MENOUFIA UNIVERSITY

LEARNING IN NON-STATIONARY ENVIORONMENT

by

Ahmed Hamdy Madkour

A dissertation submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree, Faculty of Computers and Information, Menoufia University

Supervised by Dr. Amgad Monir and Prof. Hatem Mohamed

MENOUFIA UNIVERSITY

LEARNING IN NON-STATIONARY ENVIORONMENT

by

Ahmed Hamdy Madkour

A dissertation submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree, Faculty of Computers and Information, Menoufia University

Supervised by Dr. Amgad Monir and Prof. Hatem Mohamed

The candidate The supervisor

Menoufia, August 2024

Learning in Non-stationary Envioronment

Author: Ahmed Hamdy Madkour

Supervisor: Dr. Amgad Monir and Prof. Hatem Mohamed

Text printed in Bilbao

First edition, August 2024

Dedicated to everyone who wish for a better life!

Abstract

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

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

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

Summary

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

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

Acknowledgements

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

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

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

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

Thank you everyone,

Ahmed Madkour

August 2024

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

Marie Curie

CHAPTER

1

State-of-the-art

In this chapter, we provide a comprehensive review of recent advancements in stream classification, addressing several critical challenges that arise in dynamic data environments. The rapid evolution of data streams introduces complexities such as concept drift, imbalanced multiclass scenarios, class overlap, classifier ensemble selection, emergence of new classes, and the incorporation of transfer learning. This introduction outlines the structure of the chapter and highlights the significance of each topic in advancing stream classification methodologies. The first section (Setion 1.1) focuses on concept drift, a phenomenon where the statistical properties of the data change over time. We explore various methodologies developed for real-time detection and adaptation to concept drift in streaming scenarios. This includes a review of techniques that enable systems to identify shifts in data patterns promptly, ensuring sustained classification accuracy. By analyzing the strengths and limitations of these approaches, we highlight the necessity for robust drift detection mechanisms in evolving data streams. Next, in Section 1.2, we delve into classifier ensemble selection in the context of streaming data. As the characteristics of data streams evolve, selecting the most appropriate ensemble of classifiers becomes crucial. We present algorithms designed to dynamically choose the best-performing classifiers based on the current data distribution. This section

emphasizes the importance of adaptive ensemble strategies in enhancing classification performance amidst changing data landscapes. Section 1.3 addresses the challenges posed by imbalanced multiclass scenarios, particularly in the presence of overlapping classes. We discuss specialized oversampling techniques aimed at countering the effects of class imbalance in streaming data. By providing insights into effective strategies for balancing the representation of minority classes, we lay the groundwork for developing robust frameworks that can maintain performance in imbalanced drifted streams. As new classes emerge in streaming systems, traditional classifiers often struggle to adapt effectively. In Section 1.4, we explore methodologies that specifically focus on the integration of new classes within existing frameworks. This section highlights innovative approaches that enhance the adaptability of stream classification systems, ensuring they remain effective as new data patterns emerge. In Section 1.5, we examine the role of transfer learning in stream classification. This section discusses how transfer learning techniques can leverage knowledge from related tasks to improve classification performance in dynamic environments. We explore current methodologies that facilitate the transfer of information across different domains, particularly in situations where labeled data may be scarce. The insights gained from this discussion will be pivotal for understanding how transfer learning can complement other strategies in our proposed framework. In Section 1.6,To further contextualize our discussion, we conduct a comparative analysis of recent works in the field, focusing on the recent works for each challange. We assess their contributions and identify limitations, illuminating specific gaps in the current research landscape that our work aims to address. Finally, Section 1.7 concludes this chapter by identifying key remarks and gaps, which inform the research direction for the subsequent chapters.

1.1 Concept Drift

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

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

1.2 Classifier Ensemble Selection

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

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

1.3 Imbalanced data Streams

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

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

1.4 Streams with Emerging New Classes (SENC)

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

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

1.5 Transfer Learning

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

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

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

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

1.6 Comparsion

In this section, we undertake a critical comparison of closely related works addressing the challenges of imbalanced multiclass streams 1.6.1, the emergence of new classes 1.6.2, and the integration of transfer learning 1.6.3 within streaming environments. The increasing complexity of real-world data streams necessitates advanced methodologies that can effectively manage the intricacies of these challenges. By examining various approaches in the literature, we aim to highlight their contributions, strengths, and limitations in dealing with imbalanced data distributions, adapting to new class occurrences, and leveraging transfer learning techniques. This comparative analysis not only sheds light on the current state of research but also underscores the specific gaps and unresolved issues that our work seeks to address, ultimately paving the way for more robust and adaptive solutions in the realm of streaming data classification.

1.6.1 Imbalanced Stream

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

generating synthetic samples exclusively from minority class instances within a restricted neighborhood, MLSOL produces a more compact and efficient synthetic dataset, mitigating concerns related to overfitting. As illustrated in Fig. 1.1, MLSOL is more likely to select x1 as a seed instance because it is surrounded by more neighbors of the opposite class for l3. MLSMOTE assigns the label vector [0,1,0] to all synthetic instances based on their neighbors. In contrast, MLSOL creates more diverse instances by assigning labels according to their location. Moreover, synthetic instances c2 and c3 generated by MLSMOTE introduce noise, whereas MLSOL copies the labels of the nearest instance to the new examples. In summary, MLSMOTE tends to generate new instances biased toward the dominant class in the local area, whereas MLSOL effectively explores and exploits both the feature and label space.

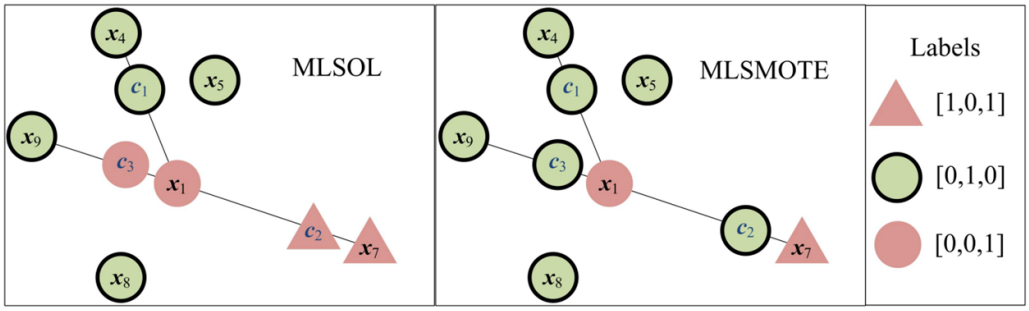

Figure 1.1: Distribution of the prediction accuracy.

Table 1.1 presents a comparison of two methods, MLSMOTE and MLSOL, which are designed to address the issue of imbalanced data in multi-class classification. MLSMOTE enhances classifier performance by generating synthetic examples for each minority class label, thereby balancing the class distribution. Its primary advantage is the generation of these synthetic examples, which helps mitigate the imbalance. However, it has significant limitations: the synthetic samples generated might be related to the majority class, which can blur the distinction between classes, and the method struggles with overlapping classes, leading to potential misclassification. On the other hand, MLSOL systematically combats local imbalances by employing distinct sampling strategies for each label within a restricted

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

Table 1.1: Comparison of MLSMOTE and MLSOL Methods

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

1.6.2 Emergence of new classes

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

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

Fig. 1.2 illustrates the SENCForest approach, which divides the space into three regions (normal, outlying, and anomaly) and detects emerging new classes (anomalies) using a calculated threshold path length. Fig. 1.3 depicts the SENNE algorithm, where hyperplanes are drawn in three dimensions (x1, x2, and x3) for each class (Fig. 1.3a). New instances are then classified as emerging or known classes based on the rank of each class (Fig. 1.3b). Fig. 1.4 presents the KNNENS algorithm, which draws hyperplanes for all class samples (Fig. 1.4a), and classifies new instances using a voting mechanism to determine if the instance is an emerging or known class (Fig. 1.4b). These visualizations highlight the operational differences between the SENNE and KNNENS algorithms in handling the classification of emerging and known classes.

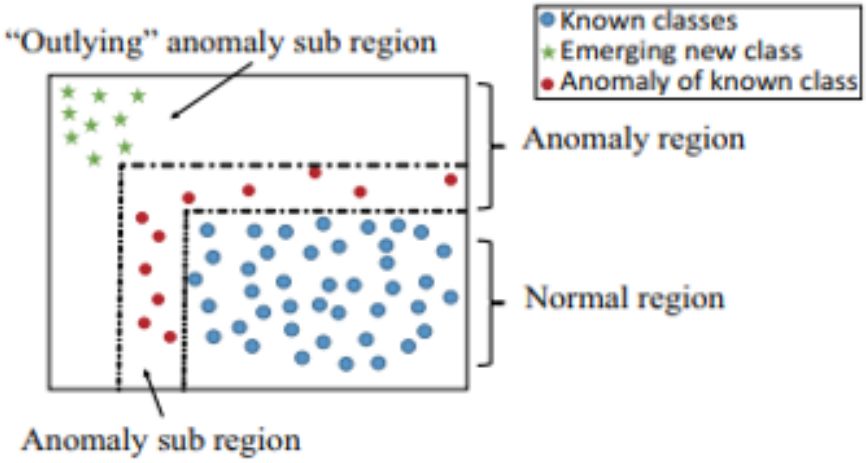

Figure 1.2: Stream Emerging New Class Overview

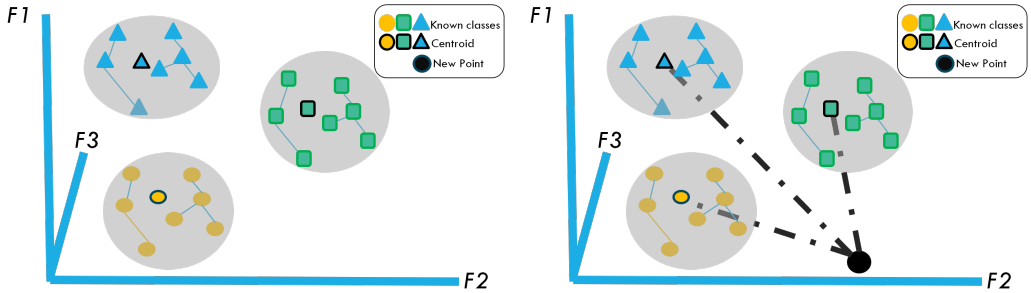

Figure 1.3: Stream Emerging Nearest Neighbor Ensemble (SENNE) Overview.

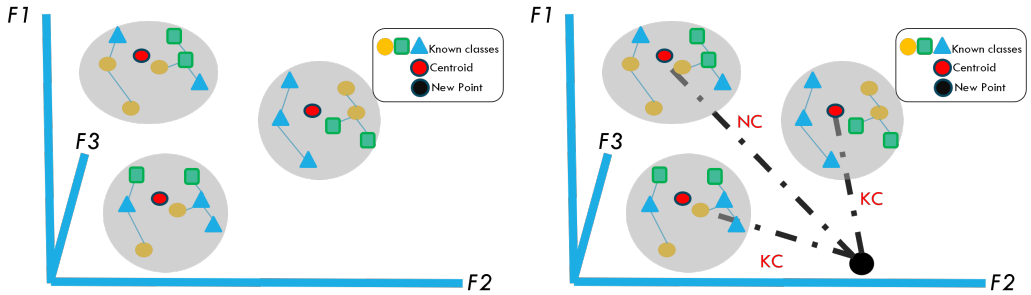

Figure 1.4: k-nearest Neighbor Ensemble-based (KENNE) Overview.

Table1.2 compares three methods for emerging class detection: SENCForest, SENNE, and KNNENS. SENCForest employs the anomaly detection method iForest for new class detection and uses a threshold path to identify anomalies, serving both as an unsupervised anomaly detector and a supervised classifier. However, it has a high potential for false positives and depends on a complex path length threshold. SENNE utilizes a nearest neighbor-based hypersphere of one class ensemble to explore local neighborhood information and sort distances, handling both low and high geometric distances between classes. Its limitations include the assumption that the distribution of known classes remains unchanged and it has lengthy update times. KNNENS employs a nearest neighbor-based hypersphere of all class

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

Table 1.2: Comparison of SENCForst, SENNE and KENNE Methods

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

1.6.3 Transfer Learning

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

tion matrix, CORAL projects source instances into the target domain, thereby minimizing domain discrepancies and reducing negative knowledge transfer. Melanie addresses the challenge of non-stationary environments through an online ensemble learning approach. It incrementally trains models from both source and target domains, dynamically adjusting their weights to handle concept drift, and combines these models via a weighted-sum approach as shown in Fig. 1.5. As Shown in Fig. 1.6 This method can be extended to Concept Drift Transfer Learning (CDTL) by using an ensemble for chunk-based concept drift. HE-CDTL, designed explicitly for CDTL, leverages knowledge from source domains and historical time steps within the target domain to enhance learning performance. It utilizes a class-wise weighted ensemble for historical knowledge and implements AW-CORAL for extracting knowledge from source domains. The class-wise weighted ensemble allows individual classes to select historical knowledge independently, while AW-CORAL minimizes domain disparities and mitigates negative knowledge transfer. Extensive experiments have shown HE-CDTL to outperform baseline methods in addressing transfer learning challenges in the context of concept drift. Together, these methods provide a comprehensive framework for effective transfer learning in dynamic and evolving data environments.

Table1.3 compares three methods: CORAL, Melanie, and HE-CDTL. CORAL (Correlation Alignment) utilizes a learned transformation matrix and Singular Value Decomposition (SVD) to project source instances into the target domain, effectively minimizing domain discrepancy and reducing negative knowledge transfer. However, it faces challenges with non-stationary and heterogeneous data. Melanie (Multi-source Online Transfer learning for Non-stationary Environments) addresses online learning problems where data in source and target domains are generated from non-stationary environments. Its advantages include considering online problems but it too is limited by the complexities of online learning and data heterogeneity. HE-CDTL (Class-wise Weighted and Domain-wise Ensemble) minimizes domain shift by aligning second-order statistics of source and target distributions, leveraging historical knowledge to reduce disparities between domains. Despite its strengths, it relies on the quality of the source domain and also struggles with heterogeneous data.

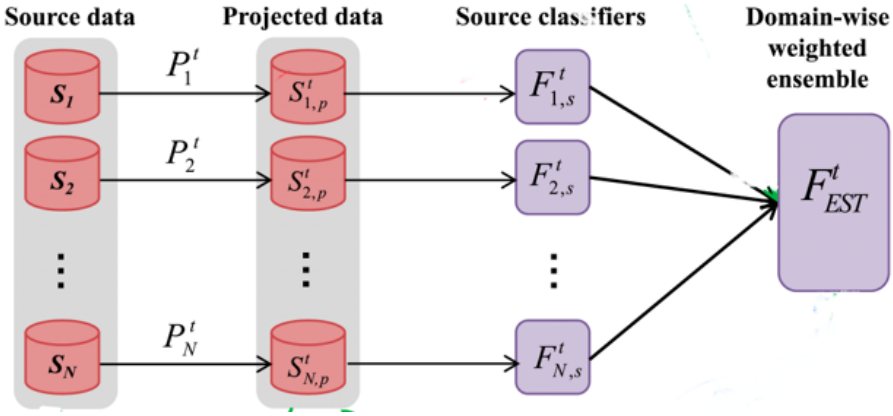

Figure 1.5: Coralation Alignment (CORAL) Overview.

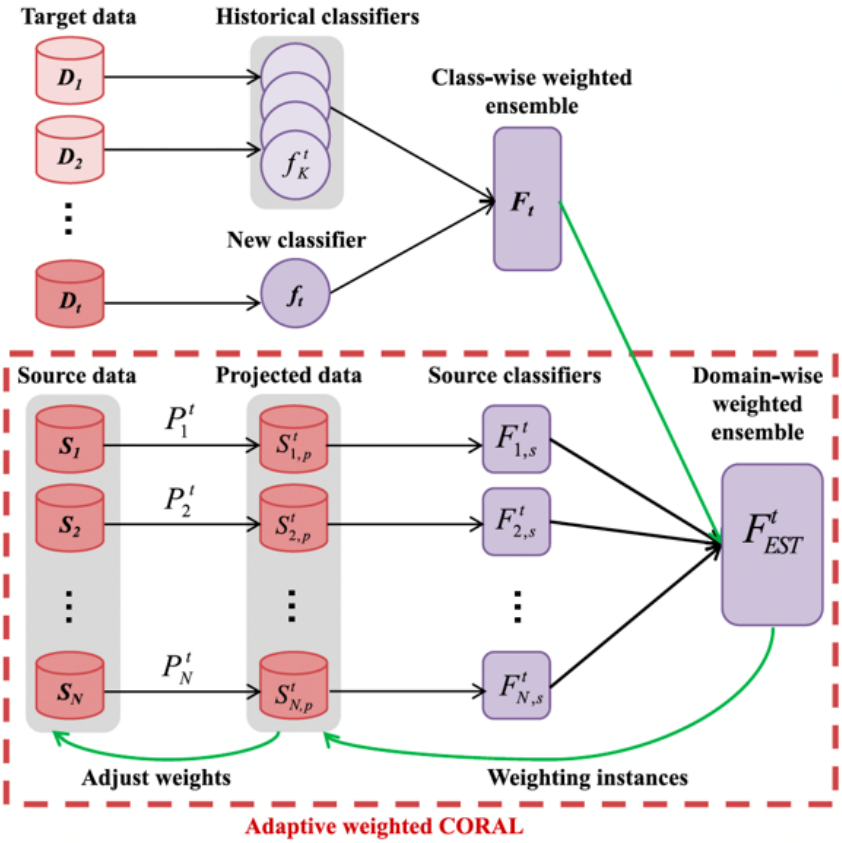

Figure 1.6: Concept Drift Transfer Learning (CDTL) Overview.

Table 1.3: Comparison of CORAL, Malanie and CDTL Methods

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

1.7 Remarks

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

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

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

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

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

As long as I have a want, I have a reason for living. Satisfaction is death.

George Bernard Shaw

CHAPTER

2

An Analysis of Heuristic Metrics for MCS Pruning

Including more classifiers in the classification task may provide more discriminating power with an equal or weighted contribution of each classifier to the final decision [1]. However, the increasing size of the ensemble hardly copes with the increasing demand to speed up the decision and to save computational resources. In bagging [2], an independent set of classifiers are generated in random order, then the final decision is adopted by a simple majority voting-based aggregation. While, it has been demonstrated that reordering the generated pool and selecting the first subset of classifiers impacts the ensemble size and the composite accuracy positively [1, 3–7]. The first subset of classifiers from the ordered list is expected to perform better than aggregating the whole list.

It has been proved that the generalization performance of a subensemble reports superior results over the traditional combination approaches, such as majority voting of the whole ensemble [4, 8]. Moreover, pruning down the redundant models reduces the memory burden [9]. Regarding that, MCS pruning is a proven mechanism to enhance the efficiency and elevate the efficacy of classification ensemble systems [1, 4, 5, 7]. From the review of the different pruning strategies in chapter

??, ordered-based pruning is a fast strategy with proven accuracy. Those strategies have the following merits: (1) return subsets that are close to optimal solution (*Efficacy*) [4]. (2) easy adaptation to any given storage and computational restrictions [1, 6]. (3) the time complexity of those strategies is low, in comparison with exhaustive or optimization-based search methods (*Efficiency*) [4].

The classifiers can be reordered via a greedy search, where the set of classifiers that are expected to perform better are aggregated first. Sequentially a new subset S_u is constructed from S_{u-1} by incorporating a single classifier from L_{u-1}; $S_u = S_{u-1} \cup \Psi_k \Big| \Psi_k \in L_{u-1}$, where $T = S_u \cup L_u$ and $u = \{1, 2, 3, \ldots, T\}$. Such that the single classifier selection from L_{u-1} is guided upon *a heuristic metric* to optimize the augmented ensemble S_u. The number of iterations or subensemble size, $\hat{T}$, can be controlled in advance to meet the computational restrictions. Furthermore, some metrics were proposed to rank all the classifiers in one batch without the sequential search. While, the property of the base classifier, an individual's accuracy, is not effective to determine this rank [3]. The generated ensemble needs to consider the hybrid between an individual's performance and an individual contribution to the ensemble diversity [6, 7]. Where it has been confirmed that the weakness of individuals can be compensated by the consensus of correct peers over different samples.

Since the practical analysis of the power of greedy search methods in [4], many research efforts have been directed to propose new heuristic measures to guide the selection of subensemble [1, 5–7]. Till now and related to our best knowledge, no work has considered the analysis of all those promising metrics together. This chapter fills that gap by comparing all these new techniques with the best performing techniques found in [4], and against other popular baseline metrics [3, 10].

In this chapter, we shed light on the importance of ordering-based ensemble pruning metrics. Reviewing and analyzing popular and recent metrics that work for bagging-based ensembles. We present a sophisticated analysis of how the pruning metrics can be affected by; the initial ensemble size (T), the required subensemble size ($\hat{T}$), the individual classifier type, and the binary or multiclass classification task. In addition, this study can be considered as a secure methodology to elevate the performance of bagging-like ensembles. This analysis is promising due to the

precision, prediction consistency, time-complexity, and space complexity of the investigated metrics to meet different computational restrictions.

Finally, this chapter is organized as follows: In Section 2.1, we present the main motivations to conduct this research and our contributions. The heuristic metrics in detail are to be introduced in Section 2.2. The experimental results and the statistical analysis are presented in Section 2.3. Finally, the conclusions are presented in Section 2.4.

2.1 Motivations and Contributions

The contribution of this proposal can be highlighted in the following points:

1. Focusing on static ensemble selection as an active research topic in multiple classifier systems.
2. Analyzing the effectiveness of different heuristic metrics to reorder the randomly bagging ensembles.
3. Separate analysis of those metrics over binary and multiclass classification tasks.
4. As far as we know, we are the first to group recent and efficient heuristic metrics for reordering bagging ensembles since they were analyzed by Martínez-Muñoz et al. [4] in 2009.

2.2 Heuristic Metrics

The investigated metrics are based on modifying the order of the classifiers in the bagging algorithm with the selection of the first set in the queue. Those techniques comprise dissimilar heuristic measures as: ensemble diversity [7], ensemble margin [3, 5], margin hybrid diversity [6], discriminating classifiers [1], ensemble error [10], Complementariness of misclassification [3], and relative accuracy with minimum redundancy [1]. Table 2.1 shows the heuristic metrics to be analyzed in this chapter over 30 datasets, divided into two parts as 15 binary datasets and 15 multiclass datasets.

Table 2.1: Heuristic metrics to guide the ordered bagging ensembles.

Name	Section	Heuristic Measure	Year	Ref.
RE	2.2.1	Reduced Ensemble Error	1997	[10]
CC	2.2.2	Complementary of Misclassification	2004	[3]
MDSQ	2.2.3	Supervised Ensemble Margin	2009	[4]
EPIC	**??**	Diversity Contribution of Individuals	2010	[7]
UMEP	**??**	Unsupervised Ensemble Margin	2013	[5]
MDEP	**??**	Margin & Diversity	2018	[6]
MRMR	2.2.4	Max. Relevance & Min. Redundancy	2018	[1]
DISC	2.2.5	Discriminant Classifiers	2018	[1]

2.2.1 Reduce-Error Pruning

Reduce error pruning (RE) was firstly proposed in [10]. The classifier with the highest (lowest) accuracy (error), as estimated on the pruning set D_{pr}, is stored in S_1 as the initial subset to be extended. The sequential addition of more classifiers, one at a time, is performed to get as much (less) accuracy (error) as possible. This heuristic incorporates into the subensemble the classifier s_u as:

$$s_u = \arg\max_{k} \sum_{(\mathbf{x}_i, y_i) \in D_{pr}} \left[\hat{\Psi}_{S_{u-1} \cup \Psi_k}(\mathbf{x}_i) = y_i \right] \tag{2.1}$$

where the index $k \in L_{u-1}$ and $S_u = S_{u-1} \cup \{s_u\}$. That metric has been applied in many articles as a baseline for the comparison purpose [1, 5] with superior performance over the unpruned ensemble [4].

2.2.2 Complementariness measure

Complementariness measure (CC) was proposed in [3] and it considers the complementariness between the incorporated models. The first subset, S_1, is initialized by selecting the classifier with the highest accuracy on D_{pr}. Then, the classifier to be nominated is the one with the highest prediction accuracy over the set of instances that are misclassified by S_{u-1}:

$$s_u = \arg\max_{k} \sum_{(\mathbf{x}_i, y_i) \in D_{pr}} \left[\hat{\Psi}_{S_{u-1}}(\mathbf{x}_i) \neq y_i \wedge \Psi_k(\mathbf{x}_i) = y_i \right] \tag{2.2}$$

where $k \in L_{u-1}$ and $S_u = S_{u-1} \cup \{s_u\}$. With this heuristic, the ensemble decision is expected to be shifted towards the correct classification. However, this metric

concentrates only on the misclassified samples with no restriction to preserve the previous correct decisions.

2.2.3 Supervised Ensemble Margin

Margin distance minimization (MDSQ) is introduced in [3, 4], where the decision space of the individual members over the selection set, D_{pr}, is transformed into signature vectors. The signature vector of Ψ_k, $r^{(k)}$, is defined by an N-dimensional vector whose ith component is calculated as:

$$r_i^{(k)} = 2\left[\Psi_k(\mathbf{x}_i) = y_i\right] - 1 \quad (\mathbf{x}_i, y_i) \in D_{pr} \tag{2.3}$$

The quantity $r_i^{(k)}$ will be 1, if the kth classifier correctly classifies the ith example in D_{pr}, otherwise it will be -1. The ensemble signature vector, $\langle R \rangle$, is defined as the average sum of all $r^{(k)}$ as:

$$\langle R \rangle = T^{-1} \sum_{k=1}^{T} r^{(k)} \tag{2.4}$$

The subensemble whose average signature vector $\langle R \rangle$ is in the first quadrant, that is all the components are positive, correctly classifies all the examples in D_{pr}. The objective is to select a subensemble whose $\langle R \rangle$ is as close as possible to a reference vector, O, placed somewhere in the first quadrant. Hence, the reference vector is mathematically represented as:

$$O_i = q \quad , i = \{1, 2, \ldots, N\} \quad and \quad 0 < q < 1 \tag{2.5}$$

The promoted classifiers are the ones with the minimum distance between their $\langle R \rangle$ and O and can be selected sequentially by minimizing:

$$s_u = \arg\min_k \; d\left(O, T^{-1}\left(r^{(k)} + \sum_{t=1}^{u-1} r^{(t)}\right)\right) \tag{2.6}$$

where $k \in L_{u-1}$ and $d(\mathrm{O},\langle R \rangle)$ is the usual Euclidean distance. The constant q should be sufficiently small, 0.075, to progressively focus on hard samples to be classified. Therefore, a subensemble with a large number of small positive values in $\langle R \rangle$ is preferred. By contrast, if the value of q is close to 1 the effectiveness of the

method will be diminished as the selection will be guided upon the easy samples. In this chapter, the modified version of this metric is applied with a moving reference point $q^{(u)} = 2\sqrt{2u}/T$ as it was discussed in [4].

2.2.4 Maximum Relevance & Minimum Redundancy

Maximum Relevance & Minimum Redundancy pruning (MRMR) was recently proposed in [1]. It is inspired by the popular algorithm mRMR [11, 12] for reducing redundancy in feature selection problem. The metric involves two relationships; one is between the candidate class and the component class, and the other is between the candidate class and the target class. The candidate class represents the class label output of the kth classifier to be included, while the component class represents the class label output of the composite ensemble. The classifier with the highest accuracy, estimated on the pruning set D_{pr}, is stored in S_1 as the initial subset to be extended. The next kth classifier to be incorporated, s_u, is selected according to:

$$s_u = \arg\max_k \left[I(\Psi_k; Y) - \frac{1}{u-1} \sum_{\Psi_i \in S_{u-1}} I(\Psi_k; \Psi_i) \right] \tag{2.7}$$

where $I(m, n)$ is the mutual information of variable m and n; Y is the target class; $k \in L_{u-1}$ and $S_u = S_{u-1} \cup \{s_u\}$. The classifier to be selected is the one with the maximum relevance with the target class, $I(\Psi_k; Y)$, and simultaneously with minimum redundancy with S_{u-1}, $\frac{1}{u-1} \sum_{\Psi_i \in S_{u-1}} I(\Psi_k; \Psi_i)$.

2.2.5 Discriminant Classifiers

Discriminant classifiers pruning (DISC) has also been proposed in [1]. The good classifier to be incorporated is the one to compensate the current subensemble, S_{u-1}, taking into account the following two assumptions.

- *Assumption (1)*: Regarding the samples correctly classified by S_{u-1}, a good candidate is expected to do the same decisions on as many of such samples as possible.

- *Assumption (2)*: In relation to the samples misclassified by S_{u-1}, a good candidate is expected to classify correctly as many of those instances as possible.

The first assumption relates to the candidate classifier and the composite ensemble, while the second assumption represents how the candidate classifier relates to the target. This metric concentrates on finding the most discriminant classifier, which is relative to both S_{u-1} and Y. The instances are divided into two parts; $\{mis\}$ represents the misclassified set by S_{u-1}, while $\{cor\}$ represents the set which is correctly classified by S_{u-1}. The incorporated classifier is selected as:

$$s_u = \arg\max_k \left[I({\Psi_k}^{mis}; Y^{mis}) + \frac{1}{u-1} \sum_{\Psi_i \in S_{u-1}} I({\Psi_k}^{cor}; \Psi_i^{cor}) \right] \tag{2.8}$$

where $k \in L_{u-1}$ and $S_u = S_{u-1} \cup \{s_u\}$. The first term $I({\Psi_k}^{mis}; Y^{mis})$ is the mutual information that Ψ_k can gain from the true labels Y according to the mislabeled instances by S_{u-1}. Whereas the second term $\frac{1}{u-1} \sum_{\Psi_i \in S_{u-1}} I({\Psi_k}^{cor}; \Psi_i^{cor})$ is the average mutual information that Ψ_k can gain from all Ψ_i members of S_{u-1} related to the correct classified samples.

2.3 Experimental Results

The experiments are dedicated to achieve objective 3, to group and analyze fast and accurate heuristic metrics for MCS pruning. The five main questions to be answered are:

- $\boldsymbol{Q_5}$. How the initial classifier pool size and the required subensemble size affect the performance of heuristic pruning metrics?
- $\boldsymbol{Q_6}$. How the heuristic pruning metrics are affected by the individual classifier type?
- $\boldsymbol{Q_7}$. How the efficacy of the pruning metrics could be affected by binary and multi-class datasets?
- $\boldsymbol{Q_8}$. How the pruning metrics are effective to reduce the performance variance?
- $\boldsymbol{Q_9}$. How the efficiency of the heuristic pruning metrics differs, in terms of time and space complexities?

2.3.1 Set Up

The design of experiments has considered the recommendations from [4] according to the following two issues: (A) The influence of training conditions (B) The influence of the initial pool size. *For training conditions*; the whole training data has been used, for both, to train the bagged ensemble and to prune it. *For the initial pool size*; the initial pool should contain a sufficient number of classifiers. However, the gained accuracy is not worthing the added complexity resulting from expanding the pool. In this chapter, two ensemble systems have been constructed as a part of the analysis.

1. Heterogeneous (*Different Classifier types-DC*)
    - *Samples*: Bootstrap samples, with replacement, are generated from the training data.
    - *Features*: Sixty percent of features are selected randomly for each classifier.
    - *Classifiers*: Five different classifier models, with their default setup parameters, have been used (*DT*[1], *NB*[2], *JRip*[3], *Multinom*[4] and *KNN*[5]) with 20% as a proportional representation by each model from the whole pool size.
2. Homogeneous (*Similar Classifiers-SC*)
    - Similar to the previous, while the difference is that all individual members are of type *DT*[1].

Finally, all the datasets are preprocessed by unifying the scales of the features via normalization. For each dataset, 10 repetitions of 10 fold cross-validation procedure have been tested to get 100 runs per dataset. In addition, MDEP depends on an internal parameter α; three values for MDEP with different $\alpha \in \{0.1, 0.5, 0.9\}$ are considered, and the best-optimized alpha according to the in train-validation is used to report the test for each dataset separately. The results for Random Forest (RF) [13], Adaboost (AdaB) [14], and the single best model (SBM) from the pool,

[1]Package C50 :https://cran.r-project.org/web/packages/C50
[2]Package e1071:https://cran.r-project.org/web/packages/e1071
[3]Package RWeka:https://cran.r-project.org/web/packages/RWeka
[4]Package nnet:http://cran.r-project.org/web/packages/nnet
[5]Package caret:https://cran.r-project.org/web/packages/caret

according to the measured accuracy of the models on the pruning set, are included as references in the comparison.

The default set-up for the individual classifiers types, RF, and AdaB are as follows: DT uses C5.0 decision trees of Quinlan [15] in its pruned version. NB uses Naïve Bayes with Laplace smoothing to solve the problem of zero probability. KNN applies k-nearest neighbor classification with $k = 3$ as the number of neighbors. RF[1] implements Breiman's random forest algorithm with ntrees=T, number of variables that are randomly sampled at each split=$sqrt(ncol(\mathbf{x}))$. AdaB[2] fits the AdaBoost.M1 using classification trees as base classifiers with iterations=T.

A total of 30 datasets that were obtained from OpenML[3] and KEEL [4] are used in this study for experimentation. The characteristics of the data are presented in Table 2.2, where #S, #F, #C, and R represent the number of samples, the number of features, the number of classes, and the ratio between the smallest, and the largest class for each dataset respectively. The number of classes varies from 2 to 10, while the maximum number of features is 100.

Table 2.2: Characteristics of the selected datasets for experimentation, *sorted by samples and classes*.

DataSet	#S	#F	#C	R	DataSet	#S	#F	#C	R
Breast-cancer	286	9	2	0.42	Wine	178	13	3	0.676
SPECTF	349	44	2	0.37	Newthyroid	215	5	3	0.2
Ionosphere	351	33	2	0.56	Cmc	1 473	9	3	0.529
Wdbc	569	30	2	0.594	Lymphography	148	18	4	0.025
Indian Liver Patient (ILP)	583	10	2	0.401	Vehicle	846	18	4	0.913
Australian	690	14	2	0.802	Wall-Following-Robot (WFR)	5 456	24	4	0.149
Wisconsin	699	9	2	0.526	Cleveland	297	13	5	0.081
Blood-transfusion	748	4	2	0.312	Dermatology	358	34	6	0.18
Mammographic	830	5	2	0.944	Flare	1 066	11	6	0.13
Tic-tac-toe	958	9	2	0.530	Wine quality-red	1 599	11	6	0.015
German	1 000	20	2	0.429	Satimage	6 435	36	6	0.408
Hill-valley	1 212	100	2	1.0	Segment	2 310	18	7	1
Kr-vs-kp	3 196	36	2	0.915	Led7digit	500	7	10	0.649
spambase	4 601	57	2	0.650	Mfeat-Karh	2 000	64	10	1
Ringnorm	7 400	20	2	0.981	Mfeat-Fourier	2 000	76	10	1

[1] randomForest:https://cran.r-project.org/web/packages/randomForest

[2] Adaboost:https://cran.r-project.org/web/packages/adabag

[3] Machine Learning Repository: https://www.openml.org

[4] KEEL Repository: http://www.keel.es

2.3.2 Influence of Pool Size and Selection Size

To answer Q_5, we analyze how ordered aggregation is affected by both the initial pool size (T) and the selection percentage (P) simultaneously. A heterogeneous ensemble, see Section 2.3.1, composed of 201 models is built. The classifiers will be ordered considering only the first 25, 51, 75, 101, 151, and 201 models from ensemble size. This means that the classifiers are nested such that all classifiers in an initial pool are also included in larger pools. The average accuracy over 100 runs is computed for each ensemble size T.

Table 2.3 shows the analysis of 1200 records ($T = 6 \times$ runs=100 $\times$ $P = 2$) for the *SPECTF* dataset. The last two columns represent the size of the ordered subensemble according to an initial pool of T. For each T and P, the best value is highlighted in bold. The prediction accuracy of the bagging increases monotonically as a function of the initial pool size, while this saturation level sometimes decreases according to the classification task. All the investigated metrics prove their superiority over bagging, *regarding SPECTF dataset*. The poor results reported for BSM, confirm that ensemble outperforms all single models from which it is composed. Regarding the selection percentage of $P = 30\%$, the accuracy of DISC, EPIC, and UMEP keep on increasing as more classifiers are included in the initial pool. *In general*, the accuracy of the ordered classifiers with a percentage of 30% is better than 20%. In addition, the poorest accuracy by any ordering metric is better than the highest accuracy of bagging. The highest accuracy of 91.49 is reported by DISC using a subensemble composed of 61 classifiers, $\hat{T}$, instead of the 201 classifiers used by bagging.

Figure 2.1 shows the complementary perspective about the results. The comparison of DISC-30% with DISC-20% and the comparison of UMEP-30% with UMEP-20% confirm that the ordered subset with a larger number of classifiers returns higher accuracy. Furthermore, the lowest accuracy by UMEP-20%, *horizontal-dashed line*, with only 5 classifiers outperforms the accuracy of bagging for any size. The prediction accuracy of DISC-30% and UMEP-30% keeps on increasing even after bagging has stabilized. In addition, the figure shows that BSM is the worst ensemble selection strategy. We confirm that the main objective of heuristic metrics is to return a subensemble with a reduced number of classifiers while

Table 2.3: The average accuracy of the subensemble related to a selection percentage (P) from an inital pool size (T); *SPECTF dataset*.

T	Bagging	BSM	RE	CC	MDSQ	MRMR	DISC	EPIC	UMEP	MDEP	P	$\hat{T}$
25	**84.73**	81.87	86.73	86.21	87.23	86.43	**88.43**	86.49	86.65	86.77	20%	5
			87.66	87.11	88.18	87.19	**88.69**	87.77	87.74	87.71	30%	7
51	**85.27**	82.96	89.22	88.41	89.91	88.35	**90.41**	89.65	89.54	89.72	20%	11
			88.97	88.54	89.82	88.62	**90.49**	89.65	89.76	89.80	30%	15
75	**85.48**	82.92	88.89	88.80	90.24	88.12	**91.10**	90.28	90.83	90.66	20%	15
			88.80	88.94	89.69	88.20	**90.89**	90.38	90.32	90.21	30%	23
101	**85.85**	83.09	89.08	89.06	89.90	87.77	**91.21**	90.56	90.22	90.44	20%	21
			88.93	88.80	88.83	88.14	**91.01**	90.84	90.41	90.78	30%	31
151	**85.82**	82.78	89.18	88.89	89.29	88.54	**91.09**	90.58	90.59	90.44	20%	31
			88.82	88.97	88.03	88.91	**91.24**	90.64	90.78	90.70	30%	45
201	**85.84**	83.68	89.12	89.19	88.48	88.82	91.07	91.07	**91.15**	91.10	20%	41
			89.22	88.88	87.36	88.70	**91.49**	91.00	91.04	91.09	30%	61

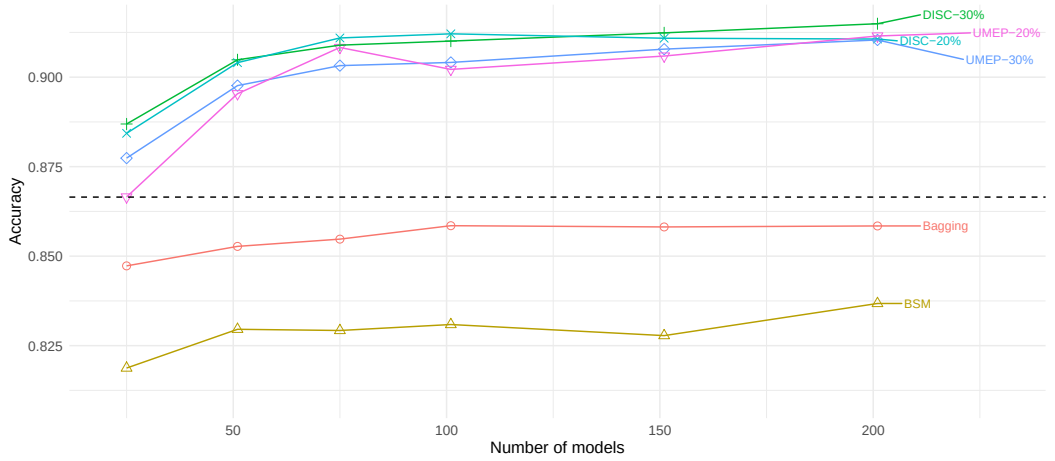

Figure 2.1: Influence of pool size and selection size on the general accuracy; *SPECTF dataset*.

keeping the accuracy unaffected. Going beyond the accuracy of the bagging is conditioned by the effectiveness to reorder the ensemble as we will discuss in Sections 2.3.4 and 2.3.5.

2.3.3 Influence of Heterogeneous Classifiers

To answer Q_6, we analyze how the general accuracy and the performance of the metrics are affected by the type of combined individual models. For that, the initial pool size is fixed with $T = 101$ as a balance between accuracy and ensemble's

complexity. Then the two types of ensembles, *homogeneous and heterogeneous*, described in Section 2.3.1, are constructed for comparison purposes. Table 2.4 presents the average accuracy and the standard deviation for the ensembles using different and similar individual classifiers over six representative datasets.

Table 2.4: Average accuracy and standard deviation for Different Classifiers (DC) and Similar Classifiers (SC); T= 101 and P=30%.

	DC				SC			
Dataset	Bagging	RE	DISC	UMEP	Bagging	RE	DISC	UMEP
Ensemble size	$T = 101$	$\hat{T} = 31$			$T = 101$	$\hat{T} = 31$		
Australian	86.70 ± 3.39	**86.97 ± 3.52**	**86.97 ± 3.48**	**87.36 ± 3.35**	86.85 ± 3.43	86.66 ± 3.85	86.34 ± 3.81	**86.95 ± 3.67**
Blood	77.37 ± 1.77	77.35 ± 2.68	**77.53 ± 2.74**	77.03 ± 2.73	76.27 ± 0.73	**77.53 ± 2.77**	**77.92 ± 2.47**	**77.14 ± 2.80**
Breast-cancer	73.41 ± 5.73	**73.94 ± 6.84**	**73.45 ± 6.03**	72.97 ± 6.90	73.48 ± 5.08	73.02 ± 5.63	73.10 ± 5.58	71.41 ± 6.39
Cmc	53.30 ± 3.74	53.29 ± 3.83	**53.80 ± 3.72**	**53.82 ± 3.69**	53.33 ± 3.84	**53.38 ± 3.41**	**53.57 ± 3.58**	**53.65 ± 3.61**
Mammographic	83.04 ± 4.02	82.72 ± 4.00	82.59 ± 4.35	**83.87 ± 4.02**	83.70 ± 3.41	82.99 ± 3.62	83.19 ± 3.56	83.64 ± 3.53
Wdbc	96.63 ± 2.42	**97.10 ± 2.32**	**97.44 ± 2.03**	**97.54 ± 1.95**	96.58 ± 2.23	96.48 ± 2.40	**96.76 ± 2.32**	**96.67 ± 2.27**
AR-Friedman	5.33	4.42	3.33	**3**	5	5.75	**4.33**	4.83

The results prove that the general accuracy of bagging can be outperformed in both cases by using pruned ensemble, *highlighted bold values in each part*. To differentiate among the methods, the average rank of Friedman test [16] *(AR-Friedman)* is calculated and is shown in the last row of Table 2.4. The diversity in decision space, which is achieved by different classifiers, guarantees effective performance for the ordering methods. The best ranks, upon the selected datasets, are reported for DC-UMEP, DC-DISC, and DC-RE in comparison with their versions under similar classifiers. We conclude that similar classifiers produce similar decisions approximately, and the ability to differentiate among them by ordering metrics decreases. For the rest of our experiments, an ensemble using different classifier types with fixed pool size, $T = 101$, will be formed for the evaluation purpose.

2.3.4 Analysis Over Binary Datasets

To answer first part of Q_7, for solving binary datasets. Table 2.5 shows the average accuracy and standard deviation for the different datasets. Adaboost achieves the highest accuracy in six datasets $\{D_6, D_7, D_8, D_{10}, D_{11}, D_{13}\}$ while it uses 101 classifiers. RE and MDEP both of them achieve the highest accuracy for $\{D_3\}$

Table 2.5: Average accuracy and standard deviation over binary datasets for ensemble size $T = 101$ and P=30%. The values that outperform bagging are highlighted in bold.

		$T = 101$	$\hat{T} = 1$	$\hat{T} = 31$								$T = 101$	
#	file	Bagging	BSM	RE	CC	MDSQ	MRMR	DISC	EPIC	UMEP	MDEP	AdaB	RF
D_1	Australian	86.70 ± 3.39	84.03 ± 4.13	**86.97 ± 3.52**	**86.87 ± 3.39**	86.47 ± 3.78	**86.75 ± 3.44**	**86.97 ± 3.48**	**87.29 ± 3.38**	**87.36 ± 3.35**	**87.38 ± 3.32**	**86.94 ± 3.70**	**86.97 ± 3.57**
D_2	Blood-transfusion	77.37 ± 1.77	73.69 ± 4.33	77.35 ± 2.68	**77.70 ± 2.85**	**77.65 ± 2.79**	**77.70 ± 2.80**	**77.53 ± 2.74**	77.34 ± 2.96	77.03 ± 2.73	77.36 ± 2.95	76.09 ± 3.85	75.31 ± 4.10
D_3	Breast-cancer	73.41 ± 5.73	68.64 ± 8.16	**73.94 ± 6.84**	**73.45 ± 5.98**	73.03 ± 4.99	**73.77 ± 6.10**	**73.45 ± 6.03**	73.28 ± 6.63	72.97 ± 6.90	73.32 ± 6.56	68.75 ± 7.62	71.50 ± 8.06
D_4	German	74.98 ± 2.90	69.26 ± 4.43	**75.65 ± 3.24**	**75.48 ± 3.10**	74.67 ± 2.87	**75.19 ± 3.22**	**75.77 ± 3.06**	**75.52 ± 3.24**	**75.77 ± 3.19**	**75.77 ± 3.22**	**75.39 ± 3.56**	**76.09 ± 3.30**
D_5	Hill-valley	67.81 ± 5.09	**86.00 ± 4.66**	**82.80 ± 6.54**	**80.10 ± 7.26**	**78.47 ± 5.23**	**80.04 ± 7.29**	**79.23 ± 5.32**	**74.82 ± 9.06**	**81.71 ± 4.12**	**81.79 ± 4.17**	59.06 ± 3.78	57.58 ± 3.74
D_6	ILP	70.33 ± 5.55	67.22 ± 5.13	69.99 ± 5.20	68.96 ± 5.75	70.28 ± 5.10	68.84 ± 5.76	70.30 ± 5.13	**71.44** ± 4.36	**71.29 ± 5.03**	**71.43 ± 4.59**	**71.59 ± 5.06**	**70.52 ± 5.30**
D_7	Ionosphere	93.37 ± 3.71	89.25 ± 5.54	**93.48 ± 3.81**	**93.54 ± 3.79**	**93.90 ± 3.11**	**93.60 ± 3.89**	**93.80 ± 3.75**	**93.60 ± 3.90**	**93.80 ± 3.89**	**93.85 ± 3.88**	**94.08 ± 3.30**	93.25 ± 4.05
D_8	Kr-vs-kp	96.25 ± 1.20	**96.46 ± 1.46**	**98.42 ± 0.83**	**98.25 ± 0.76**	**97.92 ± 0.88**	**98.20 ± 0.77**	**98.37 ± 0.68**	**96.76 ± 1.23**	**96.73 ± 1.23**	**96.76 ± 1.22**	**99.62 ± 0.33**	**98.67 ± 0.68**
D_9	Mammographic	83.04 ± 4.02	82.46 ± 4.27	82.72 ± 4.00	82.21 ± 4.36	82.35 ± 3.90	82.37 ± 4.38	82.59 ± 4.35	**83.94 ± 3.77**	**83.87 ± 4.02**	**84.00 ± 3.97**	80.73 ± 4.18	81.78 ± 4.09
D_{10}	Ringnorm	96.66 ± 0.60	93.72 ± 2.11	**96.77 ± 0.64**	**96.77 ± 0.59**	95.07 ± 0.69	**96.79 ± 0.60**	**96.85 ± 0.61**	**96.80 ± 0.66**	**96.79 ± 0.68**	**96.79 ± 0.68**	**97.33 ± 0.55**	94.98 ± 0.80
D_{11}	Spambase	94.98 ± 1.07	91.52 ± 1.44	**95.04 ± 1.00**	94.75 ± 1.04	94.72 ± 1.04	94.76 ± 1.01	**94.99 ± 1.15**	94.93 ± 1.06	94.85 ± 1.11	94.91 ± 1.08	**95.72 ± 0.87**	**95.31 ± 0.98**
D_{12}	SPECTF	85.96 ± 5.32	82.18 ± 5.98	**89.17 ± 4.78**	**89.20 ± 4.85**	**88.48 ± 5.50**	**88.64 ± 4.85**	**91.35 ± 4.71**	**90.66 ± 4.75**	**90.84 ± 4.74**	**90.92 ± 4.69**	**91.24 ± 3.96**	**92.39 ± 4.30**
D_{13}	Tic-tac-toe	76.57 ± 2.85	**77.49 ± 4.41**	**86.10 ± 3.14**	**86.18 ± 3.24**	**86.65 ± 3.24**	**86.23 ± 3.17**	**86.53 ± 3.18**	**87.08 ± 2.98**	**85.66 ± 3.22**	**87.06 ± 3.34**	**99.47 ± 0.85**	**98.72 ± 1.18**
D_{14}	Wdbc	96.63 ± 2.42	95.71 ± 2.55	**97.10 ± 2.32**	**97.10 ± 2.24**	**96.89 ± 2.10**	**97.17 ± 2.31**	**97.44 ± 2.03**	**97.42 ± 2.03**	**97.54 ± 1.95**	**97.54 ± 1.94**	**96.82 ± 2.13**	96.14 ± 2.37
D_{15}	Wisconsin	97.31 ± 2.01	95.20 ± 2.33	97.22 ± 1.95	97.22 ± 2.11	97.21 ± 2.15	97.19 ± 2.18	97.18 ± 2.15	97.27 ± 2.04	97.12 ± 1.95	97.22 ± 2.14	96.83 ± 2.13	97.12 ± 1.84
AR-Friedman		8	10.8	**5.6**	**6.73**	**7.8**	**6.73**	**4.6**	**5.07**	**5.83**	**4**	**5.87**	**6.97**

and $\{D_1, D_9, D_{14}\}$ respectively using 31 classifiers. The highest improvement percentage of 14.52% has been recorded by AdaB in comparison with MDEP for D_{13}. While MDEP recorded the highest improvement over AdaB for D_5 by 38.49%. For almost all the datasets, the reordering metrics guarantee higher accuracy and go beyond what can be achieved by bagging. MDSQ is the only metric with a tendency to form the worst subensemble related to the investigated tasks. For the *Hill-valley* dataset, the poor performance of bagging is caused by the uneven response of individual members. The performance analysis of RE to select a subensemble over the 100 executions, depends on grouping specific individual types. Where DT, Multinom, NB, JRip, and KNN are represented in-order according to the following percentages 33.97%, 28.97%, 20.48%, 10.45%, 6.13% respectively.

The average rank of Friedman test [16], *AR-Friedman*, is presented in the last row of Table 2.5 with the best ranks being (in order) MDEP, DISC, EPIC, RE, UMEP, AdaB, CC, and MRMR. Next, the Wilcoxon test [17, 18] for pairwise comparison has been performed to detect significant differences between the two sample means. From Table 2.6, BSM is the worst ensemble selection strategy. All heuristic metrics, except MDSQ, significantly outperform bagging by 95% or 90%. Furthermore, we notice that MDSQ, AdaB, and RF are at the same level of accuracy as bagging. While the heterogeneous classifiers selected by RE and CC significantly outperform bagging. Finally, MDEP is the only metric that significantly outperforms both EPIC and UMEP by 95% and is the best metric regarding the investigated tasks.

2.3.5 Analysis Over Multiclass Datasets

To answer second part of Q_7, to analyze the stability of the different pruning methods in multiclass classification tasks. Table 2.7 shows the average accuracy and the standard deviation over multi-class datasets. Adaboost and RF achieve the highest accuracy in datasets $\{D_{17}, D_{26}, D_{27}, D_{28}\}$ and $\{D_{16}, D_{25}, D_{29}, D_{30}\}$, respectively using the complete set of 101 classifiers. The highest improvement percentage of Adaboost over DISC has been recorded by 3.94% for D_{27}. While DISC recorded the highest improvement over Adaboost by 12.77% for D_{30}.

Table 2.6: Summary of the Wilcoxon test (for binary datasets). •= the method in the row improves the method of the column. ○= the method in the column improves the method of the row. Upper diagonal of level significance $\alpha = 0.9$, lower diagonal level of significance $\alpha = 0.95$.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
Bagging (1)	-	•	○	○		○	○	○	○	○		
BSM (2)	○	-	○	○	○	○	○	○	○	○	○	○
RE (3)	•	•	-	•	•	•						
CC (4)		•		-			○			○		
MDSQ (5)		•	○		-		○		○	○		
MRMR (6)		•	○			-	○			○		
DISC (7)	•	•		•	•		-					
EPIC (8)	•	•						-		○		
UMEP (9)	•	•							-	○		
MDEP (10)	•	•		•	•	•		•	•	-		
AdaB (11)		•									-	
RF (12)		•										-

For datasets with a large number of samples and classes like $\{D_{22}, D_{23}\}$, as expected, we notice that DISC, MDEP are the best. Our explanation, for complex decision spaces, it will be preferred to select classifiers that are more discriminant or that can classify difficult samples. For that, DISC is more promising to acquire complementary information inside the subensemble by reducing the internal conflict. While MDEP is preferred to balance between the individual's accuracy and ensemble diversity. In addition, DISC is the best for D_{25} as the largest size multi-class dataset.

For datasets with a small number of classes and a small number of instances like $\{D_{16}, D_{21}, D_{24}\}$, the decision space becomes easier as low conflict will exist. For that, except MDSQ for D_{16}, RE proved its superiority to select effective subensemble based on its general accuracy to outperform margin-based metrics $\{$MDSQ, UMEP, MDEP$\}$. For datasets with a small number of classes and a larger number of instances like $\{D_{17}, D_{28}\}$; DISC, EPIC, and UMEP outperform RE.

For the statistical ranking, *AR-Friedman* is presented in the last row of Table 2.7 with the best ranks scored sequentially by DISC, EPIC, RF, UMEP, CC, MRMR & MDEP, and RE. While Table 2.8 shows the pairwise comparison between the 3 unpruned ensembles and the 9 pruned ones. Table 2.8 confirms that BSM is the worst selection/pruning strategy. All the investigated metrics, except MRMR,

Table 2.7: Average accuracy and standard deviation over multiclass datasets for ensemble size $T = 101$ and P=30%. The values that outperform bagging are highlighted in bold.

		$T = 101$	$\hat{T} = 1$	$\hat{T} = 31$								$T = 101$	
#	file	Bagging	BSM	RE	CC	MDSQ	MRMR	DISC	EPIC	UMEP	MDEP	AdaB	RF
D_{16}	Cleveland	57.07 ± 4.41	51.40 ± 7.76	**57.30 ± 5.32**	57.02 ± 4.75	**57.35 ± 4.91**	**57.11 ± 4.66**	**57.38 ± 4.86**	**57.43 ± 5.08**	56.89 ± 4.81	**57.21 ± 5.05**	**57.52 ± 5.43**	**57.72 ± 5.29**
D_{17}	Cmc	53.30 ± 3.74	49.58 ± 3.81	53.29 ± 3.83	53.20 ± 3.66	**53.36 ± 3.81**	**53.49 ± 3.97**	**53.80 ± 3.72**	**53.43 ± 3.69**	**53.82 ± 3.69**	53.14 ± 4.27	**56.30 ± 3.44**	**53.98 ± 3.42**
D_{18}	Dermatology	97.66 ± 2.57	93.75 ± 4.85	97.49 ± 2.62	97.52 ± 2.63	97.59 ± 2.44	97.57 ± 2.62	**97.88 ± 2.18**	**97.99 ± 2.14**	**97.96 ± 2.24**	**97.99 ± 2.14**	96.48 ± 2.64	97.65 ± 2.28
D_{19}	Flare	73.23 ± 3.41	**73.59 ± 3.27**	**74.65 ± 2.77**	**75.22 ± 3.03**	**75.31 ± 2.72**	69.89 ± 4.29	**75.60 ± 2.81**	**75.65 ± 2.82**	**75.57 ± 2.77**	**74.78 ± 3.63**	**75.26 ± 2.99**	**75.14 ± 2.45**
D_{20}	Led7digit	69.51 ± 5.28	60.31 ± 5.72	**72.12 ± 5.34**	**70.89 ± 5.69**	**73.17 ± 5.77**	**69.65 ± 6.07**	**71.39 ± 6.11**	**72.11 ± 5.88**	**71.22 ± 5.69**	**70.42 ± 6.52**	**72.66 ± 5.94**	**71.86 ± 5.64**
D_{21}	Lymphography	85.45 ± 8.78	77.45 ± 10.29	**86.34 ± 8.17**	**86.42 ± 8.75**	**86.23 ± 9.58**	**86.57 ± 9.17**	85.35 ± 9.39	84.94 ± 9.27	84.32 ± 9.59	85.30 ± 9.72	85.11 ± 9.01	84.55 ± 9.51
D_{22}	Mfeat-Fourier	83.10 ± 2.20	71.81 ± 2.96	**83.36 ± 2.51**	**83.38 ± 2.34**	**83.19 ± 2.43**	**83.58 ± 2.27**	**83.56 ± 2.46**	**83.20 ± 2.44**	**83.38 ± 2.40**	**83.58 ± 2.36**	81.86 ± 2.56	83.10 ± 2.17
D_{23}	Mfeat-karh	96.87 ± 1.08	90.75 ± 3.61	96.83 ± 1.13	96.78 ± 1.16	96.50 ± 1.41	96.75 ± 1.23	**97.13 ± 1.09**	96.74 ± 1.18	96.86 ± 1.17	**97.01 ± 1.11**	95.40 ± 1.47	96.04 ± 1.23
D_{24}	Newthyroid	96.55 ± 3.68	95.43 ± 4.73	**96.88 ± 3.51**	**96.84 ± 3.50**	96.32 ± 4.11	**96.65 ± 3.45**	**96.60 ± 3.69**	**96.69 ± 3.78**	95.91 ± 3.98	96.46 ± 3.92	95.24 ± 4.07	96.08 ± 4.04
D_{25}	Satimage	89.66 ± 1.14	85.09 ± 1.39	**91.48 ± 1.02**	**91.63 ± 1.05**	**91.09 ± 1.04**	**91.61 ± 1.02**	**91.67 ± 1.04**	**91.52 ± 1.02**	**91.54 ± 0.98**	**91.50 ± 0.98**	88.25 ± 1.24	**91.82 ± 0.95**
D_{26}	Segment	97.05 ± 1.07	95.88 ± 1.58	**97.94 ± 0.95**	**98.01 ± 0.90**	**97.49 ± 1.06**	**97.97 ± 0.90**	**98.09 ± 0.98**	**98.13 ± 0.95**	**98.12 ± 0.91**	**98.13 ± 0.92**	**98.55 ± 0.83**	**97.93 ± 1.01**
D_{27}	Vehicle	74.99 ± 4.07	69.23 ± 4.85	**75.56 ± 4.11**	**75.42 ± 3.92**	**75.14 ± 3.90**	**75.48 ± 4.10**	**75.32 ± 3.92**	**75.13 ± 4.03**	**75.14 ± 3.80**	**75.13 ± 3.86**	**78.29 ± 4.10**	**75.13 ± 4.29**
D_{28}	WFR	96.62 ± 0.85	**99.03 ± 0.53**	**99.04 ± 0.41**	**99.06 ± 0.38**	**98.95 ± 0.44**	**99.04 ± 0.43**	**99.40 ± 0.32**	**99.40 ± 0.32**	**99.40 ± 0.31**	**99.38 ± 0.38**	**99.88 ± 0.14**	**99.43 ± 0.32**
D_{29}	Wine	98.11 ± 3.07	94.45 ± 5.44	97.94 ± 3.41	97.94 ± 3.41	97.81 ± 3.33	97.77 ± 3.53	**98.16 ± 3.17**	**98.16 ± 3.07**	**98.16 ± 3.07**	**98.16 ± 3.07**	96.20 ± 4.54	**98.22 ± 3.24**
D_{30}	Winequality-red	63.40 ± 2.97	56.71 ± 4.30	**68.38 ± 2.99**	**68.57 ± 3.01**	**68.30 ± 3.31**	**68.76 ± 3.08**	**68.55 ± 3.08**	**68.44 ± 2.92**	**68.06 ± 2.98**	**67.72 ± 3.13**	60.79 ± 3.30	**70.53 ± 3.44**
AR-Friedman		8.5	11.67	**6.13**	**5.87**	**7.03**	**6.07**	**3.9**	**4.83**	**5.83**	**6.07**	**6.6**	**5.5**

significantly outperform bagging by 95% according to both the accuracy and the ensemble size. Adaboost and RF are at the same level of accuracy with all pruning metrics, however, their performance is achieved using T=101 classifiers. MDEP waived his rank in favor of DISC and EPIC. While DISC is the best metric for selecting subensemble according to the investigated datasets. It has been confirmed that ordered bagging based on complementary decisions works well for multiclass datasets, *confirmed by CC performance*. Next, it is interesting to combine all the binary and multiclass datasets to analyze ordering based pruning metrics in general.

Table 2.8: Summary of the Wilcoxon test (for Multiclass datasets). •= the method in the row improves the method of the column. ∘= the method in the column improves the method of the row. Upper diagonal of level significance $\alpha = 0.9$, lower diagonal level of significance $\alpha = 0.95$.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
Bagging (1)	-	•	∘	∘	∘	∘	∘	∘	∘	∘		∘
BSM (2)	∘	-	∘	∘	∘	∘	∘	∘	∘	∘	∘	∘
RE (3)	•	•	-									
CC (4)	•	•		-			∘					
MDSQ (5)	•	•			-		∘					
MRMR (6)		•				-	∘					
DISC (7)	•	•					-		•	•		
EPIC (8)	•	•						-				
UMEP (9)	•	•					∘		-			
MDEP (10)	•	•					∘			-		
AdaB (11)		•									-	
RF (12)	•	•										-

2.3.6 General Analysis Over All Datasets

A nonparametric statistical test is conducted over the thirty datasets, D_1 to D_{30}, to check if there is a significant difference between the performance of the ordered bagging methods or not. Using the methodology proposed by Demšar [19], Figure 2.2 shows the Nemenyi post hoc test for $\alpha = 0.05$. Methods that are connected are not significantly different based on the absolute difference in the average rankings. The Critical Difference is shown (CD=3.06 for 12 methods, 30 datasets, $\alpha = 0.05$). The analysis shows that DISC, EPIC, and MDEP significantly out-

perform bagging by 95%. While, the inferior performance is recorded by BSM, Bagging, and MDSQ.

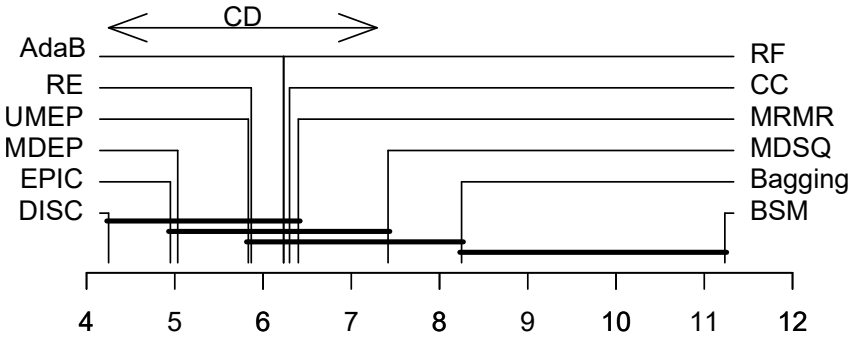

Figure 2.2: Comparison of the different metrics over the 30 datasets using the Nemenyi test. Methods not significantly different (α=0.05) are connected together.

2.3.7 Prediction Consistency

To answer Q_8, after statistical analysis and according to the Nemenyi test, we selected {DISC, EPIC, MDEP, UMEP, AdaB, RF, Bagging, BSM}, and analyzed the distribution of their prediction accuracy over the 100 executions. Figure 2.3 presents the range of the prediction accuracy around the median and how the heuristic metrics realize robust and stable predictions, with less number of internal classifiers, for a set of representative datasets $\{D_1, D_3, D_4, D_6, D_9, D_{19}, D_{25}, D_{28}\}$. The conclusion is that the ordering metrics are more effective to select a promising subensemble and their behavior reduces the performance variance.

2.3.8 Efficiency Analysis

As demonstrated in [4], the efficiency of reordering metrics can be evaluated according to the following three aspects: the computational cost to extract the pruned subensemble, the required memory space to store the pruned ensembles, and the classification speed. While, some steps can be performed in parallel: the generation of the initial pool of classifiers, and the retrieving of classification decisions from the selected classifiers.

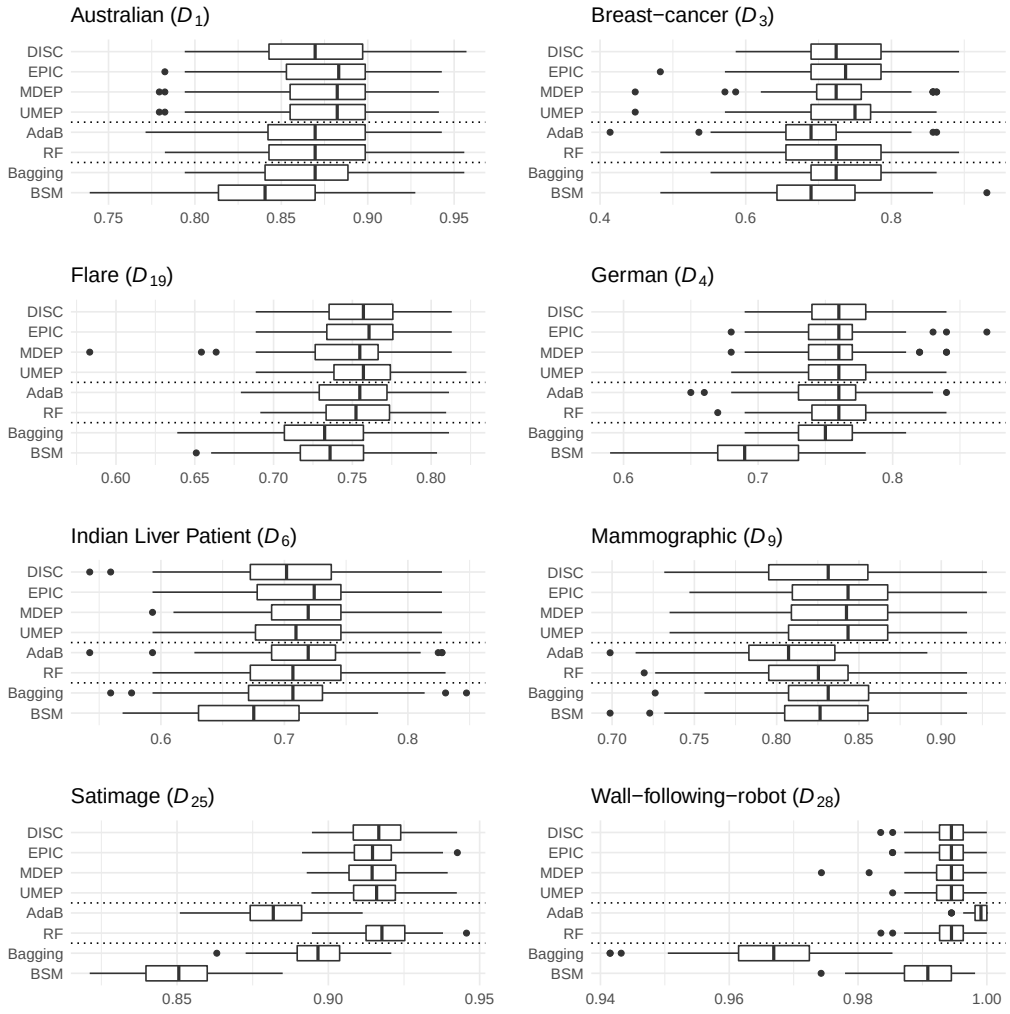

Figure 2.3: Distribution of the prediction accuracy.

To answer Q_9, space and time complexities of the different heuristic metrics are summarized in Table 2.9 in terms of T, N, M. The memory requirements are estimated assuming that the decisions of the classifiers are stored in a matrix of size $N \times T$. For large datasets, it might be difficult to store this matrix in memory. In such a case, the whole matrix can be stored to a secondary memory device like the hard disk [4]. This would reduce the memory requirements to $\mathcal{O}(N)$, but the required disk access will slow down the classification process.

To empirically investigate how the heuristic measures depend on T, a series of

Table 2.9: Space and time complexities of different metrics.

Metric	Space complexity	Time complexity
RE	$\mathcal{O}(N \cdot T + M)$	$\mathcal{O}(T^2 \cdot N \cdot M)$
CC	$\mathcal{O}(N \cdot T + M)$	$\mathcal{O}(T^2 \cdot N)$
MDSQ	$\mathcal{O}(N \cdot T)$	$\mathcal{O}(T^2 \cdot N)$
MRMR	$\mathcal{O}(N \cdot T + M^2)$	$\mathcal{O}(T^2 \cdot N \cdot M)$
DISC	$\mathcal{O}(N \cdot T + M^2)$	$\mathcal{O}(T^2 \cdot N \cdot M)$
EPIC	$\mathcal{O}(N \cdot T + M)$	$\mathcal{O}(T \cdot N + T \cdot \log(T))$
UMEP	$\mathcal{O}(N \cdot T)$	$\mathcal{O}(T \cdot N + T \cdot \log(T))$
MDEP	$\mathcal{O}(N \cdot T + M)$	$\mathcal{O}(T \cdot N + T \cdot \log(T))$

experiments over the *SPECTF* dataset are performed. Table 2.10 presents the execution time of several heterogeneous classifiers with initial bagging of 25, 51, 75, 101, 151, and 201. The values of the remaining parameters are $M = 2$, $N_{train} = 314$, $D_{pr} = N_{train}$, $P = 20\%$. The times are averaged over 100 executions in Intel(R) Core(TM) i5-7200 CPU @ 2.50GHZ with 8 GB of RAM.

Table 2.10: Average execution time in seconds for the *SPECTF* dataset.

Metric \ T	25	51	75	101	151	201
RE	0.09	0.26	0.78	1.48	3.95	5.91
CC	0.06	0.26	0.68	1.20	2.85	4.87
MDSQ	0.05	0.19	0.32	0.48	1.13	2.96
MRMR	0.10	0.91	1.10	2.83	4.09	7.66
DISC	0.10	0.58	1.43	2.17	3.64	7.01
EPIC	0.01	0.03	0.04	0.05	0.09	0.11
UMEP	0.01	0.02	0.03	0.04	0.07	0.10
MDEP	0.02	0.04	0.05	0.07	0.12	0.15

The results show that UMEP is the fastest method because the rank is calculated based on the classifier's correct classified samples from the pruning set D_{pr}. Additionally, the computation time of EPIC and MDEP is also rather fast with an increasing complexity approximately log-linear with respect to T. While, the execution time of RE, CC, MDSQ, MRMR, and DISC is quadratic in T. Both MRMR and DISC are much slower, as the selection of one more classifier is conditioned by more internal calculations.

Besides the efficacy of the ensemble pruning metrics, other main benefits concern the efficiency as; storage requirements and the classification speed. The booked memory space for complete bagging will be released to store the pruned subensemble. While the classification speed depends on both (1) the size of the pruned ensemble and (2) the complexity of the base classifiers.

2.3.9 Exploration of the Research Questions

In this part, we determine whether the research questions of the third objective are answered or not.

(Q_5) *The initial classifier pool size and the required subensemble size* affect the performance of the pruning metrics. In general, as the pool size T increases the possibility of finding better subensembles will increase. However, this adds more computational complexity to the pruning process. In addition, it is so difficult to identify a fixed subensemble size P for each pool size T, upon which a higher predictive performance can be guaranteed.

(Q_6) *The pruning metrics can be affected by the individual classifier type*. Different classifier types produce different decisions and affect the majority voting from the pool. Experimentally, the predictive performance of the identified subensemble, via the pruning metrics, is not static and can be affected based on what we use as similar or different classifier types.

(Q_7) *The efficacy of the pruning metrics could be affected by binary and multiclass datasets*. Experimentally, the rank of the pruning metrics differs upon the classification task. For the investigated binary datasets, the best ranks were scored by MDEP, DISC, EPIC, RE, and UMEP. In contrast, for the investigated multiclass datasets, the best ranks were scored by DISC, EPIC, UMEP, CC, and MRMR. Our explanation for that, the complexity of an individual's decision will increase with the number of classes, and different decisions will be encountered. Regarding that, the capability of each pruning metric could be different.

(Q_8) *The pruning metrics are effective to reduce the performance variance*. Experimentally, the pruning metrics are capable of selecting non-random subensembles. This was confirmed via the distribution of the predictive accuracy of the pruned ensembles around the median value, to be better than original ensembles.

(Q_9) *The efficiency of the pruning metrics differs in terms of time and space complexities*. For space complexity, MDSQ and UMEP require less memory space to operate. For time complexity, three metrics have the lowest computational cost which is linear with T (UMEP, EPIC, and MDEP). In contrast, the other five metrics (RE, CC, MDSQ, MRMR, DISC) perform at a larger computational cost which is quadratic in T.

2.4 Conclusions

Multiple classifier systems are superior to any random single classifier. However, three main defects are reported for those systems; (1) A large pool of classifiers should be built, (2) A Large memory space should be available to store those models, and (3) A large classification time will be consumed to combine multi-decisions. To alleviate these drawbacks, this chapter discussed the concept and the benefits of thinning, pruning, selecting a subset of classifiers. Effective, fast, and implementable heuristic metrics are analyzed to reorder the classifier's position in the generated random bagging. The main conclusions from this study are highlighted as follows:

- The accuracy from ordering metrics is affected by both, the original number of classifiers and the required percentage for the selected subensemble.
- The heuristic metrics with small size bagged ensembles can easily outperform the large size ensembles.
- The performance of heuristic methods can keep on improving regardless of the degradation in the performance of the bagged ensembles.
- The inclusion of different classifiers is more promising to create diversity and complementary in the subensemble than similar classifiers.
- For binary datasets, most of the investigated metrics are significantly outperforming bagging by 95%, while MDEP is the best among of them.
- The best selected single model from a group of classifiers is the worst strategy for selecting subensemble.
- For multiclass datasets, most of the investigated metrics are significantly outperforming bagging by 95%, while DISC and EPIC are the best.

- For the analysis over the 30 datasets, the three metrics DISC, EPIC, and MDEP are still significantly outperforming bagging by 95%.
- The behavior of heuristic metrics against randomness has been analyzed by displaying the prediction accuracies of subensembles around the median.
- Regarding efficiency, UMEP is the fastest heuristic metric to select promising subensemble. While MRMR and DISC are too much slower than other investigated methods due to their internal calculations.
- The general analysis over the 30 datasets proves that ordering metrics are comparable with Random Forest and Adaboost according to accuracy, but they are better regarding the ensemble size.

In summary, ordering based pruning metrics are more effective and efficient to outperform bagging ensembles. The subensemble that minimizes/maximizes a predefined generalization performance criterion is located easily. Analysis of removing the similarity, as in MRMR, makes the usual aggregation of majority voting no longer favorable since the majority has now been significantly reduced. Among the heuristic metrics, UMEP, EPIC, and MDEP have the lowest computational cost which is linear with the size of the initial pool of classifiers. The other investigated metrics are good, but with larger computational cost (quadratic in T).

Regarding future research, the concept of uncorrelated error is so important to alleviate the consolidated error. Regarding that, the subensemble could be boosted if the pruning metrics incorporate the uncorrelated error

Everything is theoretically impossible, until it is done.

Robert A. Heinlein

CHAPTER

3

Conclusions and Future Work

In this chapter, we summarize the main contributions, limitations, and future lines of research resulted from this thesis. First, a consolidated view of results and contributions is presented in Section 3.1. Afterward, the limitations of this work are identified in Section 3.2. Finally, the future research lines from this thesis are exposed in Section 3.3.

3.1 Summary of Contributions

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

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

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

The research developed in this dissertation has contributed to the enhancement of MCS. The experiments have been conducted on popular benchmark datasets. In chapter **??**, we presented a nice taxonomy for MCS. Afterward, we have discussed the importance of diversity and how we can measure it through pairwise and non-pairwise metrics. Furthermore, the concept of error diversity is presented to alleviate the consolidated error. Finally, the importance of soft computing is highlighted within the area of MCS. In chapter 1, we have reviewed the state-of-the-art classifier ensembles, this included; bagging-like ensembles, boosting, and gradient boosting ensembles. The presented algorithms were compared together in terms of how to promote diversity, and the type of the base model. Afterward, due to the complexity of MCS, metaheuristic algorithms (MA) were specifically applied to better integrate or prune the classifiers set. Finally, from the conducted revision, we identified the gaps and the research questions that we answered in this thesis. Among the chief contributions of this thesis:

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

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

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

classifier set from each metric is merged together to form a new subensemble to be further searched via metaheuristic methods. The proposal successfully solved the parameter tuning challenges, the alpha parameter in Equation **??**, which is part of a novel and recent pruning metric, MDEP [6]. The limited accuracy of MCS pruning metrics has been maximized via the proposed guided search-based pruning. This has been proved and clarified as in Figure **??**, and according to the statistical analysis, Section **??**. In this work, small-size ensembles with training on fewer samples could outperform significantly the large-size ensembles which use the whole available training data. Concluding, the predetermined second objective is totally achieved.

Chapter 2, is dedicated to the third objective: *"Grouping and analyzing fast and accurate heuristic metrics for MCS pruning"*. Ensemble pruning strategies are one of the hot topics to gain efficient and effective ensembles. The efficiency could be reached via forming small-size ensembles with their impact; to consume short memory space, reduce the communication cost of the distributed models, and to accelerate the prediction time. While the efficacy could be achieved via building trustable models with a high level of prediction accuracy. We applied a fast and accurate strategy to work with bagging ensembles via ranking the ensemble members. This pruning strategy is known as "ordering-based pruning" to identify the best subset of classifiers for early aggregation. The identification of this subset is mainly controlled through a heuristic measurement to optimize the augmented subensemble. In this chapter, greedy search methods and group-based ranking have been considered to reorder the pool of classifiers. Since the great analysis by Martínez-Muñoz et al. [4] in 2009, several heuristic metrics belonging to this category were proposed [1, 5–7] with no existence comparison between them. Regarding that, our proposal was conducted to solve that gap. The conclusions from this analysis proved that the efficacy of those metrics is affected by the original ensemble size, the required subensemble size, the kind of individual classifiers, and the number of classes. Furthermore, the investigated metrics realize robust and stable predictions, this is analyzed via the range of their prediction accuracy around the median as shown in Figure 2.3. Finally, the computational cost of some metrics is linear with the initial pool size T, while other metrics have a larger computational cost, which is quadratic in T. Concluding, the predetermined third objective is totally achieved.

3.2 Limitations

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

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

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

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

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

3.3 Future works

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

1. The proposed MCS is heterogeneous, containing different models, with the aim to promote diversity. While the identification of classifiers to be included in classifier ensembles remains a key issue for predictive performance. In [20], the classifiers types to be consolidated are selected manually via the majority voting error and forward search. Experimentally, the set of classifiers solving a task could not be effective for another task [21]. The inclusion of weak classifiers could degrade the general performance unless if they create complement decisions for particular samples, this is too complex and need to be further discovered and analyzed via tailored metrics to determine the individual classifier type.

2. In Chapters **??** and **??**, AllKNN [22], as a training set selection, has been applied due to its reasonable selection time, reduction rate, and limited-accuracy over test [23]. An interesting research line would be to evaluate the performance of other training set selection algorithms. Particularly, how each reduction method could add to the performance of MCS. In addition, the concept of instance selection had been widely used with the support vector machine classifier [24, 25] to alleviate its computational difficulties when dealing with huge amounts of data. In connection with that, a Pareto-based SVM ensemble has been proposed in [26] to optimize the size of the training set and the classification performance attained by the selection of the instances. Thus, the consideration of those modeling approaches could potentially lead to new promising versions for both homogeneous and heterogeneous MCS.

3. In Chapter **??**, We have identified that both the classifier's accuracy and the ensemble diversity are crucial for MCS pruning. Our proposed schema is so simple and innovative. The computational cost of our solution is light, due to the usage of ordering metrics, in comparison to evolutionary algorithms. While multi-objective optimization could be further applied to tune the calibration between the subensemble size and the subensemble accuracy. Examples of recent works in this area [27].

4. The concept of uncorrelated error, Section **??**, is so important to alleviate the consolidated error. Regarding that, the weights of Equation (**??**) can be optimized by evolutionary algorithms. Furthermore, the subensemble could be better located if the concept of uncorrelated error can be incorporated in the investigated metrics of Chapter 2.

5. In Chapter **??**, the SI-based weighted voting schema considers the abstract predictions from the individual classifiers. In contrast, the class probability distribution carries information that should not be ignored [28]. For example, if we take the votes according to the class labels, then the incorrect outputs of the mistaken members will be amplified into wrong votes. A future research line could be interesting to apply SI for the class probability distribution. Furthermore, transforming the outputs of ensemble members into class labels before considering their soft votes, leads to destroying the real distribution and losing useful information [29]. Besides, the usage of soft voting could be more efficient for ensemble pruning via decreasing the probability that a tie occurs, i.e in majority voting.

6. In [30], the authors recommended using the classification confidence and the weights of the base classifiers to define the ensemble margin. They used a homogeneous ensemble of SVM, and the classification confidence for each classifier is calculated in terms of the distance between sample $\mathbf{x}_i$ and the hyperplane. However, it is not clear how to measure the classification confidence for different classifier types, an interesting research line could be to extend their proposal to adapt to a heterogeneous ensemble. In addition, in the same article, the weighted voting of the pruned ensemble proved superior results, in terms of accuracy, over the majority voting of the pruned ensemble. Regarding that, the investigated metrics of Chapters **??** and 2 could be further enhanced.

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

Bibliography

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

[2] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[3] G. Martınez-Munoz and A. Suárez, "Aggregation ordering in bagging," in *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*. Citeseer, 2004, pp. 258–263.

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

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

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

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

[8] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: many could be better than all," *Artificial intelligence*, vol. 137, no. 1-2, pp. 239–263, 2002.

[9] R. Diao, F. Chao, T. Peng, N. Snooke, and Q. Shen, "Feature selection inspired classifier ensemble reduction," *IEEE transactions on cybernetics*, vol. 44, no. 8, pp. 1259–1268, 2013.

[10] D. D. Margineantu and T. G. Dietterich, "Pruning adaptive boosting," in *ICML*, vol. 97. Citeseer, 1997, pp. 211–218.

[11] C. O. Sakar, O. Kursun, and F. Gurgen, "A feature selection method based on kernel canonical correlation analysis and the minimum redundancy–maximum relevance filter method," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3432–3437, 2012.

[12] A. Unler, A. Murat, and R. B. Chinnam, "mr2pso: A maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification," *Information Sciences*, vol. 181, no. 20, pp. 4625–4641, 2011.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[15] J. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[16] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the american statistical association*, vol. 32, no. 200, pp. 675–701, 1937.

[17] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

[18] S. Garcia and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons," *Journal of machine learning research*, vol. 9, no. Dec, pp. 2677–2694, 2008.

[19] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.

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

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

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

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

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

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

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

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

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

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

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