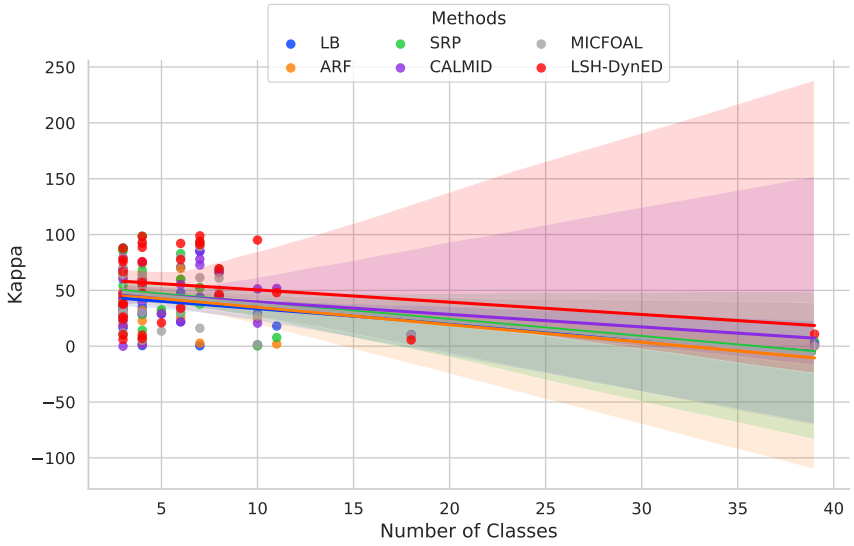
In this section, we analyze the effects of different factors. First, we look at how dataset characteristics affect the model's performance in Section 4.3.1. Next, we examine the impact of dynamic imbalance ratios in Section 4.3.2. Finally, we conduct an ablation analysis in Section 4.3.3.

4.3.1 Impact of Dataset Characteristics

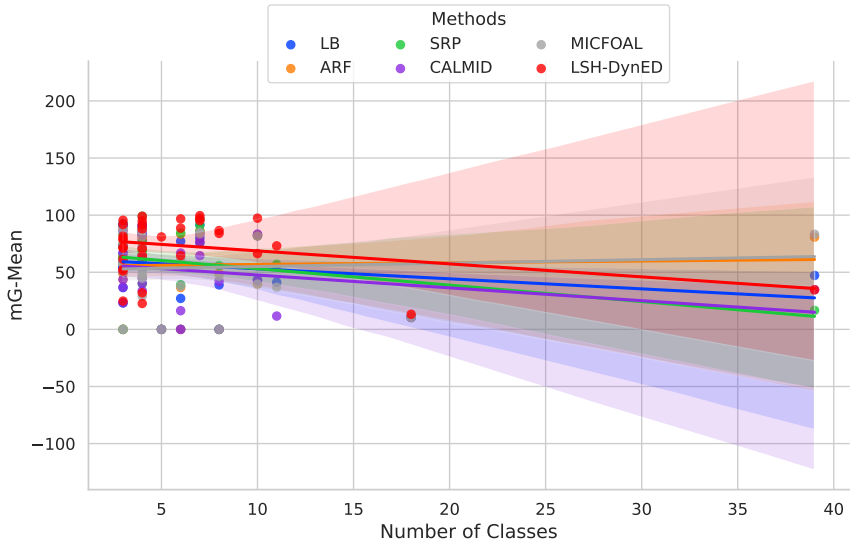
The characteristics of datasets in terms of the number of classes may significantly affect the performance of the models.

Table 4.3 and Figure 4.6 presents the correlation (Pearson correlation coefficient) [29] of each method's performance, Kappa and mG-Mean metrics against number classes on datasets available in Table 4.1. Based on our observations on these datasets, all methods show negative correlations between Kappa and the number of classes, indicating that the Kappa value tends to decrease as the number of classes increases. Several methods, such as ARF, ARFR, CSARF, LB, MicFoal, OOB, ROSE, and SRP, show a statistically significant negative correlation with the Kappa metric (p-value < 0.05). Meanwhile, ROSE with -0.51 has the strongest and most significant negative correlation with the Kappa metric. On the other hand, The correlations between mG-Mean and the number of classes vary, with both positive and negative values, and most correlations with mG-Mean are not statistically significant.

This analysis shows that the number of classes significantly impacts Kappa, with all methods showing a negative correlation. In contrast, the impact of the number of classes on mG-Mean is mixed and generally not significant, indicating varying degrees of correlation across different methods. In the case of LSH-DynED, a negative correlation between Kappa and the number of classes suggests that as the number of classes increases, Kappa tends to decrease. However, the



(a) Kappa.



(b) mG-Mean.

Figure 4.6: Effect of the number of classes over the performance of top 5 ranked methods in the Kappa metric: LSH-DynED, ARF, CALMID, LB, MicFoal, and SRP. (Related to Sec. 4.3.1).

p-value of 0.20 indicates that this correlation is not statistically significant.

Table 4.3: Correlation and p-value between the number of classes and both Kappa Metric and mG-Mean. $p < 0.05$ is statistically significant. (Related to Sec. 4.3.1).

Methods	**Kappa**		**mG-Mean**	
	Correlation	**p-value**	**Correlation**	**p-value**
LSH-DynED	-0.23	0.20	-0.31	0.08
ARF	-0.40	**0.02**	0.03	0.86
ARFR	-0.39	**0.02**	0.09	0.63
BELS	-0.02	0.91	-0.08	0.64
CALMID	-0.32	0.07	-0.26	0.14
CSARF	-0.41	**0.02**	0.08	0.64
GHVFDT	-0.28	0.12	-0.10	0.57
HDVFDT	-0.27	0.14	-0.14	0.43
KUE	-0.15	0.40	-0.25	0.17
LB	-0.38	**0.03**	-0.21	0.23
MICFOAL	-0.40	**0.02**	0.07	0.70
OBA	-0.31	0.08	-0.17	0.34
OOB	-0.35	**0.04**	-0.23	0.21
ROSE	-0.51	**0.00**	-0.15	0.41
SRP	-0.38	**0.03**	-0.32	0.07
UOB	-0.19	0.29	0.28	0.12

4.3.2 Impact of Dynamic Imbalance Ratios

Table 4.1 displays the characteristics of the datasets we evaluated, which include dynamic imbalance ratios. The last ten are explicitly made for this purpose [54], and the behavior of classes over time is presented in Figure 4.1. We independently compare the top 5 ranked methods to the proposed one to provide a clear image of their performance in the dynamic imbalance ratio challenge. Table 4.4 and Figure 4.3 (d), (e), and (f) illustrate the results. Interestingly, because of the complexity that this problem brings to the non-stationary data stream classification task, we see different performances across datasets for each method.

For instance, SRP outperforms the Kappa measure on certain datasets, including "CONNECT4-D1" and "CRIMES-D1", but only moderately on others. While MicFoal outperforms others in only two datasets in the Kappa measure,

Table 4.4: The Kappa and mG-Mean scores of the top 5 ranked methods in the Kappa metric for datasets with dynamic imbalance ratios. (Related to Sec. 4.3.2).

Dataset		LSH-DynED	ARF	CALMID	LB	MicFoal	SRP
ACTIVITY-D1	$Kappa$	**75.80**	64.58	41.22	55.72	45.60	67.90
	$mG-Mean$	**87.95**	69.36	64.13	65.10	47.39	76.03
CONNECT4-D1	$Kappa$	24.01	32.12	33.14	33.64	29.85	33.61
	$mG-Mean$	52.62	**63.54**	50.51	54.24	50.55	61.70
COVERTYPE-D1	$Kappa$	**88.56**	57.12	75.23	54.52	63.23	60.49
	$mG-Mean$	**91.69**	84.30	85.24	82.03	80.86	83.80
CRIMES-D1	$Kappa$	6.91	2.21	1.85	0.59	5.32	**14.31**
	$mG-Mean$	32.01	65.55	33.82	**71.28**	29.10	39.98
DJ30-D1	$Kappa$	98.23	73.48	58.69	39.67	93.22	**98.68**
	$mG-Mean$	**99.06**	88.66	82.76	78.17	98.07	99.03
GAS-D1	$Kappa$	77.61	**87.07**	60.06	64.50	70.90	85.21
	$mG-Mean$	**81.61**	74.42	43.69	54.89	57.49	74.13
OLYMPIC-D1	$Kappa$	6.00	29.52	26.69	27.91	29.76	**33.75**
	$mG-Mean$	24.78	61.18	50.62	54.63	56.61	**62.07**
POKER-D1	$Kappa$	**57.15**	22.73	56.92	28.25	49.94	28.74
	$mG-Mean$	64.37	85.10	**91.71**	88.80	86.34	86.39
SENSOR-D1	$Kappa$	**92.35**	56.64	59.55	55.17	60.62	55.79
	$mG-Mean$	**94.92**	79.65	90.53	90.56	72.34	68.05
TAGS-D1	$Kappa$	47.21	42.28	42.99	39.34	**57.14**	46.36
	$mG-Mean$	**70.50**	56.28	40.36	60.39	49.71	59.54
Avg. Mean	$Kappa$	**57.36**	46.77	45.63	39.93	50.56	50.11
	$mG-Mean$	69.95	**72.80**	63.33	70.01	62.84	71.07

"OLYMPIC-D1" and "TAGS-D1", it may obtain a slightly higher Kappa score than SRP. However, due to our undersampling technique, the LSH-DynED outperforms other methods in multiple datasets and achieves a higher average mean in terms of Kappa metric, demonstrating the LSH-DynED's resilience versus dynamic imbalance ratios problem.

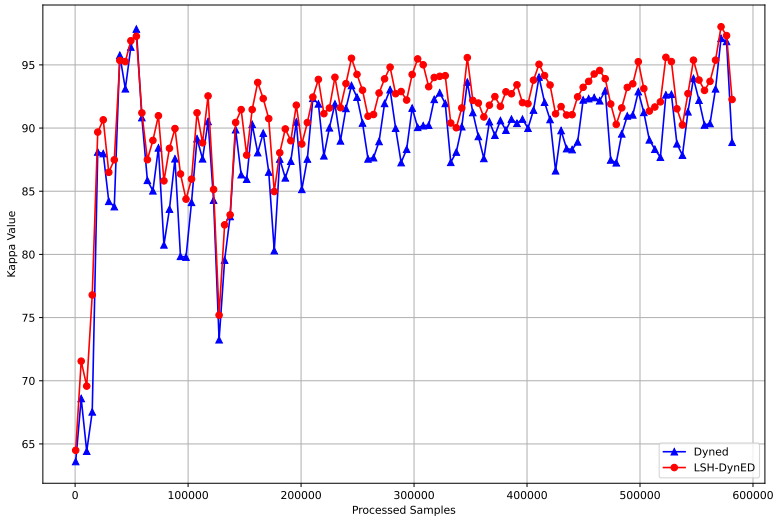
4.3.3 Ablation Analysis

In this section, we evaluated the efficacy of our proposed techniques in addressing the imbalance challenges inherent in multi-class non-stationary data streams. To this end, we selected 15 datasets from datasets presented in Table 4.1 with varying numbers of features, instance counts, and class sizes. We conducted a comparative analysis between the standard DynED model and our proposed LSH-DynED model, which incorporates multi-sliding windows and an LSh-RHP-based

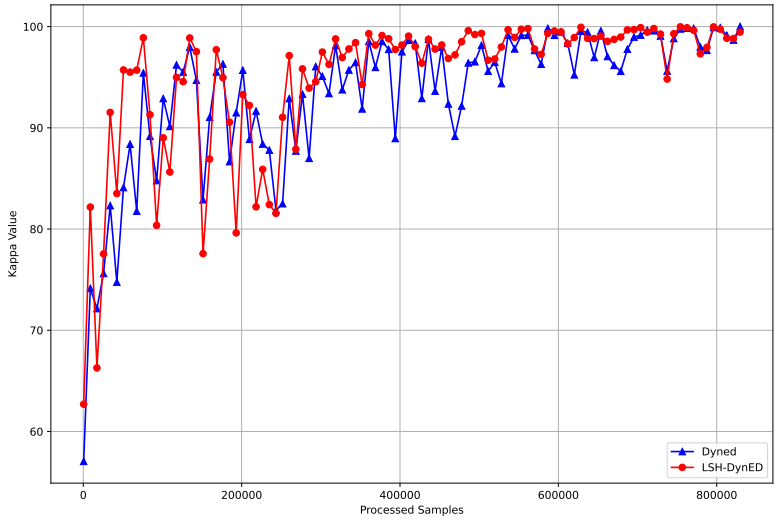
undersampling technique. The ablation analysis results, presented in Table 4.5 and Figure 4.7, support our assertion that the integrated techniques yield notable improvements in Kappa and mG-Mean metrics compared to the baseline DynED model.

Table 4.5: Ablation analysis of DynED vs LSH-DynED across various datasets. (Related to Sec. 4.3.3).

Dataset	**Kappa (%)**		**mG-Mean (%)**	
	LSH-DynED	**DynED**	**LSH-DynED**	**DynED**
Activity	**77.49**	74.35	**88.64**	47.13
Connect-4	**26.19**	22.50	**51.05**	17.04
Covtype	**91.13**	88.64	**95.24**	62.45
Gas	**92.03**	89.65	**96.74**	42.31
Poker	**95.07**	93.85	**97.36**	54.51
Shuttle	**98.82**	98.78	**99.59**	54.79
Thyroid	**87.73**	86.07	**95.53**	87.35
Zoo	**93.63**	93.24	**97.19**	89.03
ACTIVITY-D1	**75.80**	70.73	**87.95**	55.95
CONNECT4-D1	**24.10**	19.60	**52.62**	17.61
COVERTYPE-D1	**88.56**	85.98	**91.69**	69.00
DJ30-D1	98.23	**98.66**	**99.06**	98.49
TAGS-D1	**47.21**	41.09	**70.50**	18.47
Ecoli	46.38	**61.85**	**86.50**	0.00
Thyroid-s	**66.76**	54.76	**91.82**	31.08
Avg. Mean	**73.94**	71.98	**86.76**	49.68

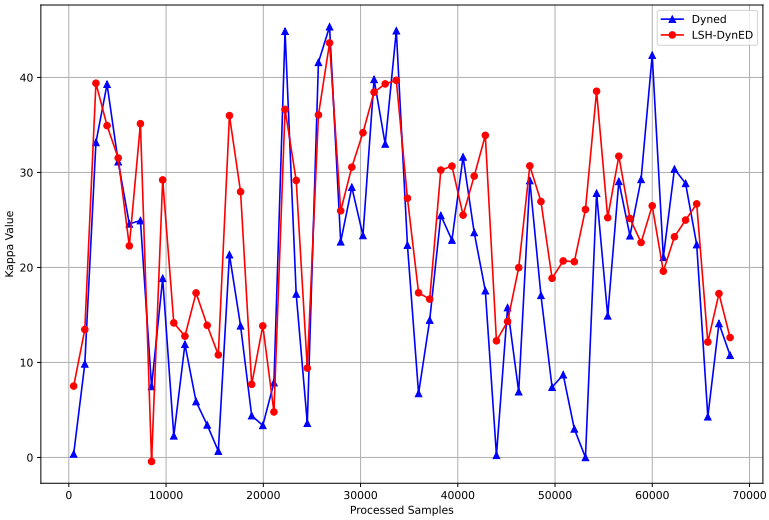


(a) Covtype.



(b) Poker.

Figure 4.7: Kappa Metric results of DynED vs. LSH-DynED for ablation analysis. (Related to Sec. 4.3.3).



(c) CONNECT4-D1.

Figure 4.7: Continuation of the Kappa Metric results.

4.4 Overall Evaluation of Experimental Observations

The comprehensive experimental evaluation of various data stream classification models, including our proposed LSH-DynED approach, reveals several key insights.

LSH-DynED demonstrates superior performance across multiple datasets, as evidenced by its higher Kappa and mG-Mean scores, particularly in challenging scenarios such as highly imbalanced datasets and those exhibiting concept drift. While all methods show a negative correlation between performance and the number of classes, LSH-DynED's correlation is not statistically significant, suggesting greater resilience to increasing the number of classes.

Our approach also outperforms other methods across datasets with dynamic

imbalance ratios, achieving higher scores in terms of both Kappa and mG-Mean measures. Statistical tests confirm LSH-DynED's superior ranking, with significant improvements over other methods. Ablation analysis highlights the effectiveness of the integrated multi-sliding windows and LSH-RHP-based undersampling technique.

Chapter 5

Conclusion & Future Directions

This study addressed the significant challenge of multi-class classification in imbalanced non-stationary data streams by proposing a novel approach that integrates Locality Sensitive Hashing with Random Hyperplane Projections (LSH-RHP) into the Dynamic Ensemble Diversification (DynED) approach.

Our method, LSH-DynED, introduces a robust and resilient undersampling technique to balance the training dataset and improve the accuracy of classification. Our comprehensive experiments on 23 real-world and ten semi-synthetic datasets demonstrate that LSH-DynED consistently outperforms the state-of-the-art methods in terms of both Kappa and mG-Mean effectiveness measures. The results highlight the method's capability in managing the classification of multi-class imbalanced non-stationary data streams and its robustness in large-scale, high-dimensional datasets. Maintaining high predictive accuracy and adaptability in ever-changing data streams positions LSH-DynED as a valuable tool for real-world applications.

Our proposed approach, LSH-DynED, represents a meaningful step forward in classifying multi-class imbalanced non-stationary data streams, offering a resilient and effective solution to a complex problem. Notably, this is the first time LSH-RHP has been employed for undersampling in the context of imbalanced

non-stationary data streams, underscoring the novelty and significance of our contribution.

Our future work will focus on three main areas. First, we will adapt our approach for binary imbalanced data streams. Second, we will study data streams with an extreme number of classes and those with emerging brand-new class labels. Third, we intend to introduce novel datasets representative of diverse streams across multiple disciplines. We believe LSH-DynED has the potential to make a significant impact in various domains, including fraud detection, network security, and healthcare monitoring. These steps will help grow this research area and show how our proposed approach can be useful in many fields.

Bibliography

[1] Abadifard, S., Bakhshi, S., Gheibuni, S., and Can, F. Dyned: Dynamic ensemble diversification in data stream classification. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (New York, NY, USA, 2023), CIKM '23, Association for Computing Machinery, p. 3707–3711.

[2] Abdulkareem Hamaamin, Rebin, Mohammed Amin Ali, Omar, and Wahhab Kareem, Shahab. Java programming language: Time permanence comparison with other languages: A review. *ITM Web Conf. 64* (2024), 01012.

[3] Aguiar, G., and Cano, A. An active learning budget-based oversampling approach for partially labeled multi-class imbalanced data streams. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing* (New York, NY, USA, 2023), SAC '23, Association for Computing Machinery, p. 382–389.

[4] Aguiar, G., Krawczyk, B., and Cano, A. A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework. *Machine Learning* (2023), 1–79.

[5] Aguiar, G. J., and Cano, A. A comprehensive analysis of concept drift locality in data streams. *Knowledge-Based Systems 289* (2024), 111535.

[6] Bahri, M., Bifet, A., Gama, J., Gomes, H. M., and Maniu, S. Data stream analysis: Foundations, major tasks and tools. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 11*, 3 (2021), e1405.

[7] Bakhshi, S., and Can, F. Balancing efficiency vs. effectiveness and providing missing label robustness in multi-label stream classification. *Knowledge-Based Systems 289* (2024), 111489.

[8] Bakhshi, S., Ghahramanian, P., Bonab, H., and Can, F. A broad ensemble learning system for drifting stream classification. *IEEE Access 11* (2023), 89315–89330.

[9] Bektas, E., and Can, F. Leveraging linear independence of component classifiers: Optimizing size and prediction accuracy for online ensembles. *arXiv preprint arXiv:2308.14175* (2023).

[10] Bernardo, A., and Della Valle, E. Vfc-smote: Very fast continuous synthetic minority oversampling for evolving data streams. *Data Mining and Knowledge Discovery 35*, 6 (2021), 2679–2713.

[11] Bernardo, A., della Valle, E., and Bifet, A. Incremental rebalancing learning on evolving data streams. In *2020 International Conference on Data Mining Workshops (ICDMW)* (2020), pp. 844–850.

[12] Bernardo, A., Gomes, H. M., Montiel, J., Pfahringer, B., Bifet, A., and Valle, E. D. C-smote: Continuous synthetic minority oversampling for evolving data streams. In *2020 IEEE International Conference on Big Data (Big Data)* (2020), pp. 483–492.

[13] Bernardo, A., and Valle, E. D. Smote-ob: Combining smote and online bagging for continuous rebalancing of evolving data streams. In *2021 IEEE International Conference on Big Data (Big Data)* (2021), pp. 5033–5042.

[14] Bifet, A., and Gavaldà, R. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)* (2007), pp. 443–448.

[15] Bifet, A., Holmes, G., and Pfahringer, B. Leveraging bagging for evolving data streams. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part I 21* (2010), Springer, pp. 135–150.

[16] Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., and Gavaldà, R. New ensemble methods for evolving data streams. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2009), KDD '09, Association for Computing Machinery, p. 139–148.

[17] Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., and Seidl, T. Moa: Massive online analysis, a framework for stream classification and clustering. In *Proceedings of the First Workshop on Applications of Pattern Analysis* (Cumberland Lodge, Windsor, UK, 01–03 Sep 2010), T. Diethe, N. Cristianini, and J. Shawe-Taylor, Eds., vol. 11 of *Proceedings of Machine Learning Research*, PMLR, pp. 44–50.

[18] Boiko Ferreira, L. E., Murilo Gomes, H., Bifet, A., and Oliveira, L. S. Adaptive random forests with resampling for imbalanced data streams. In *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), pp. 1–6.

[19] Bonab, H., and Can, F. Less is more: A comprehensive framework for the number of components of ensemble classifiers. *IEEE Transactions on Neural Networks and Learning Systems 30*, 9 (2019), 2735–2745.

[20] Breiman, L. *Classification and Regression Trees*. Routledge, 2017.

[21] Brzezinski, D., Stefanowski, J., Susmaga, R., and Szczech, I. On the dynamics of classification measures for imbalanced and streaming data. *IEEE Transactions on Neural Networks and Learning Systems 31*, 8 (2020), 2868–2878.

[22] Cano, A., and Krawczyk, B. Kappa updated ensemble for drifting data stream mining. *Machine Learning 109*, 1 (2020), 175–218.

[23] Cano, A., and Krawczyk, B. Rose: robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. *Machine Learning 111*, 7 (2022), 2561–2599.

[24] Carraher, L. A., Wilsey, P. A., Moitra, A., and Dey, S. Random projection clustering on streaming data. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)* (2016), pp. 708–715.

[25] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16* (2002), 321–357.

[26] Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003. Proceedings 7* (2003), Springer, pp. 107–119.

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

[28] Cieslak, D. A., and Chawla, N. V. Learning decision trees for unbalanced data. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part I 19* (2008), Springer, pp. 241–256.

[29] Cohen, I., Huang, Y., Chen, J., Benesty, J., Benesty, J., Chen, J., Huang, Y., and Cohen, I. Pearson correlation coefficient. *Noise Reduction in Speech Processing* (2009), 1–4.

[30] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research 7* (2006), 1–30.

[31] Derrac, J., Garcia, S., Sanchez, L., and Herrera, F. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult. Valued Logic Soft Comput 17* (2015), 255–287.

[32] Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine 10*, 4 (2015), 12–25.

[33] Douzas, G., and Bacao, F. Geometric smote: Effective oversampling for imbalanced learning through a geometric extension of smote, 2017.

[34] Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., and Jégou, H. The faiss library. *arXiv preprint arXiv:2401.08281* (2024).

[35] Drummond, C., Holte, R. C., et al. C4. 5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets II* (2003).

[36] Espíndola, R. P., and Ebecken, N. F. On extending f-measure and g-mean metrics to multi-class problems. *WIT Transactions on Information and Communication Technologies 35* (2005), 25–34.

[37] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., and Herrera, F. *Learning from Imbalanced Data Sets*, vol. 10. Springer, 2018.

[38] Gama, J. a., Sebastião, R., and Rodrigues, P. P. Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2009), KDD '09, Association for Computing Machinery, p. 329–338.

[39] Gomes, H. M., Barddal, J. P., Enembreck, F., and Bifet, A. A survey on ensemble learning for data stream classification. *ACM Comput. Surv. 50*, 2 (March 2017).

[40] Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., and Abdessalem, T. Adaptive random forests for evolving data stream classification. *Machine Learning 106* (2017), 1469–1495.

[41] Gomes, H. M., Read, J., and Bifet, A. Streaming random patches for evolving data stream classification. In *2019 IEEE International Conference on Data Mining (ICDM)* (2019), pp. 240–249.

[42] Gözüaçık, Ö., and Can, F. Concept learning using one-class classifiers for implicit drift detection in evolving data streams. *Artificial Intelligence Review 54*, 5 (2021), 3725–3747.

[43] Gulcan, E. B., and Can, F. Unsupervised concept drift detection for multi-label data streams. *Artificial Intelligence Review 56*, 3 (2023), 2401–2434.

[44] Gulowaty, B., and Ksieniewicz, P. Smote algorithm variations in balancing data streams. In *Intelligent Data Engineering and Automated Learning – IDEAL 2019: 20th International Conference, Manchester, UK, November 14–16, 2019, Proceedings, Part II 20* (2019), Springer, pp. 305–312.

[45] Han, H., Wang, W.-Y., and Mao, B.-H. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing* (2005), Springer, pp. 878–887.

[46] Japkowicz, N. Assessment metrics for imbalanced learning. *Imbalanced Learning: Foundations, Algorithms, and Applications* (2013), 187–206.

[47] Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data 7*, 3 (2021), 535–547.

[48] Kailath, T. The divergence and bhattacharyya distance measures in signal selection. *IEEE Transactions on Communication Technology 15*, 1 (1967), 52–60.

[49] Kelly, M., Longjohn, R., and Nottingham, K. The uci machine learning repository. *URL https://archive. ics. uci. edu* (2023).

[50] Khoirom, S., Sonia, M., Laikhuram, B., Laishram, J., and Singh, T. D. Comparative analysis of python and java for beginners. *Int. Res. J. Eng. Technol 7*, 8 (2020), 4384–4407.

[51] Ko, M., Yang, X., Ji, Z., Just, H. A., Gao, P., Kumar, A., and Jia, R. Privmon: A stream-based system for real-time privacy attack detection for machine learning models. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses* (New York, NY, USA, 2023), RAID '23, Association for Computing Machinery, p. 264–281.

[52] Kokate, U., Deshpande, A., Mahalle, P., and Patil, P. Data stream clustering techniques, applications, and models: comparative analysis and discussion. *Big Data and Cognitive Computing 2*, 4 (2018), 32.

[53] Korycki, Ł., Cano, A., and Krawczyk, B. Active learning with abstaining classifiers for imbalanced drifting data streams. In *2019 IEEE International Conference on Big Data (Big Data)* (2019), pp. 2334–2343.

[54] Korycki, Ł., and Krawczyk, B. Online oversampling for sparsely labeled imbalanced and non-stationary data streams. In *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), pp. 1–8.

[55] Kubat, M., Matwin, S., et al. Addressing the curse of imbalanced training sets: One-sided selection. In *ICML* (1997), vol. 97, Citeseer, p. 179.

[56] Kuncheva, L., and Whitaker, C. Ten measures of diversity in classifier ensembles: limits for two classifiers. In *A DERA/IEE Workshop on Intelligent Sensor Processing (Ref. No. 2001/050)* (2001), pp. 10/1–1010.

[57] Kuncheva, L. I., and Whitaker, C. J. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning 51*, 2 (2003), 181.

[58] Laney, D., et al. 3d data management: Controlling data volume, velocity, and variety. *META Group Research Note 6*, 70 (2001), 1.

[59] Laurikkala, J. Improving identification of difficult small classes by balancing class distribution. In *Artificial Intelligence in Medicine: 8th Conference on Artificial Intelligence in Medicine in Europe, AIME 2001 Cascais, Portugal, July 1–4, 2001, Proceedings 8* (2001), Springer, pp. 63–66.

[60] Lee, K. M. Locality sensitive hashing with extended partitioning boundaries. *Applied Mechanics and Materials 321* (2013), 804–807.

[61] Li, Z., Huang, W., Xiong, Y., Ren, S., and Zhu, T. Incremental learning imbalanced data streams with concept drift: The dynamic updated ensemble algorithm. *Knowledge-Based Systems 195* (2020), 105694.

[62] Lipska, A., and Stefanowski, J. The influence of multiple classes on learning from imbalanced data streams. In *Fourth International Workshop on Learning with Imbalanced Domains: Theory and Applications* (2022), PMLR, pp. 187–198.

[63] Liu, W., Zhang, H., Ding, Z., Liu, Q., and Zhu, C. A comprehensive active learning method for multiclass imbalanced data streams with concept drift. *Knowledge-Based Systems 215* (2021), 106778.

[64] Liu, W., Zhu, C., Ding, Z., Zhang, H., and Liu, Q. Multiclass imbalanced and concept drift network traffic classification framework based on online active learning. *Engineering Applications of Artificial Intelligence 117* (2023), 105607.

[65] Liu, X., Fan, X., Deng, C., Li, Z., Su, H., and Tao, D. Multilinear hyperplane hashing. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 5119–5127.

[66] Loezer, L., Enembreck, F., Barddal, J. P., and de Souza Britto, A. Cost-sensitive learning for imbalanced data streams. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing* (New York, NY, USA, 2020), SAC '20, Association for Computing Machinery, p. 498–504.

[67] Lu, Y., Cheung, Y.-M., and Yan Tang, Y. Adaptive chunk-based dynamic weighted majority for imbalanced data streams with concept drift. *IEEE Transactions on Neural Networks and Learning Systems 31*, 8 (2020), 2764–2778.

[68] Lyon, R., Brooke, J., Knowles, J., and Stappers, B. Hellinger distance trees for imbalanced streams. In *2014 22nd International Conference on Pattern Recognition* (2014), pp. 1969–1974.

[69] Montiel, J., Halford, M., Mastelini, S. M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H. M., Read, J., Abdessalem, T., and Bifet, A. River: Machine learning for streaming data in python. *Journal of Machine Learning Research 22*, 110 (2021), 1–8.

[70] Nguyen, H. M., Cooper, E. W., and Kamei, K. A comparative study on sampling techniques for handling class imbalance in streaming data. In *The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems* (2012), pp. 1762–1767.

[71] Olaitan, O. M., and Viktor, H. L. Scut-ds: Learning from multi-class imbalanced canadian weather data. In *Foundations of Intelligent Systems: 24th International Symposium, ISMIS 2018, Limassol, Cyprus, October 29–31, 2018, Proceedings 24* (2018), Springer, pp. 291–301.

[72] Oza, N. C., and Russell, S. J. Online bagging and boosting. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics* (04–07 Jan 2001), T. S. Richardson and T. S. Jaakkola, Eds., vol. R3 of *Proceedings of Machine Learning Research*, PMLR, pp. 229–236. Reissued by PMLR on 31 March 2021.

[73] Quinlan, J. R. Induction of decision trees. *Machine Learning 1* (1986), 81–106.

[74] Rao, C. R. A review of canonical coordinates and an alternative to correspondence analysis using hellinger distance. *Qüestiió: Quaderns d'estadística i investigació operativa* (1995).

[75] Read, J., Bifet, A., Holmes, G., and Pfahringer, B. Streaming multi-label classification. In *Proceedings of the Second Workshop on Applications of Pattern Analysis* (CIEM, Castro Urdiales, Spain, 19–21 Oct 2011), T. Diethe, J. Balcazar, J. Shawe-Taylor, and C. Tirnauca, Eds., vol. 17 of *Proceedings of Machine Learning Research*, PMLR, pp. 19–25.

[76] Read, J., and Zliobaite, I. Learning from data streams: An overview and update. *Available at SSRN 4326595* (2023).

[77] Santhiappan, S., Chelladurai, J., and Ravindran, B. A novel topic modeling based weighting framework for class imbalance learning. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data* (New York, NY, USA, 2018), CODS-COMAD '18, Association for Computing Machinery, p. 20–29.

[78] Sobhani, P., Viktor, H., and Matwin, S. Learning from imbalanced data using ensemble methods and cluster-based undersampling. In *New Frontiers in Mining Complex Patterns: Third International Workshop, NFMCP 2014, Held in Conjunction with ECML-PKDD 2014, Nancy, France, September 19, 2014, Revised Selected Papers 3* (2015), Springer, pp. 69–83.

[79] Song, Y., Lu, J., Lu, H., and Zhang, G. Learning data streams with changing distributions and temporal dependency. *IEEE Transactions on Neural Networks and Learning Systems 34*, 8 (2023), 3952–3965.

[80] Tarawneh, A. S., Hassanat, A. B., Altarawneh, G. A., and Almuhaimeed, A. Stop oversampling for class imbalance learning: A review. *IEEE Access 10* (2022), 47643–47660.

[81] Tomek, I. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics SMC-6*, 6 (1976), 448–452.

[82] Tomek, I. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics SMC-6*, 11 (1976), 769–772.

[83] Tsymbal, A., Pechenizkiy, M., and Cunningham, P. Diversity in search strategies for ensemble feature selection. *Information Fusion 6*, 1 (2005), 83–98. Diversity in Multiple Classifier Systems.

[84] Tsymbal, A., Pechenizkiy, M., and Cunningham, P. Diversity in search strategies for ensemble feature selection. *Information Fusion 6*, 1 (2005), 83–98. Diversity in Multiple Classifier Systems.

[85] Vardi, M. Y. Efficiency vs. resilience: What COVID-19 teaches computing, April 2020.

[86] Wang, C.-h., and Pham, L. *Resilience and Robustness*. Engineers Australia, Barton, A.C.T., 2012, pp. 114–121.

[87] Wang, S., and Minku, L. L. Auc estimation and concept drift detection for imbalanced data streams with multiple classes. In *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), pp. 1–8.

[88] Wang, S., Minku, L. L., and Yao, X. Dealing with multiple classes in online class imbalance learning. In *IJCAI* (2016), pp. 2118–2124.

[89] Wang, S., Minku, L. L., and Yao, X. A systematic study of online class imbalance learning with concept drift. *IEEE Transactions on Neural Networks and Learning Systems 29*, 10 (2018), 4802–4821.

[90] Wares, S., Isaacs, J., and Elyan, E. Data stream mining: methods and challenges for handling concept drift. *SN Applied Sciences 1* (2019), 1–19.

[91] Yen, S.-J., and Lee, Y.-S. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications 36*, 3, Part 1 (2009), 5718–5727.

[92] Zhu, T., Lin, Y., and Liu, Y. Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition 72* (2017), 327–340.

Appendix A

Data and Code Availability

The datasets used in the experiments and the source code of the proposed model are publicly available and listed below as hyperlinks:

- LSH-DynED model
- Imbalanced-Streams
- BELS