Dynamic Ensemble Selection using Fuzzy Min-Max Hyperboxes

by

Reza Davtalab

MANUSCRIPT-BASED THESIS PRESENTED TO
ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR A MASTER'S DEGREE WITH THESIS
IN SOFTWARE ENGINEERING
M.A.Sc.

MONTREAL, NOVEMBER 12, 2023

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Rafael Menelau Cruz, Thesis supervisor
Department of Software and IT Engineering, École de technologie supérieure

Mr. Robert Sabourin, Thesis Co-Supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Alessandro Lameiras Koerich, Chair, Board of Examiners
Department of Software and IT Engineering, École de technologie supérieure

Mr. Jose Dolz, Member of the Jury
Department of Software and IT Engineering, École de technologie supérieure

Mr. Jean Paul Barddal, External Examiner
Graduate Program in Informatics Pontifical Catholic University

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON NOVEMBER 7, 2023

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I would like to thank my wife Esmat, for her support and encouragement without which this journey would not have ended.

Thanks also to my supervisor Rafael Menelau-Cruz and my co-supervisor Robert Sabourin for their insightful efforts and feedback during this project.

Sélection d'ensemble dynamique à l'aide d'hyperboîtes floues

Reza Davtalab

RÉSUMÉ

Les systèmes de sélection dynamique sont une bonne alternative pour atteindre une précision accrue dans les problèmes complexes. Dans les techniques actuelles de sélection dynamique, la compétence des classificateurs de base pour classifier le nouvel échantillon de requête est estimée en fonction de leurs performances dans une petite région entourant l'échantillon de requête. La plupart de ces techniques utilisent l'algorithme KNN pour estimer la compétence. Cependant, l'algorithme KNN souffre d'une complexité élevée du système et est également sensible à la distribution locale des données.

Dans ce projet, nous allons introduire un nouveau cadre de sélection dynamique d'ensemble qui utilise des hyperboîtes floues pour estimer la compétence des classificateurs de base. Pour la construction des hyperboîtes, la distribution des échantillons est prise en compte et plusieurs échantillons proches les uns des autres sont représentés par une hyperboîte. Par conséquent, le système final ne sera pas sensible à la distribution locale déséquilibrée des échantillons. De plus, le système n'a besoin de conserver que les hyperboîtes plutôt que les données d'origine. De plus, les hyperboîtes sont créées à partir d'échantillons mal classés. Ainsi, ils sont généralement beaucoup moins nombreux que les échantillons. Nous nous attendons donc à ce que le cadre proposé ait une complexité beaucoup plus faible et une précision plus élevée par rapport aux techniques basées sur KNN. Les résultats expérimentaux montrent que le cadre proposé peut améliorer les performances des systèmes de sélection dynamique à la fois en termes de précision et de complexité de calcul.

Mots-clés: Ensemble de classifieurs, Sélection d'ensemble dynamique, hyperboîtes floues, échantillons mal classés, Compétence du classifieur

Dynamic Ensemble Selection using Fuzzy Min-Max Hyperboxes

Reza Davtalab

ABSTRACT

Dynamic Selection systems are a good alternative to achieve higher accuracy in complex problems. In current DS techniques, the competence of base classifiers to classify the new query sample is estimated with regard to their performance in a small region surrounding the query sample. Most of these techniques use the KNN algorithm to estimate competence. However, the KNN algorithm endures a high complexity to the system and is also sensitive to the local data distribution.

In this project, we are going to introduce a novel Dynamic Ensemble Selection framework that uses Fuzzy Hyperboxes to estimate the competence of base classifiers. For the construction of hyperboxes, the distribution of samples is considered and several samples that are close to each other are represented by a hyperbox. Therefore, the final system will not be sensitive to the local imbalance distribution of samples. Besides, the system needs only to keep the hyperboxes rather than the original data. Furthermore, hyperboxes are made based on misclassified samples. Thus, they are usually much fewer than samples. So we expect the proposed framework to have much lower complexity and higher accuracy compared to KNN-based techniques. Experimental results show that the suggested framework can improve the performance of DS systems in both terms of accuracy and computational complexity.

Keywords: Ensemble of Classifiers, Dynamic Ensemble Selection, Fuzzy Hyperboxes, Miss-classified samples, Classifier competence

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DS	Dynamic Selection
DCS	Dynamic Classifier Selection
DES	Dynamic Ensemble Selection
DSEL	Dynamic Selection Data
RoC	Region on Competence
EoC	Ensemble of Classifiers

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

c_i	an individual (base) classifier
C	the pool consisting of M base classifiers including $\{c_1, c_2, ..., c_M\}$
$\mathbf{x_q}$	a test (query) sample with an unknown class label
$\mathbf{x_t}$	a sample from DSEL set with a known class label
e_i	number of hyperboxes in DS system which belong to c_i
S_i^+	a subset of DSEL correctly classified by c_i
S_i^-	a subset of DSEL samples misclassified by c_i
η_q	the region of competence of $\mathbf{x_q}$ including $\{\mathbf{x_1}, ..., \mathbf{x_K}\}$
$\mathbf{x_k}$	one instance belonging to η_q
$P(\omega_l \vert \mathbf{x_q}, c_i)$	Posterior probability obtained by the classifier c_i for the instance $\mathbf{x_q}$
$\delta_i(\mathbf{x_q})$	the estimated competence of the base classifier c_i for the classification of $\mathbf{x_q}$
Ω	the set of L class labels $\{\omega_1, ..., \omega_L\}$
ω_i	the class predicted by c_i
k	the number of instances in the region of competence
$\phi(\mathbf{x_q})$	the ensemble of selected classifiers to classify $\mathbf{x_q}$
$\delta_{i,q}$	Estimated competence of classifier c_i for $\mathbf{x_q}$
b_j	the j-th hyperbox in the DES system
$m_j(\mathbf{x})$	the membership level of the hyperbox b_j for the sample $\mathbf{x}$
H_i^+	Positive hyperboxes of c_i

H_i^-	Negative hyperboxes of c_i
$\mathbf{v_j}$	the minimum point of the hyperbox b_j
v_{jd}	the minimum point of the hyperbox b_j in dimension d
$\mathbf{w_j}$	the maximum point of the hyperbox b_j
w_{jd}	the maximum point of the hyperbox b_j in dimension d
$\mathbf{o_j}$	the centroid of the hyperbox b_j
θ	represents the maximum size of hyperbox
μ	a predefined threshold to select the best classifiers
λ	learning sensitivity parameter
b_j	the j-th hyperbox in the DES system

INTRODUCTION

Nowadays, we are faced with a large amount of data in different areas such as advertising (Garcia *et al.* (2018)), education (Mahajan & Saini (2020)), healthcare (Dash, Shakyawar, Sharma & Kaushik (2019)), economy (Modgil, Gupta, Sivarajah & Bhushan (2021)), and entertainment (Hallur, Prabhu & Aslekar (2021)). A big part of this data is directly related to humans and generated by them. In every interaction with technology, we are creating new data that can describe us such as captured data by video cameras, credit cards, cell phones, GPS devices, and other touchpoints, our data profile is growing exponentially. According to reported statistics in 2020, an average of 4 petabytes of data were created on Facebook every day, Facebook users also click the like button on more than 4 million posts every minute, 65 billion messages are sent on WhatsApp, 500 million tweets are sent by Twitter and 5 billion searches are made [1].

Additionally, in most modern systems, we see the creation of streaming data that produces new instances at any given time. This data is usually on a large-scale and each data point should be processed online and is not available to process more in the following steps (Porto & Gomide (2022)). So we should find a way to store the information captured from the previous data points as much as possible. Thus, we have to develop good algorithms and techniques capable of dealing with this amount of data. These algorithms must be able to handle complex, large-scale, and stream data to extract valuable information.

Multiple Classifier System (MCS) is a solution to address these types of classification problems (Cruz, Sabourin & Cavalcanti (2018a); Zyblewski, Sabourin & Woźniak (2021)) because they can learn to capture multiple characteristics of data (Dong, Yu, Cao, Shi & Ma (2020)). MCS uses multiple classifiers to compensate for each other's weaknesses (Kuncheva (2014)). These systems have been used in various pattern recognition applications in recent years (Nozari,

[1] https://blog.microfocus.com/how-much-data-is-created-on-the-internet-each-day/

Nazeri, Banadaki & Castaldi (2018); Vijayanand, Devaraj & Kannapiran (2018); El-Melegy & El-Magd (2019); Kalid, Ng, Tong & Khor (2020)). The reason for this tendency is the need for more precision and efficiency, especially in cases where we face complex applications such as data streams and concept drift (Jiao, Guo, Gong & Chen (2022)), handling noise (Walmsley, Cavalcanti, Sabourin & Cruz (2022)), imbalanced data (Wang, Zhang & Yan (2023)).

Furthermore, recent works demonstrated that Dynamic Selection (DS) could be a better choice for the combination of classifiers (Britto Jr, Sabourin & Oliveira (2014); Cruz *et al.* (2018a)). In these systems, each given query sample is labeled by a subset of base classifiers from the original pool of classifiers, which are usually selected with regard to their local competence. Estimation of competence level is a key issue in DS approaches. In order to achieve that, they employ different criteria to measure the local competence of classifiers, such as accuracy (Soares, Santana, Canuto & de Souto (2006)), probabilities (Woloszynski, Kurzynski, Podsiadlo & Stachowiak (2012)), ranking (Sabourin, Mitiche, Thomas & Nagy (1993)) and fuzzy rules (Elmi & Eftekhari (2020)), in a small region surrounding the query instance (Cruz *et al.* (2018a)). In these approaches, the local region in which the base competencies are estimated is defined by K-Nearest Neighbor (KNN) technique (Cruz, Sabourin, Cavalcanti & Ren (2015c); Fernández-Delgado, Cernadas, Barro & Amorim (2014); Xiao, Xiao & Wang (2016); Krawczyk, Galar, Woźniak, Bustince & Herrera (2018); Cruz *et al.* (2018a); Cruz, Souza, Sabourin & Cavalcanti (2019b)), while other methods utilize some techniques such as clustering (Lin *et al.* (2014)), potential functions (Woloszynski & Kurzynski (2009); Woloszynski *et al.* (2012)) or decision space (Giacinto & Roli (2001); Cavalin (2012); Batista, Granger & Sabourin (2012); Nguyen, Luong, Dang, Liew & McCall (2020)).

In dynamic selection approaches, there is a pool of classifiers (C). For each given unknown sample (which is called query sample or $\mathbf{x_q}$ in the rest of this document), a single classifier c_i or a subset of classifiers ($C' \subseteq C$) is selected specifically to classify this query sample.

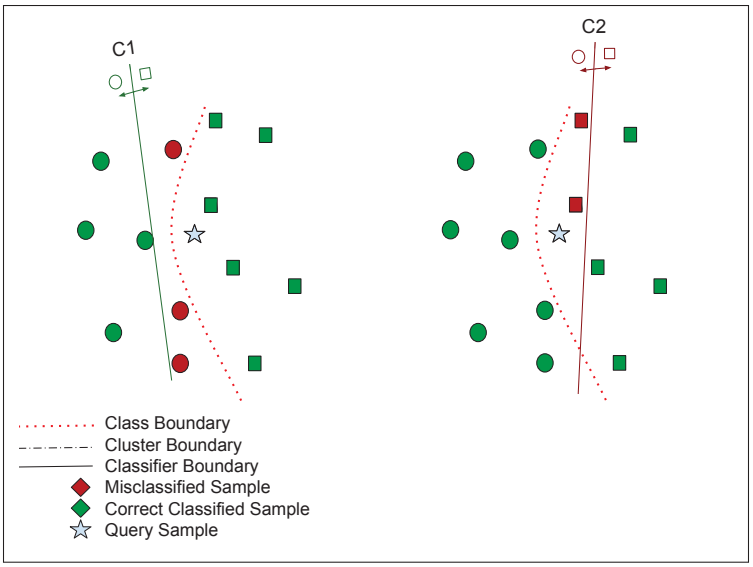


Figure 0.1 Example of a two-class problem which will be solved using two classifiers

These techniques involve identifying the subset of classifiers with the highest competence for classification of $\mathbf{x_q}$. To estimate the competence of classifiers, we need a set of labeled samples. This set can be either the training or validation set and is called the dynamic selection dataset (DSEL). To illustrate how it works, suppose there is a classification problem with two circle and square classes (Figure 0.1). We know the true class boundary. In this example, the class boundary is shown by the red dashed line, and the predicted class boundaries are shown by straight lines. All the correct classified samples by the related classifiers are highlighted in green, while the misclassified samples are highlighted in red.

0.1 Problem Statement

Dynamic Selection system should choose the classifier able to classify an unknown instance or the classifier that is competent for labeling the unknown instance. In this situation, we can use a potential function approach like (Woloszynski & Kurzynski (2009); Woloszynski *et al.* (2012)) to select the best classifier. In these approaches, the competence level of classifiers is calculated

regarding all instances of DSEL (Cruz *et al.* (2018a)). Thus, their complexity is linearly related to the DSEL size. Therefore, these approaches suffer from high computational complexity and are not a good choice for large-scale problems. Figure 0.2 shows that all the DSEL instances are engaged in this process. This process is done for each classifier separately to classify each query sample, thus we will have an extreme computational complexity.

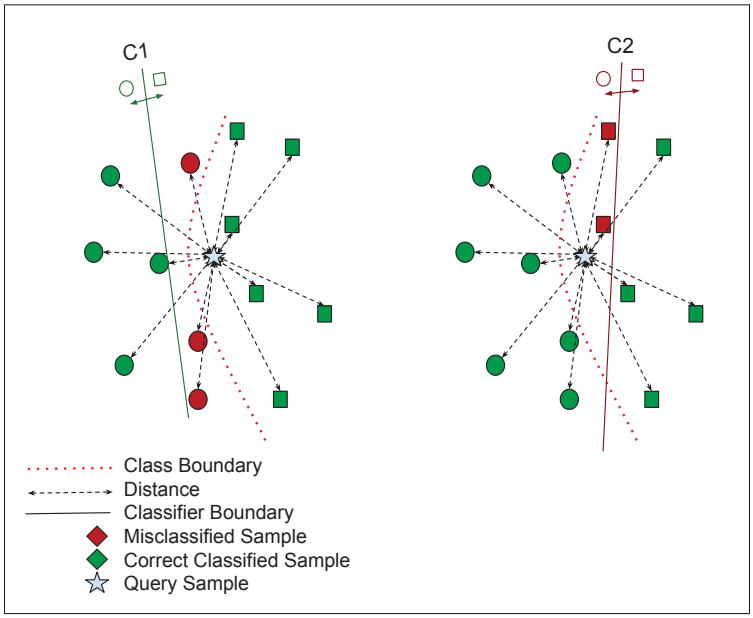


Figure 0.2 Considering all samples to select a more competent classifier in DS technique based on Potential Function

To solve the mentioned problem, we can use clustering-based DS methods (Lin *et al.* (2014)) which have much lower computational complexity in the generalization phase. In these approaches, the DSEL data are clustered during the training phase and the performance of all base classifiers is estimated on each cluster. In the generalization phase, the most competent classifiers on the nearest cluster are selected as the final ensemble. In other words, in the generalization phase, it is just needed to calculate the distance between the query sample and cluster centroids. Therefore, significantly reducing computational complexity. Figure 0.3

illustrates how the clustering approach estimates the classifiers' competence in the previous example (Figure 0.1).

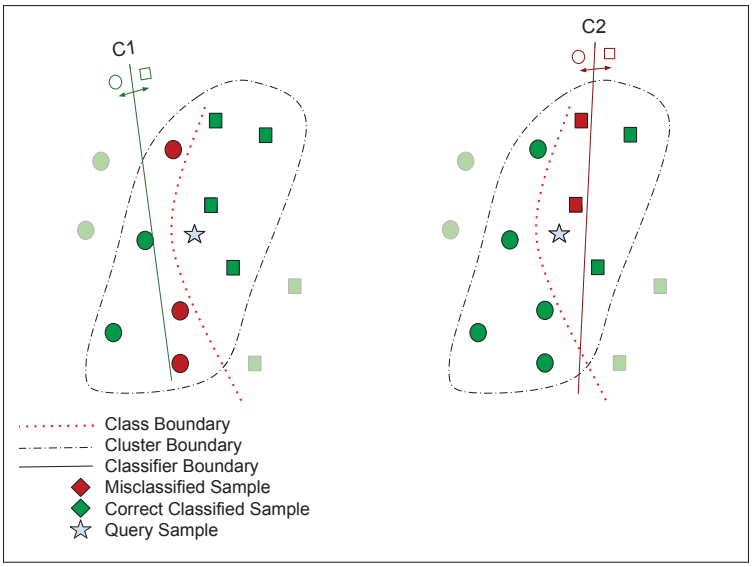


Figure 0.3 Simple example of dynamic selection approaches

Cluster-based approaches are much faster during inference because the system only requires to compute which cluster is closer to the query. Nevertheless, their accuracy falls behind approaches based on KNN (Cruz *et al.* (2018a)). In addition, the performance of these systems is highly dependent on the number of clusters determined by the user. The system is at risk of underfitting and losing information if the number of clusters is too small, losing the fine-grained information from the local region around the given query.

Therefore, most state-of-the-art DS approaches are KNN-based, which is more accurate than clustering approaches. KNN-based approaches also have less complexity rather than potential function approaches. In KNN-based techniques, the competence of base classifiers is estimated according to the efficiency of these classifiers on K nearest neighbors of the given query sample in the DSEL data, which is called Region of Competence (RoC). These techniques use different criteria to estimate the competence of classifiers such as ranking (Sabourin *et al.* (1993)),

accuracy (Woods, Kegelmeyer & Bowyer (1997)), probabilistic (Woloszynski & Kurzynski (2011); Woloszynski *et al.* (2012)), behavior (Cavalin, Sabourin & Suen (2013)), and meta-learning (Cruz, Sabourin, Cavalcanti & Ren (2015d); Cruz, Sabourin & Cavalcanti (2017a)). Despite achieving better performance compared to DS based on clustering, KNN-based methods present several limitations:

0.1.1 K value problem

In these approaches, finding the K nearest neighbors for each query sample is a high computational cost process. In addition, even the optimized K value may not work correctly in all regions. For example, as shown in Figure 0.4, two classifiers c_1 and c_2 are available to classify the query sample (star point). The goal is to select the most locally competent classifier. If we set K=5 (larger circle), c_2 will be selected as the most competent classifier because c_2 is able to classify 4 out of 5 samples correctly in this region. However c_2 is not able to classify the query sample correctly, and it leads to the wrong classification of the query sample. While in the scenario where K equals 3 (Small circle), KNN will select c_1 which could classify all three samples correctly in the local region. In this scenario, the selected classifier (c_1) is able to classify the query sample correctly.

0.1.2 Sensitivity to unbalanced Distribution

KNN works based on the distance between points and it has a large sensitivity to the local distribution of data and a high degree of overlap in data may lead to low accuracy (Cruz, Sabourin & Cavalcanti (2018b)). These approaches also are not robust against noisy data (Elmi & Eftekhari (2020)). Even KNN-based approaches can select classifiers that classify all samples in the RoC as being from the same class (Oliveira, Cavalcanti & Sabourin (2017); Cruz, Oliveira, Cavalcanti & Sabourin (2019a)). Thus, in this condition, KNN-based approaches could not perform well, especially in the case that we have a high-density distribution of samples near

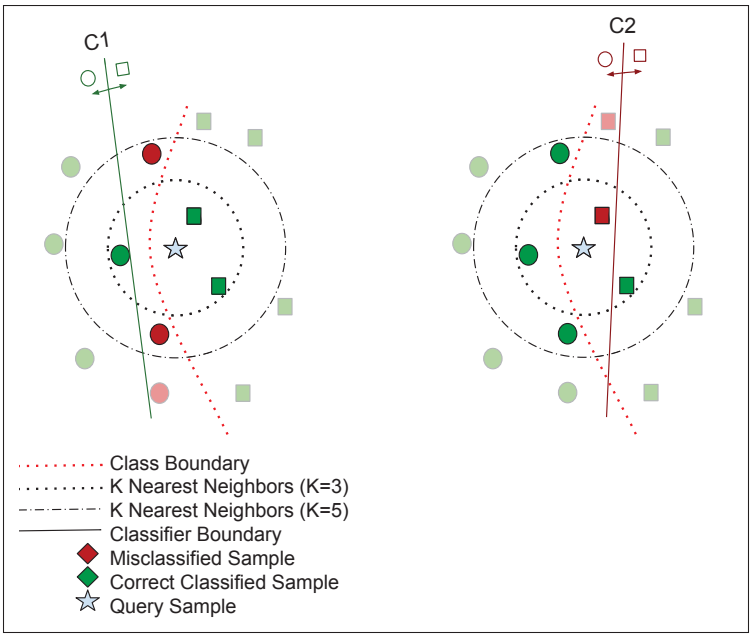


Figure 0.4 K value problem in KNN-based DS approaches. c_1 accurately classifies the query sample (star point), whereas c_2 fails to do so. c_2 is selected as the most competent classifier by KNN when K=5 because its accuracy ($\frac{4}{5}$) is higher in the local region (large circle). While if K equals 3 (Small circle), KNN will select c_1 which is a better classifier in this example

the class boundary than the other side. It could lead to wrong decisions surrounding this area. In Figure 0.5 this issue is represented.

As shown in this figure, classifier c_2 could successfully classify 4 out of 5 samples in the local region so the KNN algorithm selects this classifier as the most competent classifier. However, c_2 classifies the query sample in the wrong way. In general, the KNN algorithm has serious challenges in low-density regions and local unbalanced distribution these issues in regions close to boundaries cause very destructive effects.

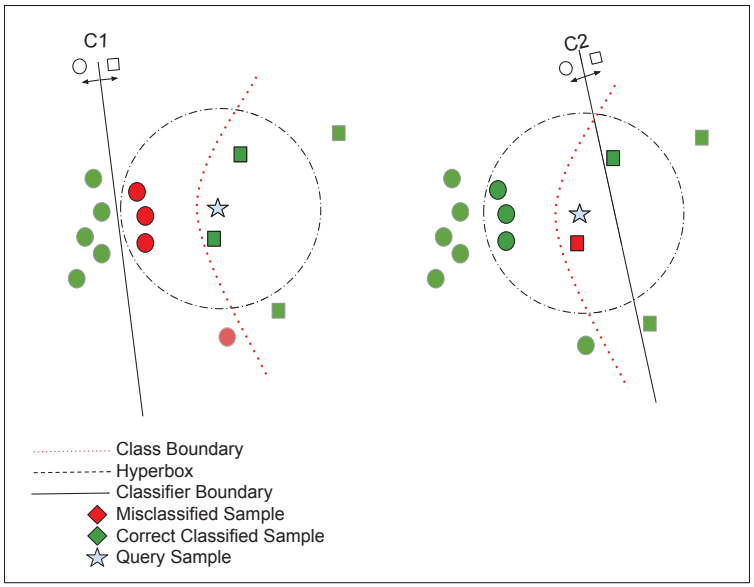


Figure 0.5 Local Distribution problem in KNN-based DS approaches. c_1 is able to classify the query sample correctly but KNN selects c_2 as the competent classifier

0.1.3 Limited Information

The next problem is using a small amount of available information to make decisions. KNN just considers the information of RoC which is a small region of feature space. Thus, in many conditions, the performance of two classifiers is the same inside the RoC, we cannot realize which one is more competent, and this issue could lead to wrong decisions. Therefore, we need more information to select the most competent classifier. Figure 0.6 shows how ignoring information around the selected region could lead to making a wrong decision.

In this example, classifier c_1 is a better classifier to classify the query sample (star point). However, both classifiers c_1 and c_2 have the same performance in the local region (k=5). Therefore KNN algorithm is not able to realize which classifier is the better choice to classify the query sample.

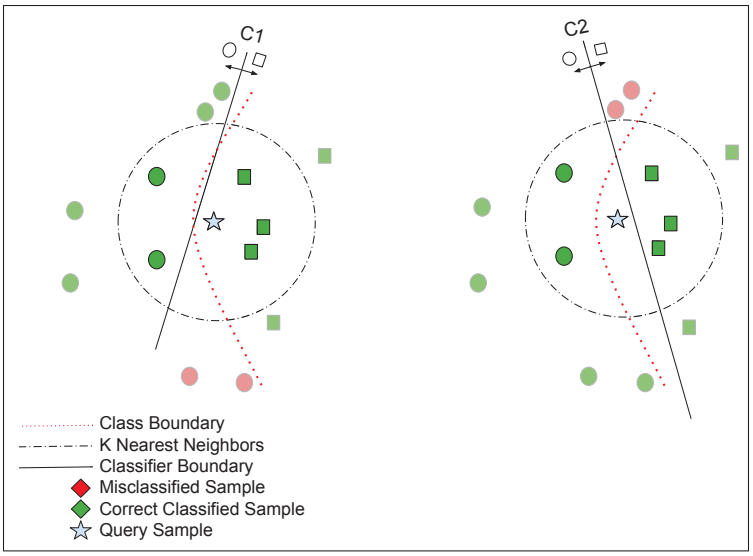


Figure 0.6 Ignoring extra information in KNN-based DS approaches

To better represent the mentioned problems, we designed a simple experiment. Some popular KNN-based DS approaches such as KNORA-E, KNORA-U, OLA, DES-KNN, and META-DES techniques (with k=7) were utilized to solve P2 (Cruz, Sabourin & Cavalcanti (2015b)) problem which is a 2-dimensional classification problem.

To fit these models, 1000 generated instances were used (which are shown in Figure 0.7) and two classifiers were utilized. The classifiers were designed in such a way that classifier 1 labeled all samples as class A ($\omega_1 = A$); on the contrary, classifier 2 labeled all samples as class B ($\omega_2 = B$). It means, that for each given unknown sample, there is a classifier that classifies the given sample correctly. In this example, 48 common misclassified samples have been found, which are represented by red star markers in Figure 0.7.

As shown in this figure, in some areas marked by blue rectangles, the errors surely occurred because of the imbalance in local distribution. In these areas, we have a dense distribution of instances close to the class boundaries. In addition, in areas marked by purple rectangles, we have some misclassified samples because of the K-value problem and limited information.

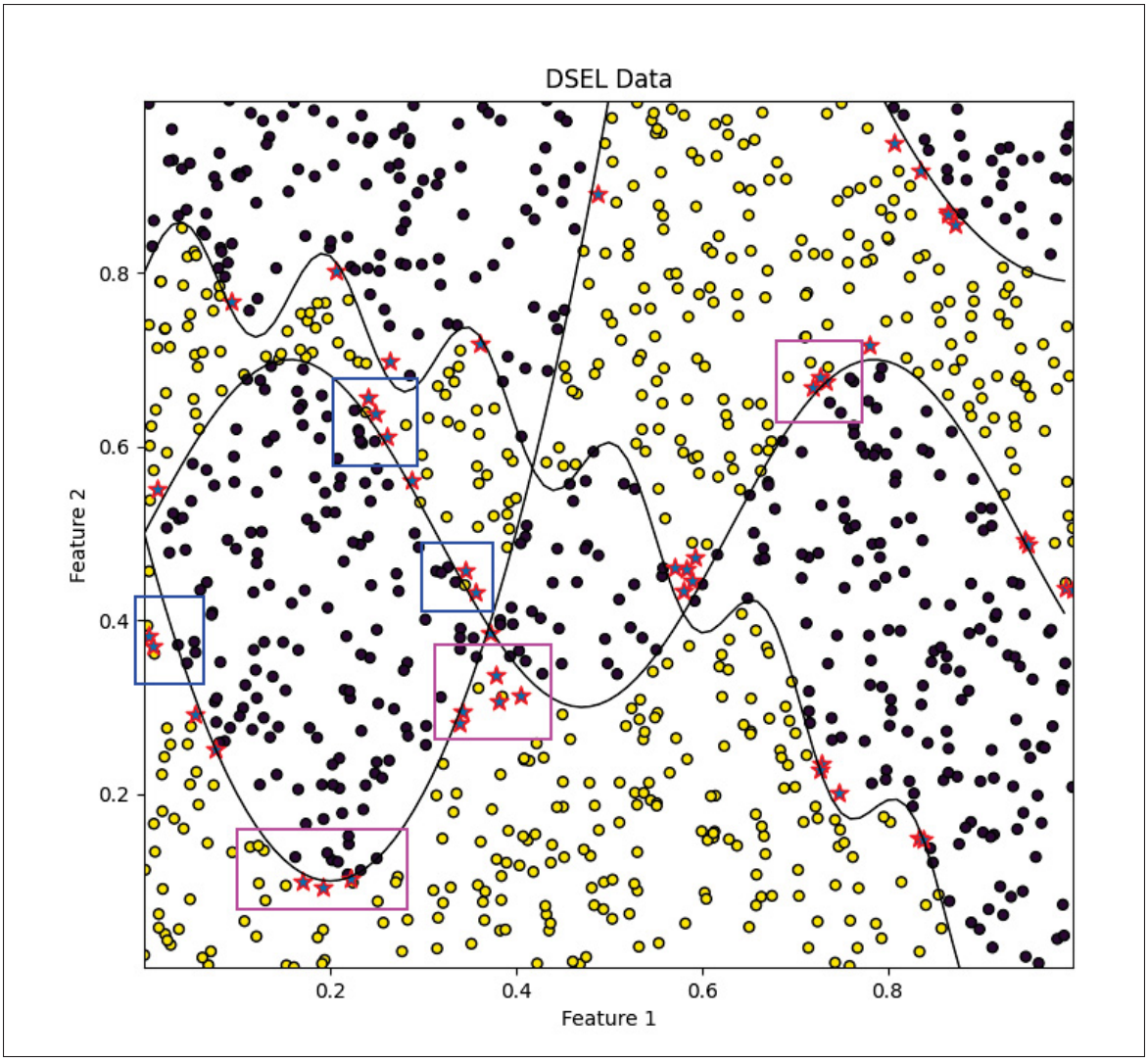


Figure 0.7 Common misclassified samples of P2 dataset with KNN-based DS approaches

To tackle these problems, some techniques are proposed that use a meta learner to select the best ensemble of base classifiers (Nguyen *et al.* (2019); Cruz *et al.* (2017a, 2015c)). Despite using meta-learners to estimate the competence level of classifiers, determining a proper set of features to feed into meta-learner and training such meta learner is not easy. In addition, most of the selected features are formed based on KNN algorithms which have the sample problem that was mentioned earlier.

0.1.4 Computational Complexity

In addition, KNN-based approaches have high complexity and are not efficient in large-scale problems and real-time applications. Labeling of each query sample with current DS techniques includes defining the region of competence (RoC). In this stage, we must find the K-nearest neighbors of the query sample. It means the distance between the given data point and all samples must be calculated, and it endures a huge calculation complexity to the system. So the complexity of processing each query sample would be $O(n)$, which n is the number of samples. Therefore, we need DS systems that are less complex for large-scale applications, especially the systems that need to answer real-time, such as self-driving cars, Air Traffic Control Systems, Command Control Systems, etc (Davis & Cucu-Grosjean (2019)).

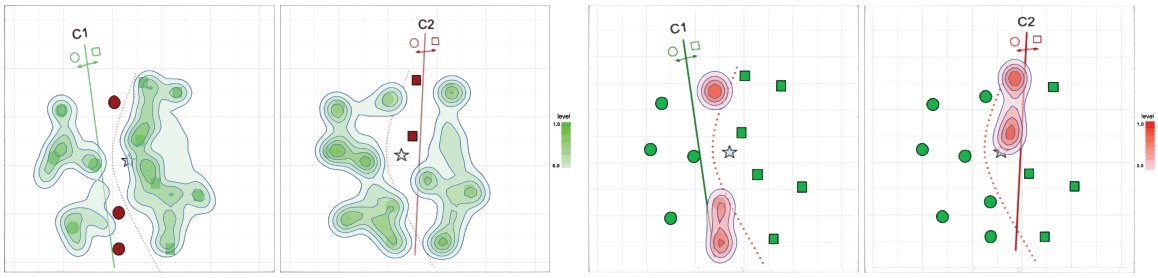
In addition, we need more efficient DS methods to process large-scale data. Worth noting that the term "large-scale" is a relative concept that depends on many factors such as the processing capacity of hardware, processing techniques, the type of problem, etc. However, in this study, we will refer to the problems with more than 100k instances (and any number of features) as large-scale problems.

Therefore, we need a new DS algorithm that can solve such problems accurately and fast.

0.2 Hypothesis

We hypothesize that defining and storing hyperboxes to represent the competence and incompetence of classifiers could speed up the process of labeling in DS systems. On the other hand, we believe that utilizing all available information surrounding the query sample could increase the accuracy. Therefore, in this project, the competence and incompetence area (map) of each classifier is defined and stored during the training phase. This information is used during the generalization phase. In this way, the feature space will be partitioned into competence and

incompetence regions for each classifier. Falling the query sample $\mathbf{x_q}$ into the competence region of the classifier c_i means that this classifier is competent to classify $\mathbf{x_q}$. On the other hand, if the sample falls into an incompetence area of c_i, this classifier is not competent to classify $\mathbf{x_q}$. In Figure 0.8 the ideal condition for solving the example of Figure 0.1 is shown.



a) Competence area of classifiers c_1 and c_2 b) Incompetence area of classifiers c_1 and c_2

Figure 0.8 Ideal solution for the example of Figure 0.1

As shown in Figure 0.8, the competence and incompetence areas are illustrated in green and red, respectively. Therefore in this example, c_1 is more competent in classifying the query sample rather than c_2 because the query sample has fallen in the green area of c_1. The main challenge is determining and storing the confine of these areas. Defining the domain of such areas is not easy and imposes a large computational complexity on the system unless some simpler structures are used to represent these areas. For example, in two-dimensional feature space, we can cover these areas with rectangles. Each rectangle could be defined using only two points. Therefore, its computational complexity will not be high if there is an acceptable number of rectangles.

Hyperbox is a virtual concept that works like these rectangles, the difference is that hyperbox can work in higher dimensional spaces too. Each hyperbox, in addition to its interior space, also covers a small part of its vicinity. As we move away from the hyperbox, the coverage decreases fuzzily. That is why it is called *fuzzy hyperbox* (Simpson (1992)).

The fuzzy aspect of the hyperbox gives us valuable information outside of the hyperbox and we can estimate how far is the query sample from the competence or incompetence area of the base classifier. Thus, we can make a decision even in case the query sample falls outside of all hyperboxes. Here, we use 2-dimensional fuzzy hyperboxes to show how we can use them to solve the problems of current dynamic selection approaches. As you can see in Figure 0.9, the query sample is located outside of all hyperboxes, however, it is located close to a hyperbox of classifier c_1 (inside its green area of c_1). Thus, this classifier is considered more competent than classifier c_2.

In Figure 0.9, some hyperboxes are used to simulate the ideal condition represented in Figure 0.8.

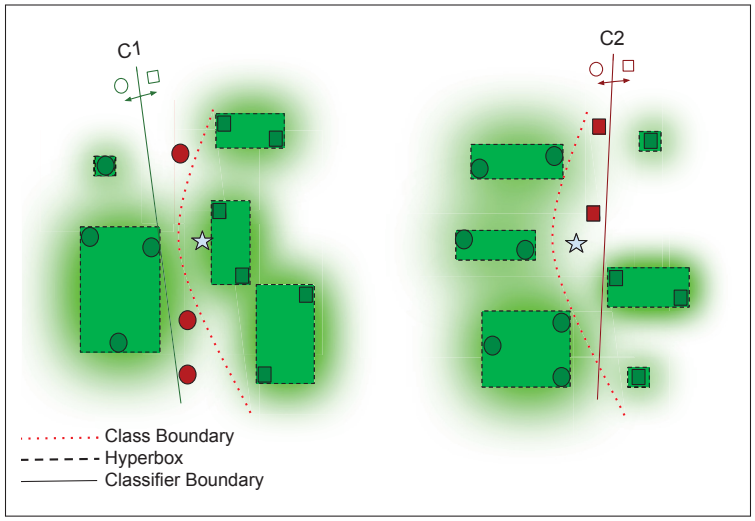


Figure 0.9 Solving the example of Figure 0.1 using competence information stored by Fuzzy Hyperboxes

Hyperboxes can also be an alternative for dealing with local regions that are unbalanced. In Figure 0.10, the sample example of Figure 0.5 could be solved using hyperboxes. As shown in this figure, hyperboxes just stored the location of the instances that are located close to each other, so in this case, the local density of samples could not affect the performance of the system.

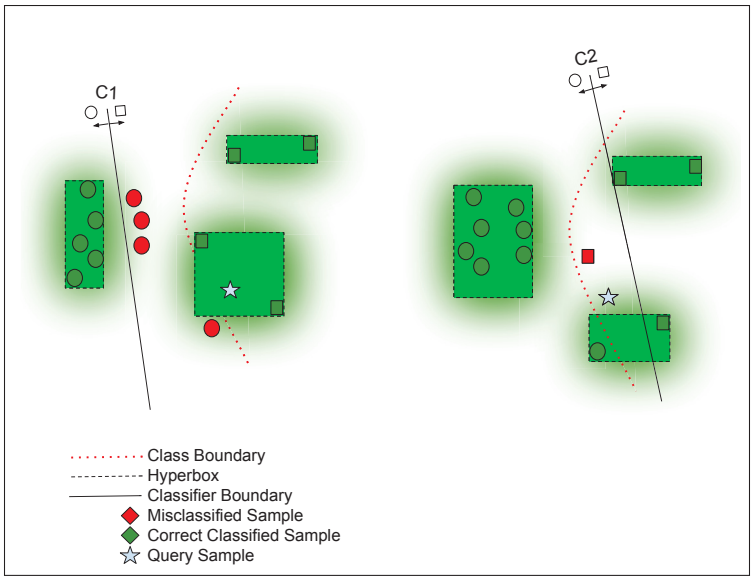


Figure 0.10 Solving the example of Figure 0.5 using competence information stored by Fuzzy Hyperboxes

As you can see, the given query sample is located inside of a hyperbox that belongs to classifier c_1, so this classifier is selected as a competent classifier.

Moreover, using hyperboxes, the information stored by all hyperboxes is available to make the correct decision. Therefore, using hyperboxes more contextual information is leveraged while performing the classifier selection and that will lead to better performance. In Figure 0.11 we used some hyperboxes to represent the incompetence regions of each classifier, as you can see in this figure, the query sample is further away from the incompetence region of classifier c_2 rather than classifier c_1. Thus, c_1 will be considered a more competent classifier than c_2 in this case.

So we can estimate the local competence of classifiers by creating hyperboxes based on their correct-classified or misclassified samples. In addition, hyperbox-based systems hyperbox-based systems offer potential solutions to challenges associated with issues such as the K value, limited information, and imbalanced local distribution of data in DS techniques. Therefore, we hypothesize that **Utilizing fuzzy hyperboxes in dynamic selection systems can increase the accuracy and speed of the system.**

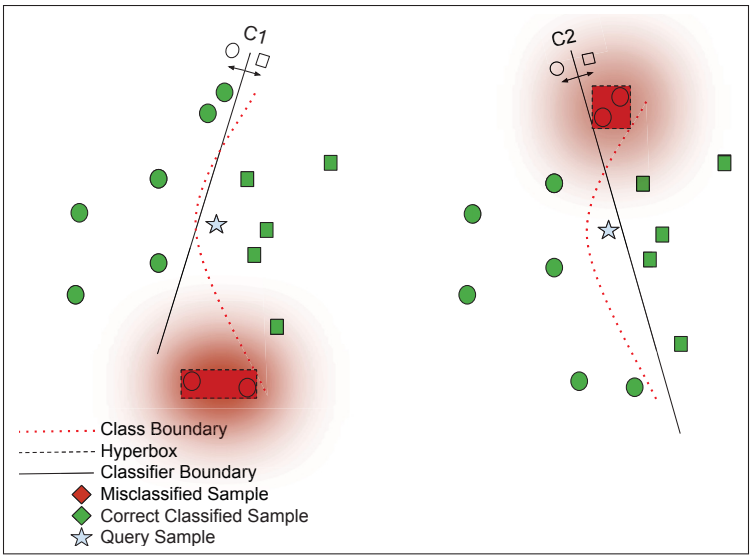


Figure 0.11 Solving the example of Figure 0.6 using incompetence information stored by Fuzzy Hyperboxes

Besides, each hyperbox can represent a group of samples and it does not need to store and process all DSEL instances during the generalization phase. It could significantly reduce memory cost and time complexity in DS systems. Thus, we hypothesize that **using hyperboxes decreases the memory cost in DS systems and makes them faster to deal with large-scale problems.**

Moreover, hyperbox-based approaches possess online learning capabilities with one pass through the data. Therefore we hypothesize that **hyperbox-based DS approach can improve its performance as more data is added to DSEL and has incremental learning capability**.

0.3 Main contributions

In this research, the concept of Fuzzy Min-Max Hyperbox has been utilized to reach better performance in a new DS framework. This framework uses fuzzy hyperboxes to select the best ensemble of classifiers among all available classifiers. In addition, the proposed approach is capable of using only the misclassified samples to evaluate the competence of classifiers.

According to the literature, the proposed framework is the first DS approach that uses only misclassified samples to select the ensemble of classifiers.

0.4 Proposed approach

In this thesis, a new dynamic ensemble selection framework is proposed called FH-DES which utilizes fuzzy hyperboxes to enhance classifier ensembles for accurate classification of unknown samples. Fuzzy hyperboxes are similar to those in FMM neural networks, but in FH-DES they are assigned to individual classifiers rather than classes and domains. Unlike traditional methods that assess classifiers' competence based on their strengths, FH-DES is able to consider both the strengths and weaknesses of classifiers. Two approaches are presented for generating the hyperboxes.

In the first approach, positive hyperboxes are created using correctly classified samples of each classifier. These hyperboxes define regions in the feature space where a classifier excels, accurately predicting the correct label. The second approach involves generating negative hyperboxes based on a classifier's misclassified samples. These hyperboxes represent regions where the classifier is less accurate. Negative hyperboxes effectively model the capabilities of classifiers, providing a more comprehensive evaluation.

According to the experimental results, the second approach which is based on misclassified has distinctively higher accuracy rather than the method based on correct classified samples. The misclassified-based FH-DES has also less computational complexity. In this approach, the required negative hyperboxes of each classifier are generated during the training phase. Hyperboxes of classifier c_i and their membership functions form incompetence map of c_i which represents the regions of feature space that c_i does not work properly.

In the generalization phase, the optimal ensemble of classifiers is selected based on the incompetence values of the maps at the given query data point. Finally, the outputs of selected

classifiers are combined using a weighted majority-voting algorithm, resulting in an effective classification of unseen test samples. The proposed approach is more efficient regarding accuracy and computational complexity. According to the experimental results, FH-DES has significantly higher accuracy rather than most current DS approaches. Moreover, in some large-scale datasets, the proposed approach generates hyperboxes only as much as 1% of the number of samples.

Related Publication: This thesis provides a journal paper submitted to the Information Fusion journal in July 2023. This paper introduces FH-DES as a dynamic ensemble selection framework to address challenges in KNN-based DES, enhancing generalization by modeling classifier mistakes and discarding incompetencies. This approach significantly improves selection and reduces computational costs compared to existing methods which makes this framework a good alternative DS approach for processing large-scale datasets. In addition, a hyperbox contraction process has been introduced in this paper to add incremental learning capability to the proposed framework.

Moreover, a preliminary version of the proposed framework was published as follows: *Davtalab, Reza, Rafael MO Cruz, and Robert Sabourin. "Dynamic ensemble selection using fuzzy hyperboxes." 2022 International Joint Conference on Neural Networks (IJCNN). IEEE, 2022*. In this paper, the misclassified instances were applied to define the incompetence areas. Moreover, this paper introduced a new membership function to measure the memberships differently with softer boundaries that slightly increase the system's accuracy.

0.5 Structure of the thesis

The next chapter has reviewed research on Multiple Classifier Systems (MCS), Dynamic Selection (DS) approaches, also discussing aspects like competence region definition, selection criteria, and methods in dynamic selection systems. Important aspects of fuzzy hyperboxes

were explored, highlighting their capabilities in processing large-scale data and online adaptive learning. In addition, various membership functions for hyperboxes have been reviewed.

Chapter 3 contains the journal paper submitted to the *Information Fusion* journal. This paper explains the framework which is a hyperbox-based dynamic selection (FH-DES) approach and contains the abstract, introduction, basic concepts on fuzzy hyperboxes, the proposed FH-DES framework, related work, experimental protocol, results, conclusion, acknowledgments, and appendix. In the conclusion chapter, a summary of the different steps and contributions of this thesis has been reviewed and the obtained results have been explained. Finally, in the appendix section, the conference paper that is extracted from the early steps of this research is discussed.

CHAPTER 1

RELATED WORK

This chapter provides the background knowledge of Multiple Classifier Systems, Dynamic Selection approaches, and also Fuzzy Hyperboxes.

1.1 Multiple Classifier System (MCS)

Multiple Classifier Systems (MCS) is an ensemble learning solution for the complex and vast amounts of data that we face today (Cruz *et al.* (2018a); Zyblewski *et al.* (2021)). A multiple classifier system consists of three primary phases (Figure 1.1).

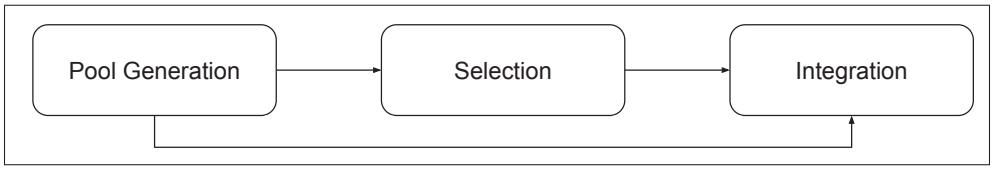


Figure 1.1 The three primary phases of an MCS system. In the Generation phase, a pool of classifiers ($C = \{c_1, c_2, ..., c_M\}$) is generated. In the Selection phase, a subset of classifiers (C') from the pool is selected ($C' \subseteq C$). In the Integration phase, the outputs of selected classifiers are aggregated to give the final decision

Several approaches have been proposed for each of the three phases (Cruz *et al.* (2018a)) and some of them are reviewed in this section.

1.1.1 Classifier generation

The objective of this phase is to generate a collection of base classifiers that are both accurate and diverse (Cruz *et al.* (2018a)). It is crucial for the base classifiers to be distinct, as there is no benefit in combining experts that consistently produce similar outputs (Kuncheva (2014)). Different primary strategies are employed to generate a diverse pool of classifiers (Cruz *et al.* (2018a)).

1.1.2 Selection

The selection process can take place in either a static or dynamic manner during the selection stage. In static selection techniques, the Ensemble of Classifiers (EoC) is selected during the training phase based on a selection criterion determined using the validation dataset. This same ensemble is then utilized to predict labels for all test samples during the generalization phase. Commonly employed criteria for selecting static ensembles include diversity (Cruz, Cavalcanti, Tsang & Sabourin (2013)) and classification accuracy (Ruta & Gabrys (2005)).

1.1.3 Aggregation

The aggregation stage involves merging the results produced by the selected classifiers using a fusion rule. The fusion of the underlying classifiers can be executed through the utilization of class labels, as seen in the Majority Voting method. Alternatively, it can be accomplished by employing the scores generated by the base classifier for each class within the classification task (Cruz *et al.* (2018a)).

1.1.4 The Oracle

The Oracle refers to a hypothetical or ideal MCS defined in (Kuncheva (2002)). The Oracle is often used as a comparison tool to assess how well other MCS approaches are performing. Since Oracle knows the target labels of the test queries, it represents the upper limit of achievable performance. The proposed MCS approaches in different research are compared to The Oracle to determine how close they come to its performance and identify the strengths and weaknesses of these approaches.

1.2 Dynamic Selection Systems (DS)

In dynamic selection systems, a subset of locally competent classifiers is selected to classify each given query sample $\mathbf{x_q}$ instead of aggregating the output of all classifiers. In this approach, we need to estimate the competence of all classifiers around $\mathbf{x_q}$. To do this, Dynamic Selection

Dataset (DSEL) is used to estimate the local competence of all classifiers. DSEL is a set of labeled samples that can be either the training or validation set (Cruz *et al.* (2018a)). Then, a subset of classifiers that have better performance around the query sample is selected. Finally, the outputs of selected classifiers are aggregated to calculate the label of $\mathbf{x_q}$. Selecting classifiers is the key point of these systems and usually involves three major steps (Cruz *et al.* (2018a)):

- **Definition of the Region of Competence (RoC):** includes defining the region where the samples inside have similar features to the query sample. The competence of each classifier is measured based on its performance in this region. For example in KNN-based DS approaches the RoC of $\mathbf{x}_q$ is defined as $\eta_q = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$ where $\mathbf{x}_1, ..., \mathbf{x}_K$ are the K nearest samples to the $\mathbf{x}_j$ from the DSEL data.
- **Determination of the Selection Criteria:** the final ensemble of classifiers for the classification of $\mathbf{x}_q$ is selected regarding the estimated competence level of base classifiers ($\delta_{i,q}$) in the defined region (η_q). Thus, we need an accurate criterion that represents the competence level of classifiers according to the given local region.
- **Determination of the Selection Approach:** in this step, the final ensemble of classifiers $\phi(\mathbf{x}_q)$ which is a subset of the pool of classifiers (C) is selected. Regarding the selection approach, dynamic selection techniques are divided into two main groups: Dynamic Classifier Selection (DCS) and Dynamic Ensemble Selection (DES). Therefore, the measured competence level ($\delta_{i,q}$) is used to either select a single classifier (the most competent one) or an ensemble containing multiple classifiers deemed competent.

Here, these three steps and their related research are reviewed.

1.2.1 Definition of the Region of Competence (RoC)

In most DS techniques, region of competence is defined by KNN technique (Cruz *et al.* (2018a)), in some other research clustering (Lin *et al.* (2014)), Potential Function (Woloszynski *et al.* (2012)), Decision Space (Cavalin *et al.* (2013)), and Graph-based methods (Hou, Xia, Xu & Sun (2016)) are used.

Clustering

In techniques that use clustering (Lin *et al.* (2014); de Souto, Soares, Santana & Canuto (2008); Soares *et al.* (2006)), the samples of DSEL are clustered and then the performance of all base classifiers are estimated on each cluster. Next, during the generalization phase, the nearest cluster to the given query sample $\mathbf{x_q}$ is estimated. Then, according to the competence of the base classifiers in this cluster, the most competent classifiers are selected to classify the sample $\mathbf{x_q}$. In these approaches, clusters are formed during the training phase, so in the generation phase, the query sample $\mathbf{x_q}$ is classified in a short time. However, these approaches often suffer from low accuracy due to the query sample being distant from the centroid or unbalanced data distribution. In addition, clustering methods usually require some user-defined parameters, and their estimation is not an easy task.

KNN

As previously mentioned, most DES approaches define the RoC based on the KNN algorithm (Cruz *et al.* (2018a); Britto Jr *et al.* (2014); Elmi & Eftekhari (2021); Choi & Lim (2021); Elmi & Eftekhari (2020); Cruz *et al.* (2015c, 2017a); Cavalin *et al.* (2013); Ko, Sabourin & Britto Jr (2008); Soares *et al.* (2006)). In these techniques, the competence of base classifiers is estimated according to the classification accuracy of these classifiers on the K nearest neighbors of the given query sample $\mathbf{x_q}$ in DSEL. This method allows the estimation of local competence of base classifiers. A lot of approaches have been introduced based on the KNN algorithm. However, there are some drawbacks, e.g., the computing competence of the base classifier that involves the calculation of distance between query sample $\mathbf{x_q}$, and the whole DSEL. This means, classifying each query sample consists of a high computational cost, and all DSEL samples must be maintained in memory. Therefore, this process imposes a high computation and memory complexity on the DES system. In addition, as we discussed in the problem statement subsection, these techniques suffer from the K value problem, imbalance local distribution, and limited information problems.

Nevertheless, some approaches have been proposed in recent years to tackle these problems (Cruz *et al.* (2015d); Elmi & Eftekhari (2021, 2020); Choi & Lim (2021)). They aimed to improve the performance of DS approaches and the quality of RoC using techniques such as prototype selection (Cruz, Sabourin & Cavalcanti (2017b)), and adaptive distance (Cruz, Cavalcanti & Ren (2011)). However, their performances are far from the performances of the Oracle, which is an abstract concept denoting the upper bound for selection methods.

Potential Function

Different from the other DS approaches, all samples from DSEL are used to calculate the competence level of classifiers regardless of the location of query sample $\mathbf{x_q}$ (Cruz *et al.* (2018a)). In other words, RoC consists of all samples of DSEL (Elmi & Eftekhari (2021)). However, their effectiveness varies so that samples closer to the query sample are more influential in the classifier competence calculation. In these approaches, a Gaussian potential function is considered to calculate the influence of each sample like $\mathbf{x_k}$ according to its distances to the query $\mathbf{x_q}$ as follows:

$$K(\mathbf{x_k}, \mathbf{x_q}) = exp(-d(\mathbf{x_k}, \mathbf{x_q})^2) \tag{1.1}$$

Many DS approaches are introduced using potential functions such as Dynamic Ensemble Selection based on Kullback-Leibler divergence (DES-KL) (Woloszynski *et al.* (2012)), DS systems based on the randomized reference classifier (RRC) (Woloszynski & Kurzynski (2011)), and the approach based on logarithmic and exponential functions (Woloszynski & Kurzynski (2009)). This type of DS approach usually has high accuracy. However, they suffer from high computational complexity because the selection criteria is applied to all data points in DSEL and aggregated according to Equation 1.1. While in KNN-based DS approaches just the influence of samples that are located inside the RoC is estimated.

Decision space

Decision space techniques are inspired by the Behavior Knowledge Space (BKS) (Huang & Suen (1995)). In these methods, the label of the given sample $\mathbf{x_q}$ all individual classifiers predict. The similarity between the output profile of the query sample and the output profiles of the samples in DSEL is used to calculate the region of competence. Several DS approaches are proposed based on decision space such as Multiple Classifier Behavior (MCB) (Giacinto & Roli (2001)), k-Nearest Output Profiles (KNOP) (Cavalin *et al.* (2013)) and META-DES (Cruz *et al.* (2015d, 2017a, 2019a)).

Graph-base

In graph-based approaches, the neighborhood of the presented sample is determined according to the similarity between the predicted labels for the query sample and other training samples (Hou *et al.* (2016)). To do so, the similarity of the query sample $\mathbf{x_q}$ and its neighborhood is estimated using two graphs named must-link and connot-link. The must-link graph connects all samples that are closely related to each other or have the same labels, and assigns weights to the edges according to the similarity of the pair of data points. The connot-link graph connects the samples in the neighborhood with different labels. The competence level of all classifiers is determined according to the structure and weights of these two graphs (Li, Wen, Li & Cai (2019)).

1.2.2 Selection Criteria

After the determination of the competence region, the competence level of classifiers in this region ($\delta_{i,q}$) should be calculated. The final ensemble of classifiers for the classification of $\mathbf{x_q}$ is selected regarding the estimated competence. Thus, we need an accurate criterion that represents the performance of classifiers according to the given local region. The criteria can be categorized into two groups: individual-based and group-based measures (Cruz *et al.* (2018a); Elmi & Eftekhari (2021)). In Individual-based criteria, the competence of each base classifier is

measured independently of the other classifiers of the pool. Many pieces of research have been done in this area and several individual-based criteria have been introduced, such as Ranking (Sabourin *et al.* (1993)), Accuracy (Woods *et al.* (1997)), Probabilistic (Woloszynski *et al.* (2012)), Behavior (Giacinto & Roli (2001)), Oracle (Ko *et al.* (2008)), Meta-learning (Cruz *et al.* (2015c)) and fuzzy sets (Elmi & Eftekhari (2020)). While in group-based measures, the competence of classifiers is measured with regard to the performance of the final selected ensemble. Diversity, Data Handling, and Ambiguity are three subgroups of this category (Cruz *et al.* (2018a)).

1.2.3 Selection approach

In this step, the final ensemble of the classifier must be selected. Regarding the selection approach, dynamic selection techniques are divided into two main groups: Dynamic Classifier Selection (DCS) and Dynamic Ensemble Selection (DES). In DCS techniques, only one is selected to classify the query sample $\mathbf{x_q}$ while DES selects an ensemble of classifiers and then aggregates their outputs.

From another perspective and according to the taxonomy proposed in (Elmi & Eftekhari (2021)), DS techniques can be categorized into three groups: Threshold-Based (TB), Output-Based (OB), and Probability-Based (PB). In the Threshold-Based group, the final classifiers are selected regarding a predefined threshold (Elmi & Eftekhari (2021)). There are a lot of DS techniques in this category such as KNOP (Cavalin *et al.* (2013)), KNORA-E (Ko *et al.* (2008)), KNORA-U (Ko *et al.* (2008)), and DES-P (Woloszynski *et al.* (2012)). Output-Based approaches select just the most competent classifier or a certain number of competent classifiers to form the final ensemble based on a predefined hyperparameter. In this case, if only one classifier is selected, the system will be DCS. MCB (Giacinto & Roli (2001)), OLA (Woods *et al.* (1997)), and LCA (Woods *et al.* (1997)) are some of DS approaches in this category that select only the most competent classifier among the pool. In the Probability-Based (PB) approach, the final ensemble of classifiers is selected according to their probability coefficient by a probability tool such as a roulette wheel algorithm. The probability is assigned to each base classifier according to its

competence level (Elmi & Eftekhari (2021)).

1.2.4 DS algorithms

Several techniques have been introduced in the DS area. In Table 1.1, some of well well-known DS approaches are listed. As shown in this table, current DS approaches use different methods to define the Region of Competence (RoC) and different selection criteria to select the final classifier(s). In this project, we are going to propose a new hyperbox-based DS algorithm in which the competence level of base classifiers is estimated according to the membership degrees of their hyperboxes. The details of the proposed approach are described in Chapter 2.

Table 1.1 A List of Dynamic Selection Techniques

Technique	RoC Definition	Construction Phase	Selection criterion	Type
DCS-Rank (Sabourin *et al.* (1993))	KNN	Generalization	Ranking	OB
OLA (Woods *et al.* (1997))	KNN	Generalization	Accuracy	OB
LCA (Woods *et al.* (1997))	KNN	Generalization	Accuracy	OB
MCB (Giacinto & Roli (2001))	KNN	Generalization	Behavior	OB
MLA (Smits (2002))	KNN	Generalization	Accuracy	OB
DES-Cluster (Soares *et al.* (2006))	Clustering	Training	Accuracy & Diversity	TB
DES-KNN (Soares *et al.* (2006))	KNN	Generalization	Accuracy & Diversity	TB
KNORA-U (Ko *et al.* (2008))	KNN	Generalization	Oracle	TB
KNORA-E (Ko *et al.* (2008))	KNN	Generalization	Oracle	TB
DES-RRC (Woloszynski & Kurzynski (2011))	Potential Function	Generalization	Probabilistic	TB
DES-P (Woloszynski *et al.* (2012))	Potential Function	Generalization	Probabilistic	TB
DES-KL (Woloszynski *et al.* (2012))	Potential Function	Generalization	Probabilistic	TB
KNOP (Cavalin *et al.* (2013))	KNN	Generalization	Behavior	TB
CLAG (Hou *et al.* (2016))	Graph	Generalization	Accuracy	TB
META-DES (Cruz *et al.* (2015d))	KNN	Generalization	Meta-Learning	TB
META-DES.Oracle (Cruz *et al.* (2017a))	KNN	Generalization	Meta-Learning	TB
DSOC (Brun, Britto, Oliveira, Enembreck & Sabourin (2016))	KNN	Generalization	Accuracy & Complexity	TB
CHADE (Pinto, Soares & Mendes-Moreira (2016b))	Meta-Learning	Generalization	Meta-Learning	TB
PCC-DES (Narassiguin, Elghazel & Aussem (2017))	Meta-Learning	Generalization	Meta-Learning	TB
DISi (Pereira, Britto, Oliveira & Sabourin (2018))	KNN	Generalization	Oracle	TB
DDES (Choi & Lim (2021))	KNN	Generalization	Oracle	TB
DES-hesitant (Elmi & Eftekhari (2020))	KNN	Generalization	Multi criteria	TB
MLS (Elmi & Eftekhari (2021))	Multi technique	Generalization	Multi criteria	TB, OB, PB
DES-ML (Elmi, Eftekhari, Mehrpooya & Ravari (2023))	A multi-label classifier	Training	Output of classifier	OB
OLP++ (Souza, Sabourin, Cavalcanti & Cruz (2023))	Recursive partitioning	Generalization	Output of classifier	OB

In the following, some of the important DS approaches are discussed. These approaches were chosen based on their performance reported in (Cruz *et al.* (2018a)) and (Elmi & Eftekhari (2021)), and also based on their number of citations on the Google Scholar website.

Overall Local Accuracy (OLA)

In this approach, the competence level of individual classifier c_i is determined by its classification accuracy in the local region (K nearest sample in DSEL set) (Woods *et al.* (1997)). Hence, the competence of c_j is defined by:

$$\delta_i(\mathbf{x_q}) = 1/k \sum_{k=1}^{k} P(\omega_l|\mathbf{x_k} \in \omega_l, c_i) \tag{1.2}$$

This approach is one of the earlier KNN-based approaches which has good performance in many cases due to the consideration of overall accuracy in a small region.

Local Classifier Accuracy (LCA)

This approach is similar to OLA, with the only difference being that this approach estimates the competence of classifiers based on the samples that belong to the same class (Woods *et al.* (1997)). The local accuracy of each classifier is estimated with respect to the corresponding samples with the same label as the query sample as follows:

$$\delta_i(\mathbf{x_q}) = (\sum_{x_k \in \omega_l} P(\omega_l|\mathbf{x_k}, c_i))/(\sum_{k=1}^{K} P(\omega_l|\mathbf{x_k}, c_i)) \tag{1.3}$$

Finally, the classifier presenting the highest competence level is selected,

MLA

This approach aims to tackle the problem of defining a suitable size of the competence region (Smits (2002)). Previous approaches are very sensitive to the value of K, MLA tries to reduce this sensitivity by weighting each K instance by its distance to $\mathbf{x_q}$. Therefore, the competence

region is defined as below:

$$\delta_i(\mathbf{x_q}) = \sum_{k=1}^{k} P(\omega_l|\mathbf{x_k} \in \omega_l, c_i)W_k \tag{1.4}$$

DES-KNN

In this method, similar to previous approaches, the region of competence η_q is defined based on K nearest neighbors of $\mathbf{x_q}$. Then, the accuracy and diversity of the individual classifiers are computed based on the K neighbors. In the next step, N most accurate classifiers and J most diverse classifiers are selected to compose the final ensemble of classifiers based on double Fault measures (Soares *et al.* (2006)). Here, J and N, ($J \leq N$) are user-defined parameters.

Multiple Classifier Behavior (MCB)

In this method, to determine the label of $\mathbf{x_q}$, the region of competence η_q is estimated first, and the similarity of the samples in this region to the sample $\mathbf{x_q}$ is determined based on the behavior knowledge space (BKS) and profile matrix of each sample (Giacinto & Roli (2001)). Next, samples less similar to $\mathbf{x_q}$ are removed from the set η_q (based on a predefined parameter). Therefore, the size of η_q in this approach is not constant and may be variable. Finally, the sample label is determined by the classification which is significantly more accurate than other individual classifiers. If there is no such classifier, the output is determined by all classifiers in the pool and by the majority voting method.

Modified Classifier Ranking (DCS-Rank)

In this approach, similar to previous methods, the region of competence η_q is estimated (Sabourin *et al.* (1993)). Next, the ranking of each individual classifier is computed as the number of correctly classified samples in the region η_q. The classifier with the highest rank is considered the most competent and query sample $\mathbf{x_q}$ is classified by this classifier.

KNORA-Eliminate

This approach is one of the efficient DES algorithms that firstly estimate the region of competence η_q according to the predefined value of K (Ko *et al.* (2008)). The classifiers which have 100% accuracy in this region are selected. Then, the outputs of these classifiers are aggregated using the majority voting method. If there is no such classifier, the value of K is reduced, and the procedure is restarted.

KNORA-Union

In this algorithm, the classifier ensemble includes all individual classifiers that are able to correctly classify at least one sample in the region of competence η_q (Ko *et al.* (2008)). The outputs of these classifiers are aggregated by a majority voting scheme. In this scheme, the number of votes of each individual classifier is equal to the number of samples that the classifier classified correctly in η_q.

K-Nearest Output Profiles (KNOP)

This approach is similar to KNORA-U with the difference being that the similarities between the query and the validation samples are measured in the decision space rather than the feature space (Cavalin *et al.* (2013)). For this purpose, the output profiles of all samples of DSEL are calculated. At the next stage, the similarity of these profiles is measured with the output profile of query sample $\mathbf{x_q}$. Corresponding samples of similar profiles will form the region of competence. Next, similarly to KNORA-U, the number of votes of each classifier equals the number of samples in the region of competence that the classifier correctly recognizes.

META-DES

This method uses a meta-learner to determine the competence level of each basic classifier (Cruz *et al.* (2015c)). After training all base classifiers in the pool, the meta-features are extracted from the instances of the train set and dynamic dataset (DSEL). In the next step, a meta learner

is trained by the extracted meta-features. This learner determines the competence level of each classifier. According to a predefined threshold, the competent classifiers are selected for the ensemble classifier. If no base classifier is selected, all classifiers of the pool are used to label query sample $\mathbf{x_q}$ with majority voting.

Another variant of this approach has been introduced known as META-DES.Oracle (Cruz *et al.* (2017a)). This approach uses the Binary Particle Swarm Optimization (BPSO) algorithm to select the best subset of meta-features for the training meta-learner. In this approach, the different level of competence was estimated by oracle and the meta-learner is considered as the fitness of the corresponding meta-features.

Local oracles with Discrimination Index (DISi)

This algorithm is an oracle-based dynamic ensemble selection method called Local Oracles with Discrimination Index (DISi). This approach uses a discriminant index originally proposed in the Item and Test Analysis (ITA) (Matlock-Hetzel (1997)) to better define the region of competence (RoC). The closest neighbors are selected by KNN and then the discriminant index is calculated for each of them. The concepts of Professor, Questions, and Student in ITA are represented by competence measures, advisor (nearest samples), and classifiers respectively. The professor uses this kind of index to rank questions to select the most promising ones to evaluate its students in an exam (Pereira *et al.* (2018)).

Chained Dynamic Ensemble (CHADE)

This method is based on the multi-label classification technique, Classifier Chains (CC) (Read, Pfahringer, Holmes & Frank (2011)). CHADE utilizes a meta-learner to predict the competence of base classifiers and select a subset of them. In order to do so, the problem of dynamically combining a set of classifiers is transformed into a multi-label classification problem. In the first stage, the ensemble of classifiers is used to make predictions on DSEL. The predicted labels are compared to true targets to achieve meta-features. Then, a meta-learner is utilized to learn and

select the best set of classifiers. This approach does not rely on the nearest neighbors therefore it does not need to define the regions of competence, however, this approach may converge to an incorrect or local optimal.

Dynamic Ensemble Selection with Probabilistic Classifier Chains (PCC-DES)

This method is based on Probabilistic Classifier Chains. In reality, this approach is an improved version of CHADE, but here, the label dependencies are captured explicitly. To do so, a multi-label procedure based on Probabilistic Classifier Chains and Monte Carlo sampling is utilized to minimize the actual loss function directly. Despite promising results reported in the paper, it suffers from high computational complexity because many calculations are required to classify any query sample.

Distribution-Base Dynamic Ensemble Selection (DDES)

This approach works based on KNN but utilizes different distance measurements to find the nearest instances to overcome traditional problems of KNN such as sensitivity to the local structure of the data and the presence of noisy or irrelevant attributes. Two different versions of this approach are introduced in Choi & Lim (2021): DDES-I, which is an Independent dispersion version, and DDES-M in which Mahalanobis distance is utilized. Both approaches are proposed with the aim of improving the performance of the DS system by properly selecting reference data points for the given query sample. The reported accuracy of these two approaches is promising compared to other DES approaches, especially the DDES-M method which has better accuracy than other DES methods. However, both these methods are based on the KNN algorithm, so they have high computational complexity in the generalization phase.

Multi-Layer Selector (MLS)

In this method, the idea of combining DS methods via multi-layer selectors is offered (Elmi & Eftekhari (2021)). Here, various competence measures are selected in a multi-

layer structure that is expected to lead to an efficient selection of classification dynamically. In each layer, the competence level of classifiers is calculated and the classifiers with high competence levels are passed to the next layer. The best classifiers are passed to the last layer therefore outputs of the last layer will be the final ensemble of classifiers. Different criteria could be used in each layer, so in this way, some criteria are combined to measure the competence level of classifiers. The authors of this paper have designed three different versions of MLS to select classifiers in each layer, including MLSTB, MLSOB, and MLSPB. MLSTB utilizes a Threshold-based selection approach and the classifiers with competence levels that are higher than a predefined threshold are selected. In MLSOB, a certain number of classifiers are selected, and in the MLSPB version, all classifiers have a chance to be selected according to their level of competence. The reported results of this approach are very promising, however as same as the DES-hesitant approach, it suffers from high time complexity to classify each query sample.

1.3 Fuzzy Hyperbox

Hyperbox was introduced by Simpson in 1992 (Simpson (1992)) to use in Fuzzy Min-Max Neural Networks (FMM) (Simpson (1992); Simpson & Jahns (1993)). Hyperbox defined by its two corners named *Min (***v***)* and *Max (***w***)* points. The size and location of hyperboxes are easily adjusted by changing these two corners. Thus, it has a very simple and feasible structure to use. Furthermore, hyperbox-based learning systems have some features that make them good tools in machine learning applications (Khuat, Ruta & Gabrys (2021b)). These features are listed in the following:

- **Make Soft and Hard decision**: Since hyperboxes have a fuzzy membership function they can be used to make soft or hard decisions.
- **Simple and Flexible structure**: This feature of hyperboxes makes them an easy-to-use component and allows us to combine and utilize them within other AI systems such as feature selection (Akbulut (2019)), preprocessing (Kumar & Prasad (2020)), and security applications (Vijayanand *et al.* (2018)).

- **Scalability**: Since the number of hyperboxes is usually much less than the number of learning instances, we expect to have a faster system in the generalization phase.
- **One pass learning**: hyperbox-based approaches are single pass through learning that makes them be able to learn data just by reading it once. This feature helps us to use hyperboxes in cases where we need high-speed information processing, or real-time learning (Khuat & Gabrys (2020)).
- **Online adaption**: Hyperboxes are generated during a learning process with one pass through the data. Therefore, the hyperbox creation can continue until new samples arrive. It means they can learn new concepts over time. Besides, hyperboxes are independent of each other, so each of them can be removed without destroying the whole system. Therefore, the hyperboxes that are not compatible with new concepts can be removed easily. These two key features make hyperboxes capable of online adaption learning (Khuat, Chen & Gabrys (2020)).
- **Granular data modeling**: Hyperboxes could be utilized to design a granular model of data (Lu, Ma, Pedrycz & Yang (2021); Liu, Diao & Guo (2019); Lu *et al.* (2018)). Granular representation of data could decrease the processing complexity, especially in imbalanced data, since the geometric domain of different classes can be represented by granules at a suitable granularity level (Lu *et al.* (2021)).

1.3.1 Creation-Adjustment process

The creation and adjustment of hyperboxes is a simple process. When a new training sample ($\mathbf{x}$) arrives, the system checks if it falls inside a hyperbox. If there is such a hyperbox, no further processing is required and the next training sample will be picked up. Otherwise, we must find a hyperbox that is capable of expansion to include $\mathbf{x}$. To expand the hyperbox B_j to include this sample, the following equations are used:

$$\begin{aligned} v_{jd} &\leftarrow min(v_{jd}, x_d);\ 1 \leq d \leq n \\ w_{jd} &\leftarrow max(w_{jd}, x_d);\ 1 \leq d \leq n \end{aligned} \tag{1.5}$$

Where x_d is the value of sample **x** at $d-th$ dimension. n shows the number of feature space dimensions, and v_{jd} and w_{jd} are respectively *minimum* and *maximum* points at the $d-th$ dimension of hyperbox B_j. Figure 1.2 shows how the hyperbox B_j is expanded to involve the sample **x**. In this example, v_{j2} and w_{j1} are changed to expand the hyperbox.

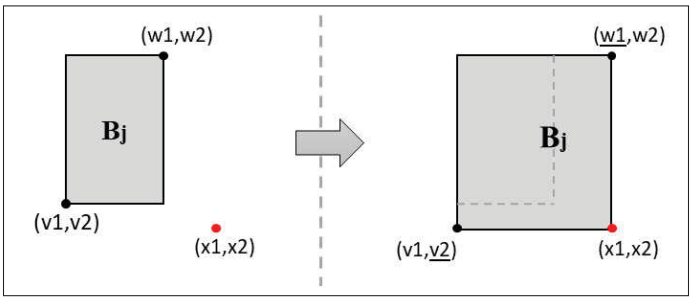


Figure 1.2 Expansion of hyperbox B_j to involve sample **x**

During this process, The maximum size of hyperboxes is limited by the user-defined parameter θ, which in normalized datasets (mapped to [0-1]) are usually between *0* and *1*. So, when a hyperbox B_j is expanding, the following constraints must be met:

$$\theta \geq \max_{\forall d}(w_{jd} - v_{jd});\ 1 \leq d \leq n \tag{1.6}$$

If no expandable hyperbox is found, a new hyperbox is created with min and max points equal to the corresponding points of the sample. Generally, the larger θ, the fewer hyperboxes are created, and the system is more traceable. However, as the θ gets large, the system underfits the data that causes the increased error. On the other hand, small θ creates more hyperboxes that potentially cause overfitting and decrease traceability. So there is a trade-off between the accuracy and traceability of these networks.

Each hyperbox covers, in addition to its internal space, a port of its surroundings. The covered domain by hyperboxes is defined by the following equation.

$$B_j = \{\mathbf{v_j}, \mathbf{w_j}, b_j(\mathbf{x})\}\ \forall\, \mathbf{x} \in I^n \tag{1.7}$$

In this equation, $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ is a single data point. $\mathbf{w_j} = \{w_{j1}, w_{j2}, ..., w_{jn}\}$ and $\mathbf{v_j} = \{v_{j1}, v_{j2}, ..., v_{jn}\}$ are min and max points respectively. b_j is the membership function of the hyperbox B_j. Also, I^n is n dimensional feature space.

1.3.2 Contraction

After creating a new hyperbox or expanding one of the existing hyperboxes, the system inspects the overlapping area of the extended hyperbox with hyperboxes from other classes. Two hyperboxes overlap when each dimension is recognized in one case of Eq. 1.8. To handle this overlap, the dimension (Δ) that has the least overlap, is selected for contraction.

$$\begin{aligned} case1: &\quad v_{ji} < v_{ki} < w_{ji} < w_{ki} \\ case2: &\quad v_{ki} < v_{ji} < w_{ki} < w_{ji} \\ case3: &\quad v_{ji} < v_{ki} \leq w_{ki} < w_{ji} \\ case4: &\quad v_{ki} < v_{ji} \leq w_{ji} < w_{ki} \end{aligned} \tag{1.8}$$

If there is no overlap, no contraction is needed, otherwise regarding the type of overlap in the Δ dimension; one case of Eq. 1.9 is activated to eliminate the overlapped area. Fig. 1.3 shows an example of the contraction process between two hyperboxes.

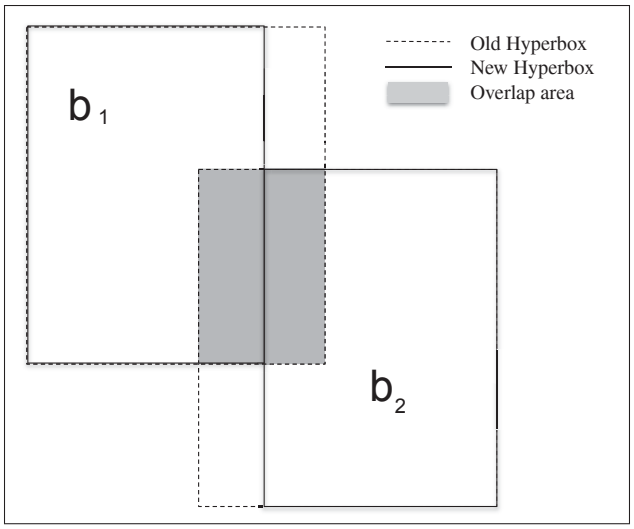


Figure 1.3 Contraction Process between two overlapped hyperboxes

$$
\begin{aligned}
&\text{Case1: } v_{j\Delta} < v_{k\Delta} < w_{j\Delta} < w_{k\Delta} \\
&\qquad v_{k\Delta}^{new} = w_{k\Delta}^{new} = \frac{v_{k\Delta}^{old} + v_{j\Delta}^{old}}{2} \quad \text{or alternatively} \\
&\qquad\qquad (w_{j\Delta}^{new} = v_{k\Delta}^{old}). \\
&\text{Case2: } v_{k\Delta} < v_{j\Delta} < w_{k\Delta} < w_{j\Delta} \\
&\qquad v_{j\Delta}^{new} = w_{k\Delta}^{new} = \frac{v_{j\Delta}^{old} + w_{k\Delta}^{old}}{2} \quad \text{or alternatively} \\
&\qquad\qquad (v_{j\Delta}^{new} = w_{k\Delta}^{old}). \\
&\text{Case3: } v_{j\Delta} < v_{k\Delta} \leq w_{k\Delta} < w_{j\Delta} \\
&\qquad if(\ w_{k\Delta} - v_{j\Delta} < w_{j\Delta} - v_{k\Delta}) \\
&\qquad v_{j\Delta}^{new} = w_{k\Delta}^{old}; \quad else \ \ w_{j\Delta}^{new} = v_{k\Delta}^{old}; \\
&\text{Case4: } v_{k\Delta} < v_{j\Delta} \leq w_{j\Delta} < w_{k\Delta} \\
&\qquad if(\ w_{k\Delta} - v_{j\Delta} < w_{j\Delta} - v_{k\Delta}) \\
&\qquad w_{k\Delta}^{new} = v_{j\Delta}^{old}; \quad else \ \ v_{k\Delta}^{new} = w_{j\Delta}^{old};
\end{aligned}
\tag{1.9}
$$

Here, Δ indicates the selected dimension. It is worth noting that some part of the available information is removed by conducting contraction (Khuat & Gabrys (2021)) and many researchers

have tried to introduce alternative processes (Khuat *et al.* (2020); Davtalab, Dezfoulian & Mansoorizadeh (2013)).

1.3.3 Membership Function

The membership function of the hyperbox is a crucial component in the fuzzy min-max neural network technique. It is utilized to measure the membership degree of belonging of an arbitrary instance to the hyperbox B_j. The membership function of the hyperbox is usually defined in a way that the degree of membership inside the hyperbox B_j equals one, and it decreases when the feature point moves away from the hyperbox.

Simpson's Membership Function

In the original version of FMM, two different hyperboxes are introduced by Simpson for supervised and unsupervised learning (Simpson (1992); Simpson & Jahns (1993)). In the supervised version, the degree of membership is calculated by equation 1.10.

$$\begin{aligned} b_j(\mathbf{x}) = \; & \tfrac{1}{2n} \textstyle\sum_{i=1}^{n} \big[\\ & \max(0, 1 - \max(0, \gamma \min(1, x_i - w_{ji}))) + \\ & \max(0, 1 - \max(0, \gamma \min(1, v_{ji} - x_i))) \big] \end{aligned} \tag{1.10}$$

Where γ is a coefficient between 0 and 1 that regulates how fast the membership values decrease as the distance between $\mathbf{x}$ and B_j increases. Another version of this membership function has been introduced in (Simpson & Jahns (1993)) which is used in unsupervised problems:

$$b_j(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} [1 - f(x_i - w_{ji}, \gamma) - f(v_{ji} - x_i, \gamma)] \tag{1.11}$$

$$f(r, \gamma) = \begin{cases} 1 & \text{if } r\gamma > 1 \\ r\gamma & \text{if } 0 \leq r\gamma \leq 1 \\ 0 & \text{if } r\gamma < 0 \end{cases}$$

Here, γ is the sensitivity parameter regulating how fast the membership values decrease. This function is similar to the previous one. Just with this difference, the membership levels of this membership function decrease faster around the hyperbox. Figure 1.4 (a) depicts a two-dimensional hyperbox and how to cover its vicinity in the supervised case. Figure 1.4 (b) shows the introduced membership function for clustering applications. As is shown in these figures, when the distance to the hyperbox increases, the membership levels in Figure 1.4 (b) decrease faster than the other one.

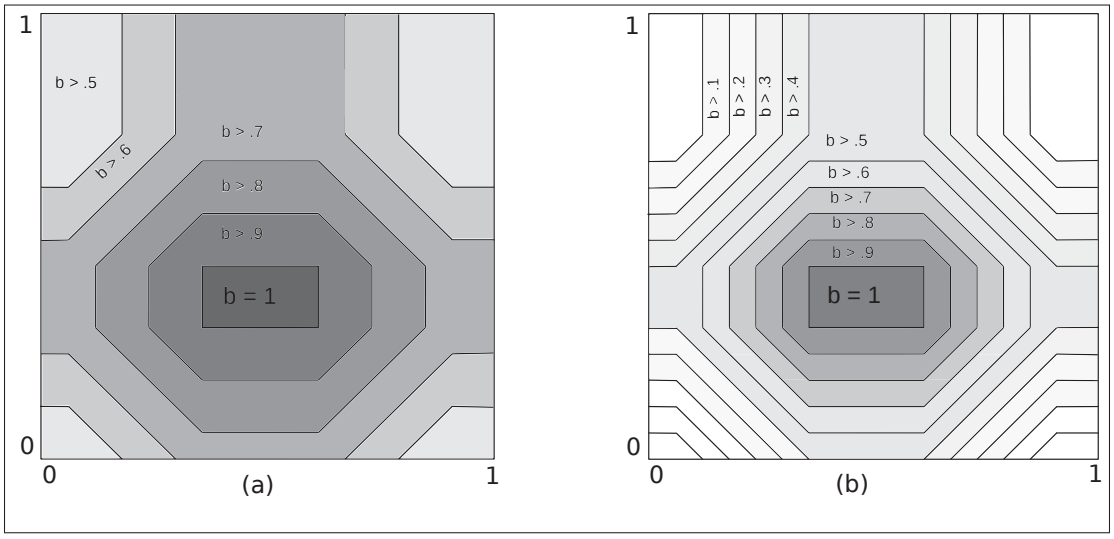


Figure 1.4 Membership function of Hyperbox in (a) introduced for supervised applications, (b) introduced for unsupervised applications. (Sensitivity parameter $\gamma = 0.4$ in both)

Gabrys's Membership Function

Another important membership function has been provided by Gabrys and Bargiela (Gabrys & Bargiela (2000)) as below:

$$\begin{aligned} b_j(\mathbf{x}) \;=\; & \min_{i=1..n} (min([1 - f(x_i - w_{ij}, \gamma_i)], \\ & [1 - f(v_{ij} - x_i, \gamma_i)])) \end{aligned} \tag{1.12}$$

$$f(r,\gamma) = \begin{cases} 1 & \text{if } r\gamma > 1 \\ r\gamma & \text{if } 0 \leq r\gamma \leq 1 \\ 0 & \text{if } r\gamma < 0 \end{cases} \tag{1.13}$$

Where a_i is i_{th} dimension of a sample A, and γ is the sensitivity parameter that regulates how fast the membership values decrease out of the hyperbox. Figure 1.5 illustrates the membership value levels defined by this equation inside and around a two-dimensional hyperbox.

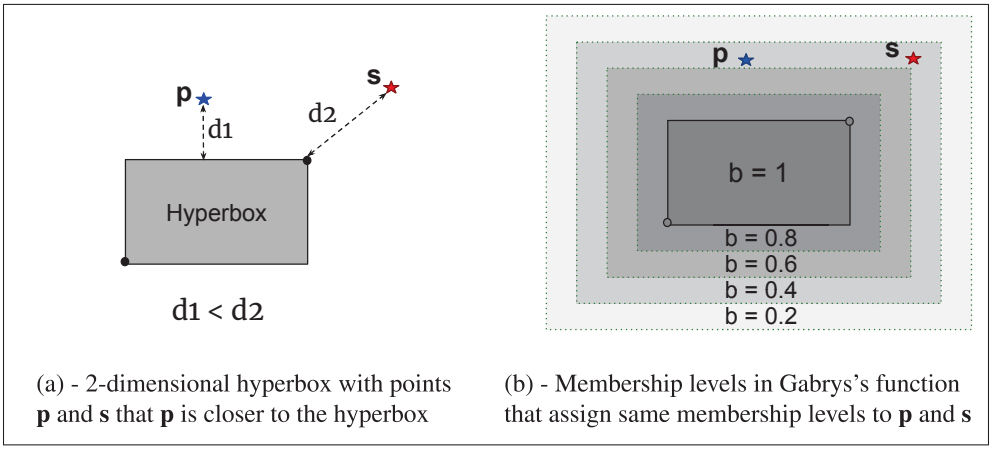


Figure 1.5 Hyperbox's Membership function in GFMM approach

Here, we give an example to show how these membership functions work. Suppose, we have two small groups of samples in which samples of the first group belong to class 1 (Circle) and the other samples belong to class 2 (Square) as shown in Figure 1.6.

To analyze the effect of different membership functions and their predicted boundaries, two hyperboxes are used to represent two groups of samples. Then the predicted boundary of these two hyperboxes is investigated using different membership functions. As a simple benchmark, we used a separator line that has the maximum distance to all samples (purple dashed line). In Figure 1.7, we used Simpson's membership function to predict the class boundary.

As can be observed in this figure, the predicted boundary in the region between the hyperboxes is similar to the benchmark line but in the further areas, the predicted line tends to be parallel to

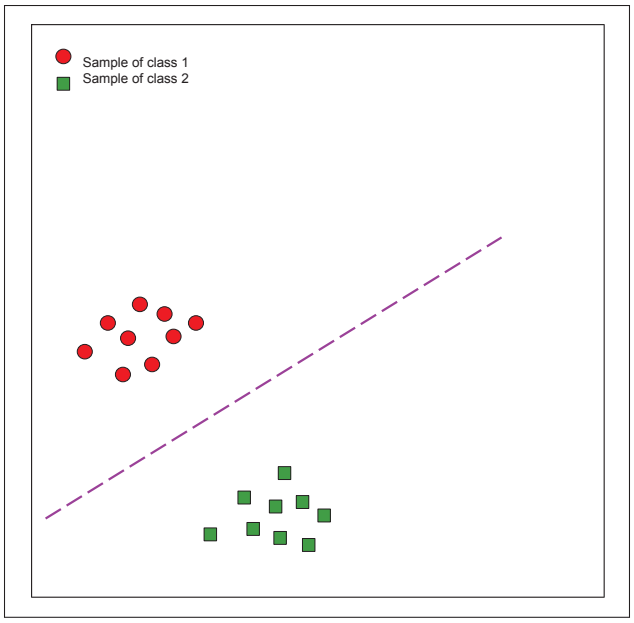


Figure 1.6 Simple example: samples of two classes with a separator line which has the maximum distance to all samples

the coordinate axis. This can lead to many errors in decision-making. In Figure 1.8, Gabrys's membership is applied to the hyperboxes to predict the boundary between the classes. As illustrated in Figure 1.8, although the overall shape of the predicted boundary is similar to the benchmark line, regarding the distribution of samples, it is far from the benchmark line, especially around the hyperboxes that the predicted line is close to the samples and errors are more likely to occur.

Zhang's Membership Funciton

These reviewed membership functions follow the basic definition of a hyperbox (defined using only V and W points). Some other functions have been introduced that use information such as the density of samples inside the hyperbox (Zhang, Liu, Ma & Wang (2011)), the geometric centroid of samples (Alhroob, Mohammed, Lim & Tao (2019)), etc.

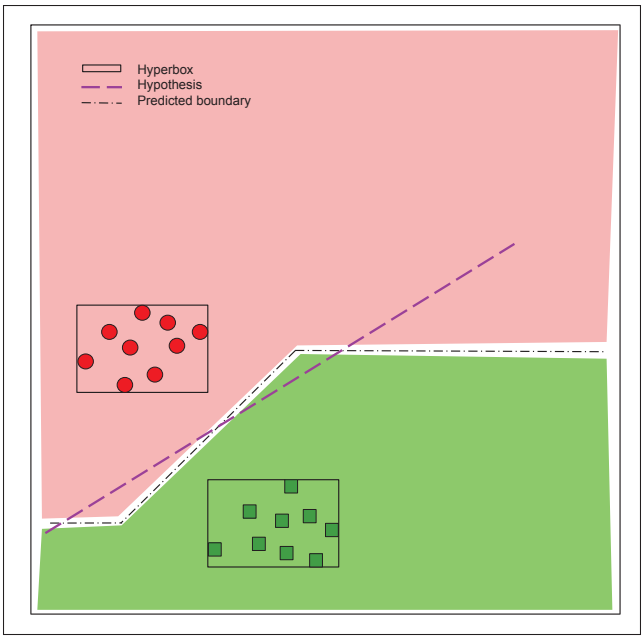


Figure 1.7 Predicting the class boundary using Simpson's membership function

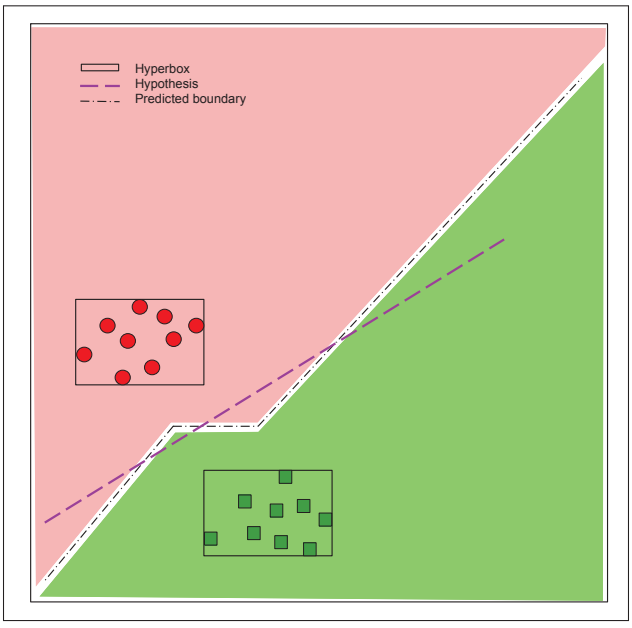


Figure 1.8 Predicting the class boundary using Simpson's membership function

For example, the membership function introduced in (Zhang *et al.* (2011)) considers the density of data in the hyperbox to bypass the effect of noise on the final decision. This membership function is defined as below:

$$\begin{aligned} b_j(\mathbf{x}) \;=\; & \min_{i=1..n}(min(f(x_i - w_{j,i} + \epsilon, p_{j,i}), \\ & f(v_{j,i} + \epsilon - x_i, p_{j,i}))) \end{aligned} \tag{1.14}$$

where ϵ is a parameter representing noise, c is the difference between the data core in the hyperbox and the geometric center of the corresponding hyperbox, and f is the ramp threshold function which is defined as follows:

$$f(r,c) = \begin{cases} e^{-r^2 \times (1+p) \times \lambda} & \text{if } r > 0, p > 0 \\ e^{-r^2 \times (1-p) \times \lambda} & \text{if } r > 0, p < 0 \\ 1 & \text{if } r < 0 \end{cases} \tag{1.15}$$

In this equation, λ is a user-defined parameter to control the descending speed of the membership function.

In Figure 1.9, the influence of parameter c on the shape of the membership function is shown. In this example, v = 0.45 and w = 0.55 are the min and max points of the hyperbox, respectively.

So, this membership function and similar approaches consider the distribution of samples inside the hyperbox. Using this extra information could lead to more accuracy. However, these types of membership functions usually impose a huge computational complexity on the system. Therefore, they are not a good fit for large-scale problems and online learning which are two important data types in this project. Thus, we focus only on the simple fuzzy hyperboxes that are defined only using *Min (***v***)* and *Max (***w***)* corners.

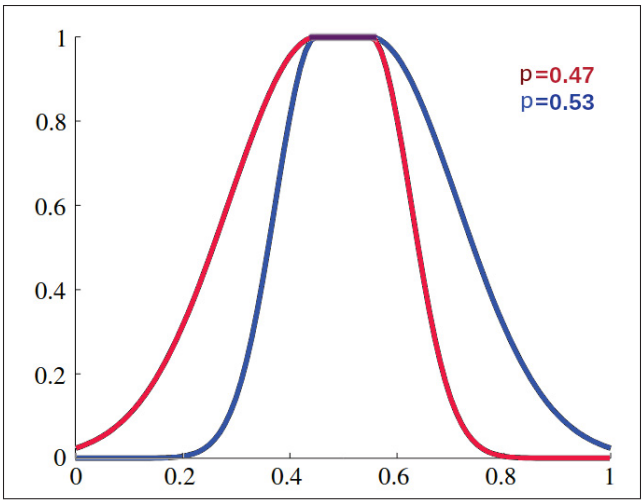


Figure 1.9 Influence of parameter p on the shape of the membership function

1.3.4 Hyperbox and Fuzzy theory in Multiple Classifier System

Fuzzy hyperbox was initially used in fuzzy min-max neural networks for classification and clustering purposes Simpson (1992); Simpson & Jahns (1993). Simultaneously, a similar approach has been introduced, called Fuzzy-Artmap (Carpenter *et al.* (1992)) which uses a similar manner to cover the required regions. Then, a generalized version of these networks called general fuzzy min-max neural network (GFMM) was introduced in Gabrys & Bargiela (2000). This approach utilizes a novel membership function for hyperboxes that uses smoother changes at close distances (Equation 1.12). In the following years, many fuzzy min-max neural networks were developed using fuzzy hyperboxes, primarily in classification area (Nandedkar & Biswas (2007); Zhang *et al.* (2011); Davtalab *et al.* (2013); Waghmare & Kulkarni (2019); Khuat *et al.* (2020); Xue, Huang & Wang (2020)).

Ensemble learning using fuzzy theory has also been active in recent years. In (Kurzynski & Krysmann (2014)), Kurzynski and Krysmann applied two fuzzy inference systems (Mamdani and Sugeno) to develop a fuzzy competence estimator based on Randomized Reference Classifier (RRC) (Woloszynski & Kurzynski (2011)) method to the learning competence measures in

dynamic classification. In this way, the obtained probabilities of RRC are enhanced by the fuzzy inference systems to reach better classification accuracy.

In 2013, Fatemipour and Akbarzadeh introduced a dynamic fuzzy rule-based system for combining learners in a distribution environment (Fatemipour, Akbarzadeh-T & Ghasempour (2014)). In this approach, the reliability of base learners over the entire feature space is estimated using a fuzzy rule-based system. Then, to classify the unknown data x_q the weight of each classifier is estimated by the fuzzy system. The proposed system in this research is a fast combination system. However, it suffers from a lack of accuracy and a large number of rules that must be stored. An improved version of this system is proposed in (Fatemipour & Akbarzadeh-T (2014)), in this version, unnecessary rules are eliminated using the genetic algorithm, however, the improved system steal has low accuracy.

Davtalab et al in (Davtalab *et al.* (2013)) presented a multi-layer fuzzy min-max neural network (MLF) that uses a number of similar classifiers in its structure. This network uses Fuzzy hyperboxes to cover class domains. In practice, MLF is a combination of classifiers that utilizes hyperbox-based classifiers.

In Trajdos & Kurzynski (2016), Trajdos and Kurzynski have utilized the local fuzzy confusion matrix and the concept of RRC and confusion matrix to estimate the competence of classifiers to determine class-dependent probabilities of misclassification and correct classification. This model has been applied for multi-label (Trajdos & Kurzynski (2018)) and Imbalanced data (Trajdos & Kurzynski (2020)) problems in recent years. As reported, the accuracy of these approaches is promising. However, due to using RRC and determining the local region, they have high computational complexity.

Dynamic Ensemble Selection based on Hesitant fuzzy (DES-hesitant) has been introduced recently in Elmi & Eftekhari (2020). In this method, an appropriate ensemble of classifiers is composed by combining different measurement methods using fuzzy hesitant. The best classifiers are passed to the last layer. Then a hesitant fuzzy averaging method and arithmetic mean score function is utilized to select the final ensemble of classifiers. Although this approach

has good accuracy, all DSEL samples are engaged in the competence estimation process. This issue increases its computational complexity.

In Table 1.2 a summary of fuzzy-based approaches is reported.

Table 1.2 Fuzzy-based methods of combining classifiers

Year	Description	Author	Reference
2014	Extracting Fuzzy Rules	Fatemipour and Akbarzadeh	(Fatemipour *et al.* (2014))
2014	Extracting Fuzzy Rules and Optimization with Genetic	Fatemipour and Akbarzadeh	(Fatemipour & Akbarzadeh-T (2014))
2014	Enhance RRC results using Fuzzy inference	Kurzynski et al.	(Kurzynski & Krysmann (2014))
2016	Enhance RRC Using Fuzzy confusion matrix	Trajdos et al.	(Trajdos & Kurzynski (2016))
2018	Combining Multi-Label classifiers using Fuzzy confusion matrix	Trajdos et al.	(Trajdos & Kurzynski (2018))
2020	Imbalanced data DES using Fuzzy confusion matrix	Trajdos et al.	(Trajdos & Kurzynski (2020))
2020	Dynamic Ensemble Selection based on Hesitant fuzzy	Elmi and Eftekhari	(Elmi & Eftekhari (2020))

1.4 Critical Analysis

In this chapter, related research in dynamic selection and fuzzy hyperboxes were reviewed. According to the literature, high computational complexity in the generalization phase, the lack of efficiency in unbalanced datasets, and problems related to the KNN algorithm in KNN-based DS approaches are the main limitations of dynamic ensemble selection approaches. These problems affect the efficiency of dynamic selections in large-scale and unbalanced datasets. Therefore, we need alternative DS approaches to tackle these problems.

In addition, some important aspects of fuzzy hyperboxes were discussed in this chapter. A fuzzy hyperbox is a simple tool with some good capabilities such as making soft and hard decisions, processing large-scale data, and online adaptive learning. Different membership functions have been introduced for hyperboxes. Some of them follow the original definition of hyperboxes and use just min and max points. However, none of them allocate the membership levels appropriately. Some other membership functions also use extra information such as the density or centroid of samples to achieve higher accuracy. However, this type of membership function has higher computational complexity too. Given that in this project we face large-scale data, the first group of membership functions (simpler and faster) has been preferred.

According to the literature, some research has been conducted to apply the fuzzy theory in the DS field like DES-hesitant (Elmi & Eftekhari (2020)) but they suffer from high computational complexity. Thus, we require a different approach for handling large-scale classification problems and addressing imbalanced data.

CHAPTER 2

A SCALABLE DYNAMIC ENSEMBLE SELECTION USING FUZZY HYPERBOXES

Reza Davtalab[1] , Rafael M.O. Cruz[1] , Robert Sabourin[1]

[1] LIVIA, École de technologie supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper submitted for publication, July 2020.

Abstract

Dynamic ensemble selection (DES) systems work by estimating the level of competence of each classifier from a pool of classifiers and selecting the most competent ones for the classification of a given test instance during inference time. The majority of dynamic ensemble selection (DES) methods evaluate the competence of classifiers using the K-Nearest Neighbors to the unknown query sample. However, KNN is very sensitive to local data distribution and needs to store all data in memory. Moreover, it performs several computations for each individual query sample. Thus, relying on the KNN technique hampers the use of DES approaches for large-scale problems and situations where data distributions are non-uniform. This article introduces a novel DES framework called FH-DES, which employs fuzzy hyperboxes to generate a competence map or incompetence map for each classifier. The competence map is generated from correctly classified samples to indicate the competence level of the classifier at each data point in the feature space, whereas the incompetence map, which shows regions where the classifier has low accuracy, is generated from misclassified samples. In this way, we can assess the competence or incompetence level of the classifier just by using the map without having to process previous samples. This feature results in a more accurate dynamic selection system with lower computational complexity compared to other dynamic selection methods. Moreover, we introduce several hyperbox expansion and contraction strategies that add incremental learning capability to the framework while keeping the computational cost low. Experimental results demonstrate that FH-DES achieves high classification accuracy with lower complexity than

state-of-the-art dynamic selection methods. The source code for FH-DES is available at https://github.com/redavtalab/FH-DES.

2.1 Introduction

Multiple Classifier Systems (MCS) is a popular research area in machine learning and pattern recognition due to the fact that using several models leads to improved accuracy (Kuncheva (2014)). In MCS, a pool of classifiers is generated and a subset of these models is used to classify unseen samples. This subset may be static (SS) or dynamic (DS). In static selection, the same subset of classifiers is used to classify all unknown samples. In contrast, dynamic selection techniques use a specific subset of classifiers to classify each unknown sample. DS approaches have been found to lead to more robust ensemble models as they identify and select classifiers that are experts in the local region of the query instance, known as the Region of Competence (RoC) (Kuncheva (2014); Britto Jr *et al.* (2014); Cruz *et al.* (2018a, 2017a)).

Thus, a crucial step in dynamic selection is the RoC definition and associated competence evaluation processes, which are necessary to identify competent experts. The RoC is determined based on a labeled set of examples, which is commonly composed of either the training or the validation data and an algorithm for delineating a local region in the feature space where the samples show similar characteristics with the query. This region is delineated using methods such as K-Nearest Neighbor (KNN) in the majority of the state-of-the-art DS techniques (Cruz *et al.* (2015d); Xiao *et al.* (2016); Krawczyk *et al.* (2018); Cruz *et al.* (2018a); Elmi & Eftekhari (2020)).

Most of the state-of-the-art DS approaches are KNN-based (Cruz *et al.* (2018a)). Although recent studies demonstrated that KNN-based techniques achieve better performance compared to the other approaches (Cruz *et al.* (2018a)), they still suffer from several problems (as illustrated in Figure 2.1):

- **Sensitivity to Hyperparameters:** K is a hyperparameter, even if it has been optimized using an optimization process (Zhang, Cheng, Deng, Zong & Deng (2018)), which is a costly

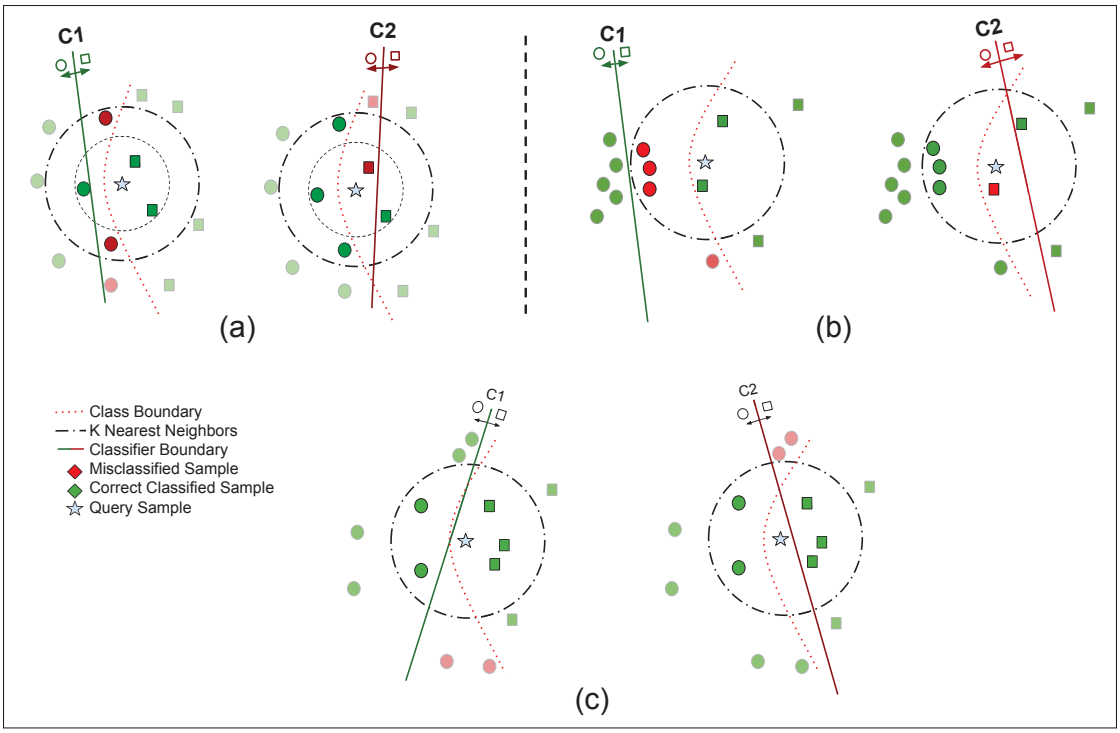


Figure 2.1 Problems of KNN-based DS approaches. (a) Sensitivity to hyperparameter K, different classifiers are selected with K=3 and K=5 (K-value problem), (b) High sensitivity to the local distribution of data, and (c) Limited information problem. KNN uses limited available information. Only c_1 could correctly classify the query sample in all cases, while KNN (K = 5) selects c_2 as a competent classifier

process for large-scale problems. As shown in Figure 2.1(a), K = 5 results in the selection of the wrong classifier (c_2) while K=3 can select the correct one.

- **Sensitivity to local distribution:** KNN algorithms work based on Euclidean distance and have a great sensitivity to the local distribution of the data (Cruz *et al.* (2018b)) as well as noisy data (Elmi & Eftekhari (2020)). Figure 2.1(b) shows an example in which an unbalanced local distribution of instances can negatively affect the competence estimation as it will be biased towards the most frequent class (majority class).
- **Limited Information:** KNN algorithm only considers the RoC information, which is a small region of feature space. Only a small amount of available information is utilized to make decisions. Thus, when the performance of two classifiers is the same inside the RoC, we

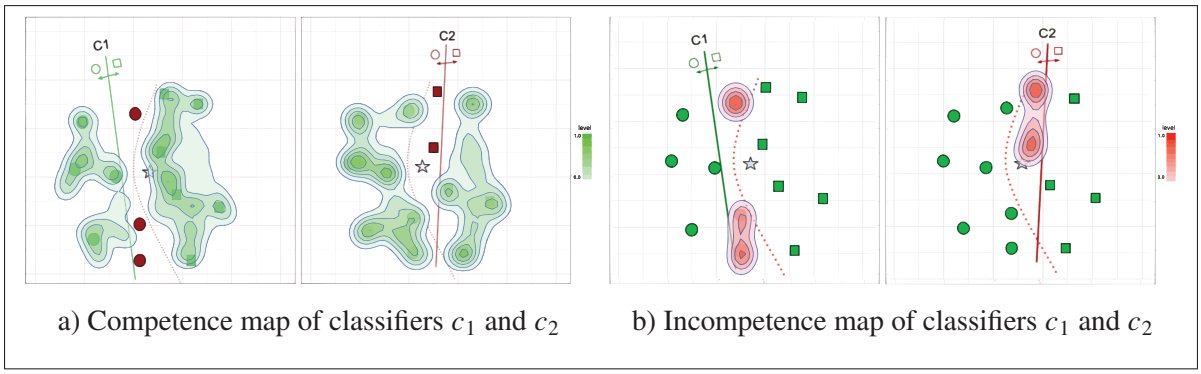
cannot determine which is more competent, and this issue could lead to wrong decisions, as shown in Figure 2.1(c).

- **High computational and storage complexity:** In the generalization phase of KNN-based DS approaches, the nearest K neighbors of each query sample must be found, imposing a considerable computational complexity on the system. Furthermore, all DSEL samples should be stored, which is infeasible in large-scale and data stream applications.

Another alternative for RoC estimation is Clustering-based approaches (Soares *et al.* (2006)), which utilize clusters as regions of competency and only require the storage and computation of distances to the cluster's centroids during the generalization stage. However, this method provides a granular view of the region, which can cause the loss of local information necessary for the proper selection of classifiers, thus resulting in lower accuracy when compared to KNN-based approaches (Cruz *et al.* (2018a)). On the other hand, techniques based on potential functions work by taking into account all data points in the competence estimation process. Such an approach assigns higher weights to samples that are closer to the query sample, while the weight diminishes with increasing distance (Woloszynski & Kurzynski (2011)). However, it is even more computationally expensive than KNN-based approaches, as it not only entails the calculation of the distance between all samples stored in memory but also requires the combination of the information of the entire set with the application of the potential function. Thus, current DS techniques are limited in their application to defining regions of competence, resulting in sub-optimal competence estimates. Novel approaches are needed to improve classification results and reduce computational costs, particularly for large-scale problems and high-dimensional data where the notion of similarity in the Euclidean space is not trivial (Souza *et al.* (2023))

Intuitively, if we have a competence map for each base classifier representing its competence level at each data point in the feature space (Figure 2.2(a)), the labeling process of unknown samples could be much faster in the generalization phase. As shown in Figure 2.2(a), the competence map of the classifier c_i can be formed based on its correct classifications (on DSEL). Also, we can use misclassified samples to create the incompetence map of the classifier. In

Figure 2.2(b), we have used the misclassified instances to create the incompetence map of the classifiers c_1 and c_2 (the same classifiers from Figure 2.1(a)).



a) Competence map of classifiers c_1 and c_2 b) Incompetence map of classifiers c_1 and c_2

Figure 2.2 Competence and incompetence areas of classifiers in Figure 2.1(a)

As shown in this example, each classifier has a high level of competence surrounding its correct classified samples (green areas). In this example, only c_1 can correctly classify the query sample. Moreover, the competence of c_1 is estimated to be higher than that of c_2 using their competence maps. In contrast, the areas around the misclassified samples are marked in red to indicate the low competence level of these regions. Farther distance from these areas means that the classifier is more likely to be a competent classifier, and thus can be selected for prediction. We can see that the query sample in this example is far from the incompetence region of c_1, so we can state that c_1 is a better classifier than c_2.

However, defining such a competence map for each classifier is not easy and imposes a large computational complexity on the system unless some simpler structures are used to represent these areas. For example, in a two-dimensional feature space, we can represent these areas using rectangles. Each rectangle is defined by only two points: the minimum and minimum corners. Each rectangle can summarize the information from multiple instances. Figure 2.3 (a) shows how we can cover the competence areas of each classifier using a few rectangles. Similarly, Figure 2.3 (b) shows how we can cover the incompetence areas (i.e., regions in which the classifier makes mistakes) by the red rectangles. As such, the system's computational complexity will not be high if an acceptable number of rectangles represents all training samples.

In this work, we propose a new DES framework, called Fuzzy Hyperbox Dynamic Ensemble Selection (FH-DES), to achieve the properties mentioned above. FH-DES is based on Fuzzy Hyperboxes (Simpson (1992)), which are virtual rectangles capable of working in high-dimensional spaces. Each hyperbox covers the interior space and also its surrounding area with a fuzzy membership function. This fuzzy aspect gives us valuable information outside the hyperbox, allowing us to estimate the competence of classifiers even if the query sample falls outside all hyperboxes. Moreover, it can leverage more contextual information from the local regions while other approaches, such as KNN-based methods, cannot. During the training phase, FH-DES creates the hyperboxes for each base classifier c_i to represent either its competence or incompetence regions. This creates competence maps that are then used during the generalization phase to estimate the competence level of classifiers. This approach allows for most of the computation required to estimate the competence level of classifiers to be performed during the training phase instead of the generalization phase, thus resulting in a much faster DS system in the generalization phase. In addition, by using hyperboxes, FH-DES is not sensitive to the local distribution of data as it is not affected by instance density.

Hyperboxes of classifier c_i can be generated from either correctly classified samples or misclassified samples. If hyperboxes are constructed from correctly classified samples, they constitute the competence regions of c_i (see Figure 2.3(a)) and are referred to as *positive hyperboxes*. If hyperboxes are created from misclassified samples, they are known as *negative hyperboxes*. In this case, the hyperboxes indicate the incompetence regions of c_i (see Figure 2.3(b)), so if the query sample is distant from the hyperboxes (negative hyperboxes), the classifier is more likely to be competent due to its distance from the misclassified samples of the classifier. Furthermore, as illustrated in Figure 2.3, we can determine which classifier is more competent even when the query sample lies outside of all hyperboxes. In this example, the query sample is situated near the hyperbox of the classifier c_1. Therefore, classifier c_1 is inferred to be more competent than classifier c_2. Consequently, FH-DES captures the information of all hyperboxes distributed in the feature space, thus averting the restricted information issue that current DES approaches face.

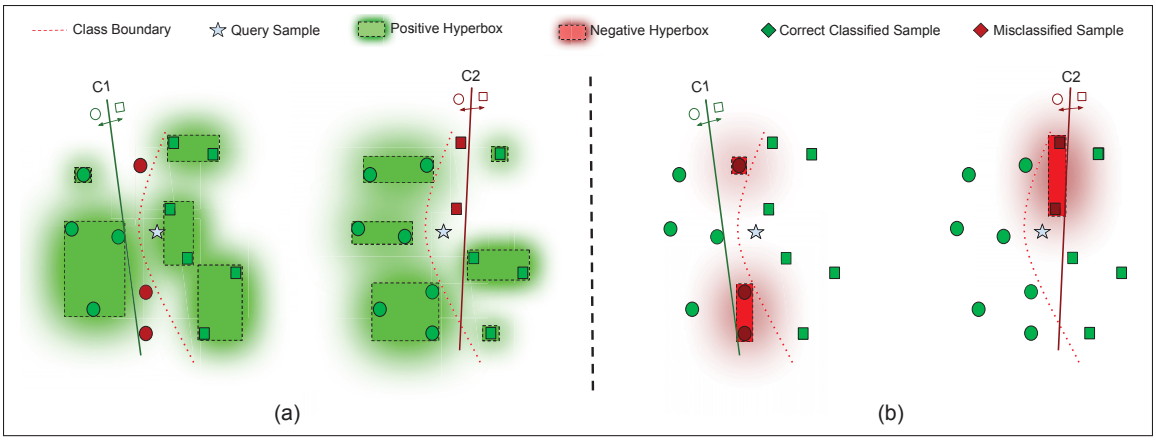


Figure 2.3 (a) Representing the competence map of classifiers using the positive hyperboxes, and (b) Representing the incompetence map of classifiers using the negative hyperboxes for the illustrated example in figure 2.1(a)

In our previous research (Davtalab, Cruz & Sabourin (2022)), we introduced a preliminary version of FH-DES, which demonstrated that fuzzy hyperboxes offer a promising approach to reducing computational complexity while maintaining satisfactory classification accuracy in dynamic selection systems. However, it was hindered by the lack of mechanisms to control the unauthorized expansion of the hyperboxes, thus limiting its classification accuracy and incremental learning capabilities. In this paper, we expand on the fuzzy hyperboxes for dynamic selection idea by incorporating a new learning algorithm and several hyperbox contraction mechanisms to prevent unauthorized expansion hyperboxes. We also propose a new fuzzy mechanism that can control the trade-off between accuracy and system complexity to avoid over-generating new hyperboxes.

Experiments involving multiple datasets of different sizes and characteristics have demonstrated that the proposed approach is more precise and has lower computational complexity than existing DS techniques. Additionally, the system is capable of learning incrementally, as it can improve its complexity while maintaining a low number of hyperboxes when applied to large-scale datasets. Furthermore, our results have shown that taking into account misclassifications through the use of negative hyperboxes results in more robust FH-DES systems. All evaluated scenarios

indicate that accounting for misclassifications leads to enhanced DES systems with regard to accuracy and complexity, thus demonstrating the importance of such information for classifier selection and suggesting that future works in this field must factor it in when designing new methodologies for classifier selection.

In summary, the main contributions of this paper are as follows:

- We present the FH-DES, a fuzzy-hyperbox dynamic ensemble selection framework. FH-DES integrates fuzzy min-max neural networks and a novel smooth borders membership function in order to address the intrinsic issues of KNN-based DES approaches, such as high computational complexity in inference, sensitivity to local distribution, and lack of contextual information. In addition, our proposed framework is able to model the classifier's mistakes (regions of incompetence), unlike conventional DES strategies that only model the competencies of the base model;
- Our approach of modeling the misclassifications of classifiers and discarding incompetent classifiers leads to an improved selection scheme, resulting in a significant improvement in generalization performance when compared to existing methods, as well as a reduction in computational cost;
- We evaluate several strategies for hyperbox selection, expansion, and contraction and their relative impact on the proposed FH-DES system. We also propose a mechanism to significantly reduce the number of generated hyperboxes during the training stage while maintaining the generalization performance;
- We demonstrate the efficacy of the FH-DES in comparison to existing DES solutions over large-scale classification tasks and demonstrate its incremental learning capabilities.

The rest of the paper is organized as follows. Section 2.2 provides an overview of the theoretical background of dynamic selection systems and Fuzzy Hyperboxes. The proposed FH-DES method is described in Section 2.3. Related work in DES and fuzzy hyperboxes is discussed in Section 2.4. Experimental setup and results are presented in Sections 2.5 and 2.6, respectively. Finally, the conclusion and future works are presented in Section 2.7.

2.2 Basic Concepts on Fuzzy Hyperboxes

Table 2.1 Mathematical notation used in this paper

Symbol	Description
θ	represents the maximum size of hyperbox
μ	a predefined threshold to select the best classifiers
λ	learning sensitivity parameter
b_j	the j-th hyperbox
$C = \{c_1, c_2, ..., c_M\}$	the pool consisting of M base classifiers
$\mathbf{x}_q$	a test (query) sample with an unknown class label
S_i^+	a subset of DSEL correctly classified by c_i
S_i^-	a subset of DSEL samples misclassified by c_i
$\eta_q = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$	the region of competence of $\mathbf{x}_q$
$\Omega = \{\omega_1, ..., \omega_L\}$	the set of L class labels
$\phi(\mathbf{x}_q)$	the ensemble of selected classifiers to classify $\mathbf{x}_q$
$\delta_{i,q}$	Estimated competence of classifier c_i for $\mathbf{x}_q$
H_i^+	Positive hyperboxes of c_i
H_i^-	Negative hyperboxes of c_i
H^*	Selected hyperboxes for expansion process

Fuzzy MinMax Neural Networks (FMNN) is a type of neural network that is based on fuzzy set theory and fuzzy hyperboxes. Fuzzy hyperbox was first introduced in fuzzy min-max neural networks for classification and clustering applications (Simpson (1992); Simpson & Jahns (1993)). The idea behind fuzzy hyperboxes is to define a box-like region in the feature space that encompasses a set of training data points belonging to a particular category or class. Fuzzy hyperbox has a simple and flexible structure that makes it possible to use it in different machine-learning applications (Khuat *et al.* (2021b)). The Learning process in hyperbox-based approaches is a one-pass learning mechanism that adapts to online and real-time stream data modeling on a

per-sample basis (Porto & Gomide (2022)). As data is inputted, the data space is granulated and continually adjusted through expansion and contraction operations. This ensures that the number of hyperboxes accurately matches the data, and the structure of hyperboxes is modified whenever necessary (Porto & Gomide (2022); Khuat *et al.* (2021b)). Each hyperbox is assigned a membership function that represents the hyperbox's domain. The granular rule-based model created during learning is transparent, easily interpretable, and understandable (Porto & Gomide (2022)). Hyperbox fuzzy modeling is ideal for data-intensive applications as the models it generates are parsimonious (Porto & Gomide (2022)). In addition, hyperbox-based can be used in different machine learning applications such as speech recognition, image processing (Kumar, Kumar, Bajaj & Singh (2019)), rule extraction (Mohammed & Lim (2017)), feature extraction (Akbulut (2019)), health care (Jahanjoo, Tahan & Rashti (2017)), cybersecurity (Ahmed & Mohammed (2018)), missing value handle (Rey-del Castillo & Cardeñosa (2012)), etc (Khuat, Ruta & Gabrys (2021a)).

The size and location of hyperboxes are easily adjusted by changing their corners, as it uses a simple geometrical structure. Furthermore, hyperbox-based learning systems have some properties that make them good tools for machine learning applications. These features are as follows:

- **Make Soft and Hard decision**: Since hyperboxes have a fuzzy membership function, they can be used to make soft or hard decisions.
- **Simple and Flexible structure**: This feature of hyperboxes makes them an easy-to-use component and allows us to combine and utilize them within other AI systems such as feature selection (Akbulut (2019)), preprocessing (Kumar & Prasad (2020)), and security applications (Vijayanand *et al.* (2018)).
- **Scalability**: Since the number of hyperboxes is usually much smaller than the number of learning instances, we expect to have a faster system in the generalization phase.
- **One pass learning**: hyperbox-based approaches are single-pass through learning that enables them to learn data just by reading it once. This feature helps us use hyperboxes when we need high-speed information processing or real-time learning (Khuat & Gabrys (2020)).

- **Online adaptation**: Hyperboxes are generated during a learning process with one pass through the data. Therefore, the hyperbox creation can continue until new samples arrive. It means they can learn new concepts over time. Besides, hyperboxes are independent of each other, so each of them can be eliminated without destroying the whole system. Therefore, the hyperboxes incompatible with new concepts can be eliminated easily. These two key features make hyperboxes capable of online learning (Khuat *et al.* (2020)).
- **Granular data modeling**: Hyperboxes could be utilized to design a granular model of data (Lu *et al.* (2021); Liu *et al.* (2019); Lu *et al.* (2018)). Granular representation of data could decrease the processing complexity, especially in imbalanced data, since the geometric domain of different classes can be represented by granules at a suitable granularity level (Lu *et al.* (2021)).

2.2.1 Creation and Adjustment process.

When a new training sample (**x**) arrives, the system checks to determine if this sample falls inside an existing hyperbox. If such a hyperbox is found, no further processing is necessary, and the next training sample is picked up. Otherwise, we must find a hyperbox b_j that is capable of expanding to incorporate **x**. To accomplish this, the following equations are used to extend the hyperbox b_j:

$$\begin{aligned} v_{jd} &\leftarrow min(v_{jd}, x_d);\ 1 \leq d \leq r \\ w_{jd} &\leftarrow max(w_{jd}, x_d);\ 1 \leq d \leq r \end{aligned} \tag{2.1}$$

Where x_d is the value of sample **x** at $d-th$ dimension. r shows the number of feature space dimensions, and v_{jd} and w_{jd} are respectively *minimum* and *maximum* points at the $d-th$ dimension of hyperbox b_j. Figure 2.4 shows how the hyperbox b_j is expanded to involve the sample **x**. In this example, v_{j2} and w_{j1} are changed to expand the hyperbox.

During this process, The maximum size of hyperboxes is limited by the user-defined parameter θ, which in datasets with features scaled in the range [0-1] are between *0* and *1*. If no expandable

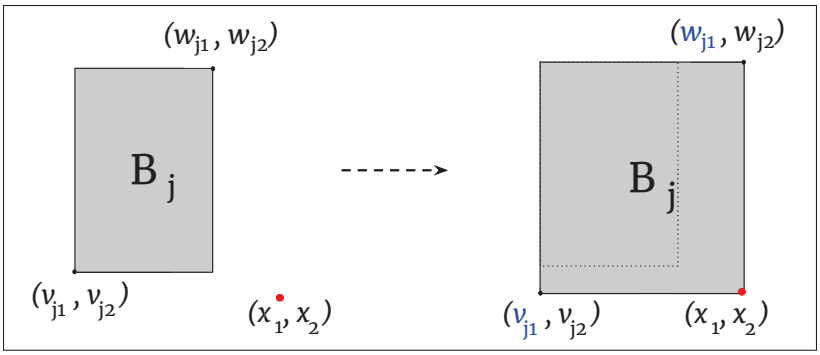


Figure 2.4 Expansion of hyperbox b_j to involve sample $\mathbf{x}$

hyperbox is found, a new hyperbox is created with min and max points equal to the corresponding points of the sample. Generally, the larger θ, the fewer hyperboxes are created, and the system is simpler. However, as the θ gets large, the system may underfit the data, which causes increasing errors. On the other hand, small θ creates more hyperboxes that potentially cause overfitting.

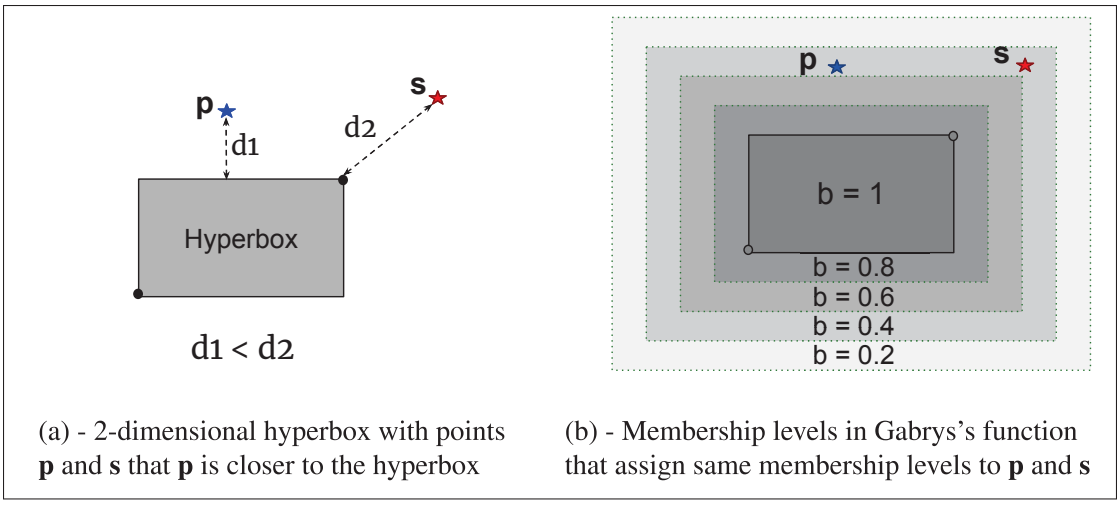
Each hyperbox covers, in addition to its internal space, a part of its surroundings according to a fuzzy membership function. The covered domain by hyperboxes is defined by the following equation:

$$b_j = \{\mathbf{v}_j, \mathbf{w}_j, m_j(\mathbf{x})\} \; \forall \, \mathbf{x} \in I^r \tag{2.2}$$

where, $\mathbf{x} = \{x_1, x_2, ..., x_r\}$ is a single data point. $\mathbf{w}_j = \{w_{j1}, w_{j2}, ..., w_{jn}\}$ and $\mathbf{v}_j = \{v_{j1}, v_{j2}, ..., v_{jr}\}$ are min and max points respectively. m_j is the membership function of the hyperbox b_j and I^n represents an r dimensional feature space.

2.2.2 Membership Function

The membership function is a fundamental element in the hyperbox-based approach (Gabrys & Bargiela (2000); Simpson (1992)), being employed to quantify the degree of membership or association of an arbitrary example to the hyperbox b_j. This membership function is often defined such that the membership degree within the hyperbox b_j is equivalent to one, and decreases as the data point drifts away from the hyperbox. According to the fuzzy min-max neural networks literature



(a) - 2-dimensional hyperbox with points **p** and **s** that **p** is closer to the hyperbox

(b) - Membership levels in Gabrys's function that assign same membership levels to **p** and **s**

Figure 2.5 Hyperbox's Membership function in GFMM approach with its membership levels

(Khuat *et al.* (2021b); Kenger & Özceylan (2023)), one of the most important membership functions has been introduced by Gabrys and Bargiela (Gabrys & Bargiela (2000)) as follows:

$$\begin{aligned} m_j(\mathbf{x}) &= \min_{i=1..n}(min([1 - f(x_i - w_{ij}, \gamma_i)], \\ &\quad [1 - f(v_{ij} - x_i, \gamma_i)])) \end{aligned} \tag{2.3}$$

$$f(r, \gamma) = \begin{cases} 1 & \text{if } r\gamma > 1 \\ r\gamma & \text{if } 0 \leq r\gamma \leq 1 \\ 0 & \text{if } r\gamma < 0 \end{cases} \tag{2.4}$$

Where x_i is the i_{th} dimension of the sample $\mathbf{x}$, and γ is the sensitivity parameter that regulates how fast the membership values decrease out of the hyperbox. Figure 2.5 illustrates the membership value levels defined by this equation inside and around a two-dimensional hyperbox. However, in some cases, this membership function assigns a higher membership to further samples because of its sharp corners. For example, in figure 2.5, points P and S have the same membership

degree while S is farther than P to the hyperbox. This issue can affect the accuracy of the system. Therefore, designing a new membership function with smoother corners can improve the efficiency of hyperbox-based systems.

2.3 The proposed FH-DES framework

2.3.1 System Overview

In this paper, we present the Fuzzy Hyperbox-based Dynamic Ensemble Selection (FH-DES) framework, which employs fuzzy hyperboxes to select the ensemble of classifiers containing the most competent ones for classifying each unknown sample. These hyperboxes are analogous to the hyperboxes of FMM neural networks, except that they are assigned to individual classifiers rather than classes and domains. There are two options to generate the required hyperboxes. In the first way, the correct classified samples of each classifier are used to generate the hyperboxes (called positive hyperboxes). These hyperboxes delimit the regions in the feature space where a classifier is competent. In other words, regions where it predicts the correct label accurately. In the second option, the hyperboxes of the classifier are generated based on its misclassified samples (called negative hyperboxes). The combination of these hyperboxes represents the regions where the classifier is inaccurate. Hence, FH-DES can model the strengths and weaknesses of the classifier, in contrast to other DS approaches where the classifiers are only evaluated based on their strengths (competence) (Cruz *et al.* (2018a)). Figure 2.6 shows the training and generalization phase of the proposed framework. The training phase contains *Hyperbox Creation* and during the generalization phase *Ensemble Selection* process is conducted. In the Hyperbox Creation step, the performance of base classifiers is evaluated using both correctly and misclassified samples. Positive hyperboxes (H_i^+), which indicate competence regions, are created from correctly classified samples (S_i^+), while negative hyperboxes (H_i^-), representing incompetence regions, are formed using misclassified samples (S_i^-). The utilization of misclassified samples is advantageous due to their tendency to reside in informative boundary regions of classification tasks, as most classifiers make errors close to the class boundaries

rather than close to the class means. Moreover, usually, there are fewer misclassified points than correct ones, and that leads to less complex systems as fewer samples are needed for the training and generalization steps.

During the Generalization step, as a new $\mathbf{x}_q$ is presented to the system, its membership degree related to each computed hyperboxes is calculated to estimate each base classifier's competence level. If positive hyperboxes are employed, the membership degree computed over positive hyperboxes directly reflects the classifier's competence level. Conversely, when negative hyperboxes are utilized, the membership degree is subtracted from 1 to identify the classifiers more likely to be competent in classifying the query sample. After estimating their competencies, a selection threshold is computed based on all estimated competence levels, and used to filter out the most competent classifiers to compose the ensemble $\phi(\mathbf{x}_q)$. Then, the outputs of the selected classifiers are combined using the weighted majority-voting algorithm regarding the calculated competence levels as in (Cruz *et al.* (2019b)).

2.3.2 Training phase

The FH-DES training phase involves the generation of hyperboxes to delineate the correct and misclassified samples for each base classifier c_i. This hyperbox creation process is analogous to the hyperbox create-adjustment process in FMM neural networks (Subsection 2.2.1). For each classifier c_i, two sets of samples are used: correctly classified samples (S_i^+) and misclassified samples (S_i^-). They are used to generate the hyperboxes, with S_i^- used to create the negative hyperboxes and S_i^+ used to prevent unauthorized growth. When generating the negative hyperboxes, a sample $\mathbf{x}_t$ is taken from S_i^-, and a hyperbox in H_i^- is found, which encompasses $\mathbf{x}_t$. If such a hyperbox is found, $\mathbf{x}_t$ is omitted from the learning process, and no new hyperbox is created. If no such hyperbox exists, then one is created at the same data point. At all times, the negative hyperboxes should exclusively contain misclassified samples; if this is not the case, then a contraction mechanism should be implemented to shrink the expanded hyperbox. Algorithm 2.1 formalizes this process based on misclassified samples, generating the list of negative hyperboxes (H_i^-) which constitute the incompetence map (Figure 2.3 (b)). If the system

is instead based on a competence map (Figure 2.3 (a)), then the correctly classified samples (S_i^+) are used in the main loop of the algorithm (line 2) to generate the positive hyperboxes (H_i^+).

Algorithm 2.1 Hyperbox creation process in the training phase (based on misclassified samples)

```
   Input: S_i^-, S_i^+
 1 H_i^- = { } ;
 2 for each x_t in S_i^- do
 3     if the maximum membership of x_t in H_i^- is lower than λ then
 4         Sort H_i^- hyperboxes based on their centroids distance to x_t
 5         Select subset H* ⊆ H_i^- based on hyperbox selection strategy (subsection 2.3.2.1)
 6         if There is any expandable hyperbox, b_e in H* according to expansion criterion
            (subsection 2.3.2.2) then
 7             Expand b_e to contain x_t.
 8             if b_e contains any sample of S_i^+ or causes any overlap then
 9                 Contract b_e to handle overlap according to the contraction strategy
                    (subsection 2.3.2.3).
10             end if
11         end if
12         else
13             Create a new hyperbox at the x_t data point and add it to H_i^-.
14         end if
15     end if
16 end for
17 return H_i^-
```

In traditional hyperbox-based algorithms, training samples that are located inside one of the current hyperboxes belonging to the same class are not processed and kept by the system because they do not have any influence on the learned hypothesis. In other words, they do not change the current set of hyperboxes distributions and do not need further processing. All other samples are engaged in the learning process as they can affect the hyperboxes distribution either by expanding or contracting existing ones or by generating an entirely new hyperbox. These steps occur even for samples with high membership degrees to any of the existing hyperboxes sharing the same class. Processing these samples may lead to the creation of unnecessary hyperboxes and increase the computational complexity during training and generalization. Thus, in this

paper, we propose using a *learning sensitivity, hyperparameter (λ)* to define a threshold and determine samples that do not need to be engaged in the learning process.

This hyperparameter can control the computational complexity of the model. Setting $\lambda = 1$ implies that all samples are taken into account in the learning process, analogous to FMM techniques (Khuat & Gabrys (2020)), regardless of the membership values computed for the hyperboxes. Conversely, lowering this value allows for the system to scale to large datasets as fewer samples are used in the learning process. Moreover, fewer hyperboxes are generated, leading to a lower number of stored hyperboxes and membership calculations that need to be performed during inference. Therefore, if accuracy is a priority over computational cost in small to medium-sized classification problems, it is preferable to set it to 1. On the other hand, for large-scale problems, the value can be decreased, as only samples with maximum membership values to the same class's hyperboxes that are lower than λ will be engaged in the learning process; if not, the process will be terminated for $\mathbf{x}_t$, and the next learning sample will be selected.

If the maximum membership value is lower than λ (line 3), the training sample is deemed important, and the adjustment-creation step from the hyperbox learning process is activated. In this case, hyperboxes within H_i^- are sorted in ascending order with respect to their centroid distance from the sample $\mathbf{x}_t$ (Line 6). Then, the expansion candidates hyperboxes are selected regarding *Hyperbox Selection* Strategy (subsection 2.3.2.1). In the next step, the expandability of the selected hyperbox(es) is checked in order of distance. This step aims to find the best hyperbox, among the selected ones, to be expanded to contain $\mathbf{x}_t$. This step is conducted according to the *Hyperbox Expansion* criterion (subsection 2.3.2.2). If any hyperbox is expanded, the final step is to validate whether the expanded hyperbox exceeds its region leading to hyperbox overlap (line 11). In a case where such overlap occurs, the contraction procedure removes possible overlaps by shrinking the concerning hyperbox with respect to *Hyperbox Contraction* Strategy (subsection 2.3.2.3). If none of the selected hyperboxes can be expanded, a new hyperbox is created at the same point and added to H_i^- (lines 13 to 15). Hence, the training process consists of three main

steps: *Hyperbox selection*, *Hyperbox expansion*, and *Hyperbox contraction*, which are described in the following subsections.

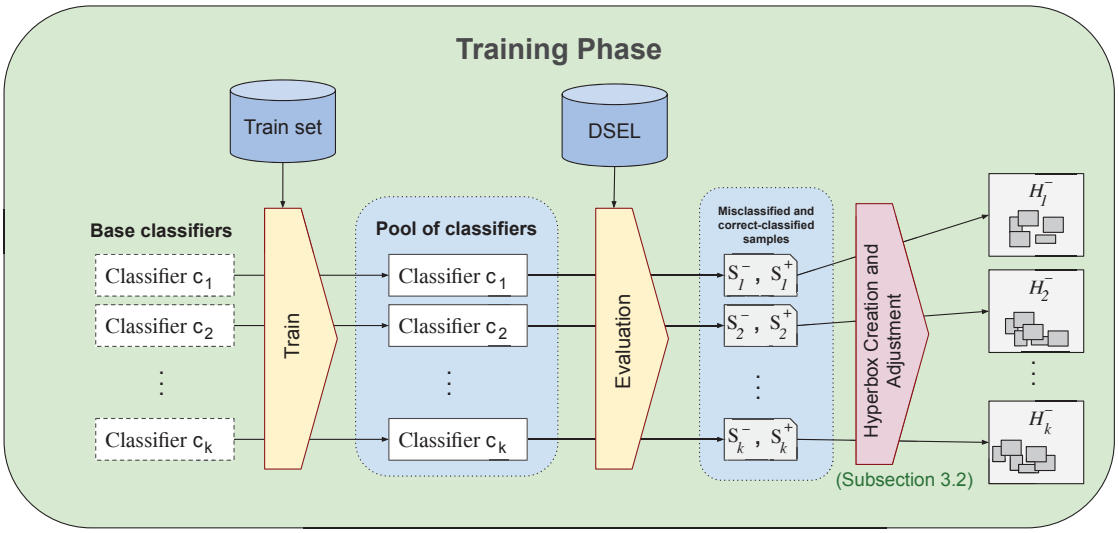
2.3.2.1 Hyperbox Selection

The candidate hyperboxes are selected and passed to the expansion state in this step. There are two strategies for selecting hyperboxes. The first strategy is *selecting all hyperboxes* in H_i^-, which means that all existing hyperboxes in H_i^- are selected and passed to the expansion state. And the second option is *selecting the nearest hyperbox* from the sample $\mathbf{x}_t$ in which the first hyperbox of the sorted list (with the closest centroid) is selected and passed to the next state. In the first selection strategy, similar to FMM neural networks, all hyperboxes are selected and checked to find an expandable hyperbox. However, checking the expandability of all hyperboxes may impose high computational complexity on the system for dealing with large data volumes. Furthermore, the closest hyperbox likely requires the smallest increase in size to include the sample $\mathbf{x}_t$. Thus, if this hyperbox cannot satisfy the expansion criterion, the other hyperboxes that requires a bigger increase in hypervolume are even more likely to exceed the expansion limitation and invalidate the expansion criterion.

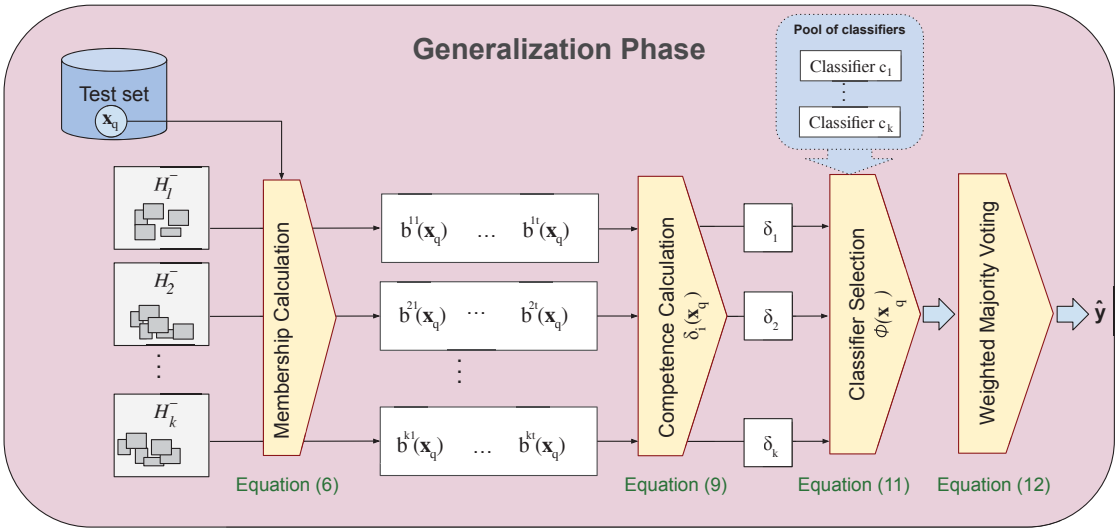
In the second selection strategy, only the hyperbox with the closest centroid to the query sample is selected. It could be a cheaper way to decrease the computational complexity of the FH-DES system in the training phase. However, it may lead to the creation of more hyperboxes that increase computational complexity in the generalization phase. The performance of both selection strategies is investigated in the experimental section.

2.3.2.2 Hyperbox Expansion

This phase aims to expand the selected hyperbox(es) to contain the sample $\mathbf{x}_t$. Only one expandable hyperbox is needed, so the rest of the hyperboxes are not checked if any expandable one is found. The found hyperbox is expanded and passed to the contraction state in this case. Otherwise, a new hyperbox is created to accommodate the training instance.



(a) - Training Phase of proposed framework



(b) - Generalization Phase of proposed framework

Figure 2.6 The proposed FH-DES framework. (a) Training phase; during this phase, all required hyperboxes are formed for each base classifier c_i. S_i^- is the set of samples that were misclassified by c_i. H_i^- is the set of negative hyperboxes formed based on S_i^- and belongs to the classifier c_i. (b) Generalization phase, the membership degree of each hyperbox is calculated. So the maximum membership degree among the hyperboxes which belong to the classifier c_i is reported as the competence level of classifier c_i. The best ensemble of classifiers $\phi(\mathbf{x}_q)$ is selected based on their competence level estimation. Finally, the output of selected classifiers is aggregated using the weighted majority-voting method

Similarly to the FMM neural network learning process, hyperboxes cannot be extended infinitely, and we need a strategy to limit its expansion. In this paper, we consider two criteria to prevent the excessive expansion of hyperboxes. In the first approach, we use the hyperparameter θ in the same way as used in FMM neural networks (Khuat *et al.* (2021b)). So θ works as a threshold to control the maximum hyperbox size as defined in Equation 2.5. Generally, the smaller θ, the more hyperboxes are created. So this hyperparameter decides how many hyperboxes will be created. After the expansion process also, all dimensions of hyperboxes should be smaller than θ otherwise the expansion is not acceptable.

$$\theta \geq \max_{\forall d}(w_{jd} - v_{jd});\ 1 \leq d \leq n \tag{2.5}$$

In the second approach, similar to the IOL-GFMM algorithm (Khuat *et al.* (2020)), an overlap pre-checking process is utilized as an expansion criterion. This process aims to prevent hyperbox expansion only if it leads to an overlap (conflict area). So, in this process, all possible overlaps are checked before doing the expansion. If the expansion causes any overlap, it will be prevented, and another hyperbox will be selected for expansion. Otherwise, the hyperbox is selected as an expandable hyperbox. Therefore, this approach allows hyperboxes to expand as much as it does not cause any overlap. The pseudocode of this expansion stage using the overlap pre-checking criterion is represented in the Algorithm 2.2. This algorithm is based on misclassified samples and negative hyperboxes; if positive hyperboxes are used, the misclassified sample should be utilized to confine these hyperboxes (line 5).

It should be noted that when we use the pre-check approach, no overlap occurs during the competence hyperbox creation process, as an expansion only occurs when it does not invalidate any of the exceeding conditions. Therefore, we do not need to use the contraction mechanism. Furthermore, the pre-check approach allows us to eliminate the hyperparameter θ, which controls the maximum size a hyperbox can grow and can significantly influence the performance of the framework (Davtalab *et al.* (2022)). However, this approach can generate many hyperboxes in the boundary regions of large-scale problems because there are many negative and positive

Algorithm 2.2 Hyperbox Expansion process using Overlap pre-checking criterion (based on misclassified samples)

```
   Input: x_t, H*
1  for each b_e in H* do
2  |  b_candid ← a copy of b_e ;   /* Consider a copy of the current hyperbox
   |   as candidate hyperbox */
3  |  Expand b_candid to contain x_t (using Equation 2.1)
4  |  if Any correct classified sample is located in b_candid or the expansion causes any
   |   overlap then
5  |  |  Reject b_e and check the next hyperbox ;  /* Candidate hyperbox is not
   |  |   acceptable */
6  |  end if
7  |  else
8  |  |  return b_e ;  /* Stop the process and return b_e as an expandable
   |  |   hyperbox */
9  |  end if
10 end for
```

hyperboxes and samples which prevent any expansions of current hyperboxes. So for each new DSEL sample more likely a new tiny hyperbox is created. Figure 2.7 illustrates how a large number of samples cause the creation of tiny hyperboxes in boundary regions. As such, this approach may suffer in the face of large-scale problems and incremental learning.

2.3.2.3 Hyperbox Contraction

Each hyperbox expansion might lead to hyperbox overlap. In the contraction state, these overlaps should be discovered and handled. In this paper, we investigate two types of contraction mechanisms. The first approach is *Hyperbox-based Contraction*, which is similar to the contraction process of the original FMM neural networks (Simpson (1992)). In the second approach, *Instance-based Contraction* is proposed that uses correct-classified samples to discover overlaps.

In the **hyperbox-based approach**, the same original FMM contraction process (Simpson (1992); Gabrys & Bargiela (2000)) is used to handle the overlaps. In this process, to have the minimum adjustment, the smallest dimension of the overlapped area is selected to shrink the overlapped hyperboxes. However, in the proposed approach we have positive and negative hyperboxes

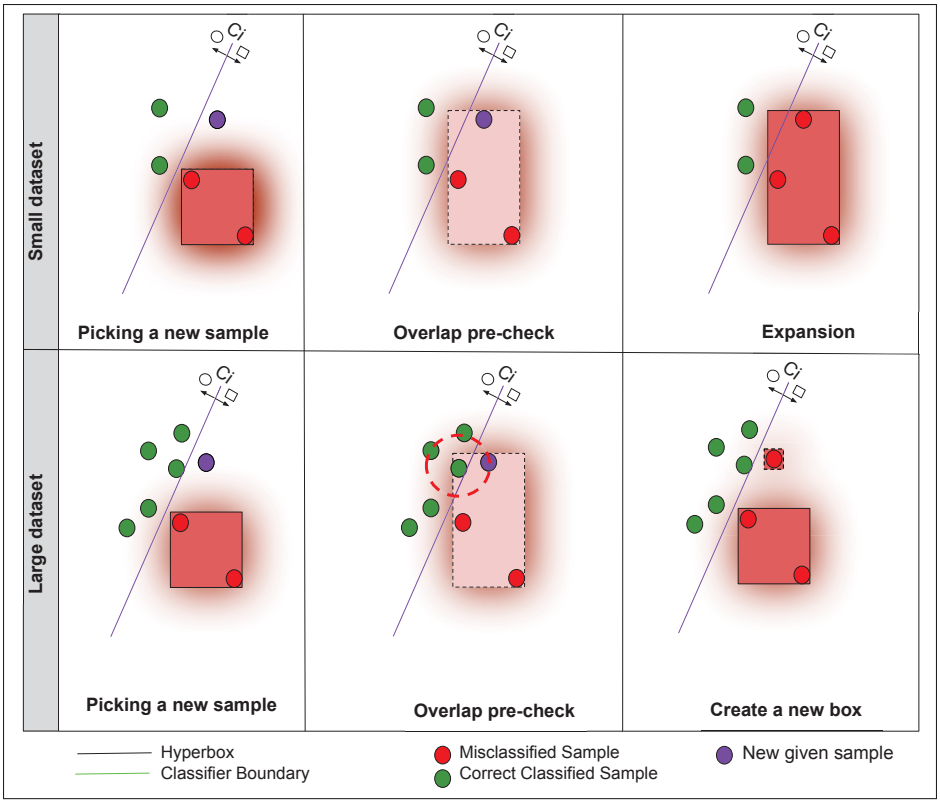


Figure 2.7 Overlap pre-check problem in large-scale datasets. In the first row, there is a sparse dataset. The purple sample has arrived recently. The red hyperbox expands to contain the new sample. Since no correct classified sample falls inside the hyperbox (no overlap), the expanded hyperbox is preserved. In the second row, we have the same problem with more samples. When the hyperbox is expanded to contain the new sample (purple sample), it also contains a correct classified sample. So, the expansion is canceled (the expanded hyperbox returns to the previous state), and a new hyperbox is generated for the purple sample

(H_i^+ and H_i^-) instead of hyperboxes of different classes. Therefore, both positive and negative hyperboxes are needed to find and handle overlaps. However, only one hyperboxes set is used in the generalization phase. When the framework works based on misclassified samples, only the negative hyperboxes (H_i^-) are kept. The pseudocode of this method is illustrated in Algorithm 2.3. If the framework is based on positive samples, in this algorithm, the set of negative hyperboxes (H_i^-) is used to find overlapped areas (line 1). Figure 2.8(a) represents how hyperbox-based contraction eliminates the conflict area.

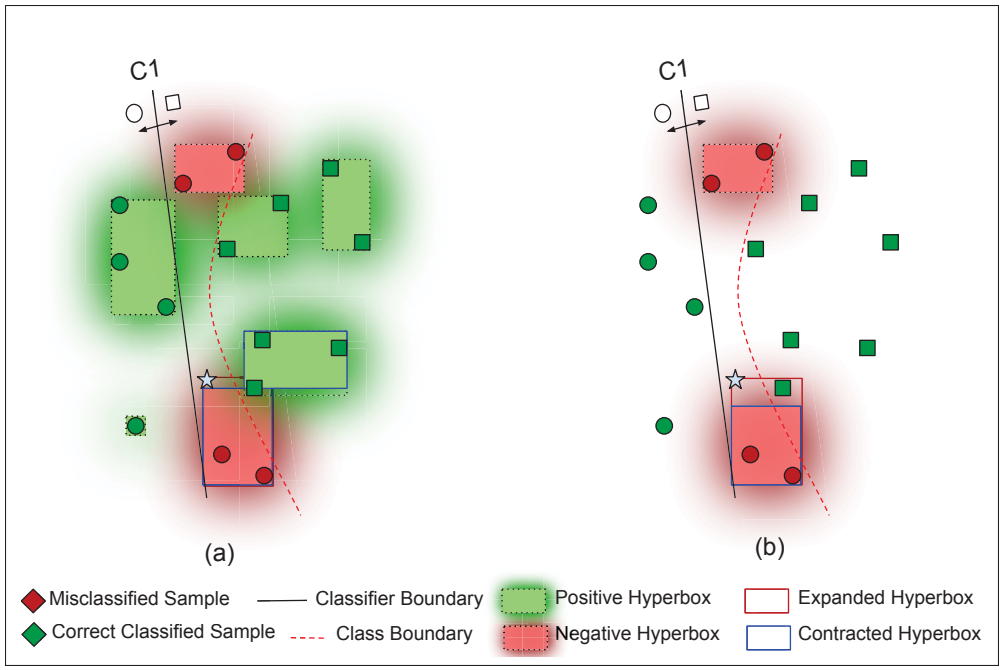


Figure 2.8 Illustration of the hyperbox-based and instance-based contraction strategies. (a) **Hyperbox-based contraction:** The given new sample (star) was misclassified by c_1, and the nearest negative hyperbox of this classifier is expanded to contain it. However, this expansion leads to an overlap. So the contraction mechanism is activated, shrinking the involved hyperbox. (b) **Instance-based contraction:** the nearest negative hyperbox of this classifier is expanded to contain the given sample. However, after this expansion, a correct-classified sample falls inside the expanded hyperbox. So the contraction mechanism is activated to shrink this hyperbox

Algorithm 2.3 Hyperbox-based Contraction Process (based on misclassified samples)

Input: b_e, H_i^+
1 **if** *there is any overlap between the expanded box and hyperboxes of* H_i^+ **then**
2 **for** *each overlap area* **do**
3 Contract b_e and the overlapped hyperbox
4 **end for**
5 **end if**

On the other hand, in the **instance-based contraction** approach, we do not need to inform both negative and positive hyperboxes to perform the contraction mechanism. Instead, the correct-classified samples (S_i^+) are used to discover overlaps and confine negative hyperboxes of the classifier (H_i^-). Figure 2.8(b) represents how the instance-based contraction handles the

conflict area. The pseudocode of this approach is illustrated in Algorithm 2.4. As this model does not need to keep and manage two sets of hyperboxes, i.e., negative and positive, it will likely be more suitable for fast online training.

Algorithm 2.4 Sample-Based Contraction Process (based on misclassified samples)

```
   Input: b_e
 1 if there is any correct classified sample located in b_e then
 2 |   for each correct classified samples x_t located in b_e do
 3 |   |   shrink(b_e,x)
 4 |   end for
 5 end if
 6 ---------------------------------------------------------------------
 7 shrink(b_e, x) ;                          /* x = {x_1, x_2, ..., x_n} */
 8 for each dimension d in feature space do
 9 |   if (x_d − v_ed) < (w_ed − x_d) then
10 |   |   v_ed = x_d
11 |   end if
12 |   else
13 |   |   w_ed = x_d
14 |   end if
15 end for
16 return b_e
```

2.3.3 Generalization phase

The generalization phase in FH-DES consists in selecting the most competent models for the prediction of each new query, $\mathbf{x}_q$, presented to the system. Its main steps are presented in Algorithm 2.5. The first step consists in estimating the competence level of each base classifier c_i. This process is conducted based on its set of hyperboxes generated during the training phase (e.g., H_i^+ for positive modeling, H_i^- for negative modeling) and a membership function which is used to estimate the membership value of $\mathbf{x}_q$ to each hyperbox.

The competence level of each classifier c_i for the classification of $\mathbf{x}_q$ is measured based on the calculated membership degree between $\mathbf{x}_q$ and its set of hyperboxes, which is performed based on a membership function. In this work, we propose a new membership function with smoother

borders, called Smooth Borders Membership (SBM), in order to tackle the problems of Gabrys' membership function that suffers from sharp borders as presented in Section 1.3. The proposed SBM is shown in Equation 2.6.

Algorithm 2.5 Generalization process

Input: $\mathbf{x}_q$
for *each* c_i *in* C **do**
    Calculate the membership value m_j to all hyperboxes
    Calculate the competence $\delta_i(\mathbf{x}_q)$ by eq 2.7,2.8
end for
Compute the selection threshold τ_q by eq 2.9
Select the ensemble of classifiers $\phi(\mathbf{x}_q)$ using eq 2.10
Aggregate outputs of selected classifiers using eq 2.11 to obtain the prediction $\hat{y}$ Return $\hat{y}$

$$m_j(\mathbf{x}_q) = (||ReLU(|\mathbf{o}_j - \mathbf{x}_q| - (\mathbf{w}_j - \mathbf{v}_j)/2)||_2)^2 \tag{2.6}$$

where $\mathbf{o}_j$ denotes the center of the hyperbox b_j, $\mathbf{v}_j$ and $\mathbf{w}_j$ represent the minimum and maximum corners, respectively. $||.||_2$ indicates the L2-norm, and $ReLU(\cdot)$ is the Rectified Linear Unit (ReLU) (Fukushima (1975)) function. The membership levels of SBM are depicted in Figure 2.9, which exhibits smooth corners that result in closer data points receiving a higher membership level. This is in contrast to Gabrys's membership function (Figure 2.5), where point P is closer to the hyperbox than point S yet has a lower membership estimation. This property consequently provides better membership value estimates across the set of hyperboxes, leading to increased accuracy (Davtalab *et al.* (2022)).

Note that in some cases, a noisy data point may cause a hyperbox to be formed in an undesirable position leading to an inaccurate final decision. To address this issue, we consider the top two hyperboxes with the highest membership value (m^{i+}) and aggregate their membership values to compute the competence level of the base classifier c_i, denoted by $\delta_i(\mathbf{x}_q)$). We refer to these hyperboxes as the *strong hyperboxes* of c_i. Thus, when the proposed framework works based on

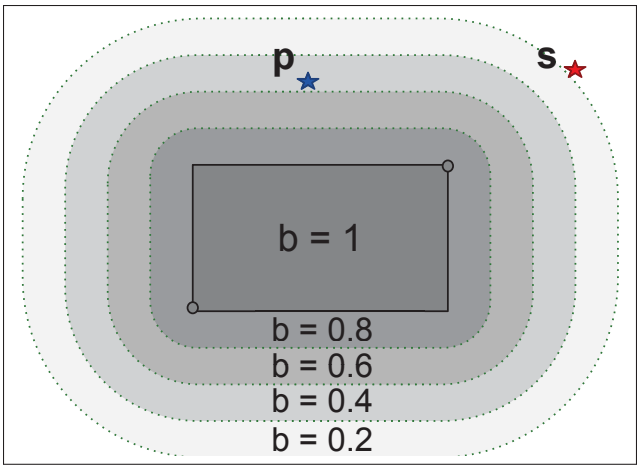


Figure 2.9 The proposed Smooth-Border membership function (SBM) utilized in the proposed FH-DES framework

correctly classified samples, the competence of the classifier c_i to classify the query sample $\mathbf{x}_q$ can be calculated as follows (Equation 2.7):

$$\delta_i(\mathbf{x}_q) = (m^{i*} + m^{i+})/2 \tag{2.7}$$

Analogously, when the framework is based on modeling misclassified samples, the competence of classifiers is calculated as follows (Equation 2.8):

$$\delta_i(\mathbf{x}_q) = 1 - (m^{i*} + m^{i+})/2 \tag{2.8}$$

After obtaining the competence estimates of all base models, the system proceeds to the selection process. In this phase, models are chosen based on their competence level estimates, with a focus on identifying those more inclined towards making correct label predictions. The selection process initiates by estimating a threshold, denoted as τ_q, that is defined based on the competence estimates computed for $\mathbf{x}_q$ (Equation 2.9). This threshold plays a pivotal role in filtering the best classifiers to compose the ensemble.

$$\tau_q = \mu \times \max_{i=1..M} (\delta_i(\mathbf{x}_q)) \tag{2.9}$$

where μ is a predefined hyperparameter set in a range between 0 and 1. Classifiers whose competence is higher than this threshold are deemed competent and are selected to compose the ensemble. Therefore, the final ensemble of classifiers for the classification of $\mathbf{x}_q$ ($\phi(\mathbf{x}_q)$) is formed considering the threshold τ_q according to Equation 2.10.

$$\phi(\mathbf{x}_q) = \{c_i | \delta_i(\mathbf{x}_q) \geq \tau_q\} \tag{2.10}$$

It is important to mention that, When μ equals one, only the most competent classifier(s) is selected. Lower μ values lead to a more permissive classifier selection system, increasing the number of selected experts for performing the final decision.

Finally, in the aggregation step, the outputs of the selected classifiers are combined with weighted majority voting such that classifiers with higher competence estimates ($\delta_i(\mathbf{x}_q)$ have a greater influence on the final decision (Cruz *et al.* (2015c)). Equation 2.11 formalizes the weighted majority voting used by FH-DES.

$$\hat{y} = arg \max_{\Omega} \sum_{\forall l \ \in \Omega} \delta_i(\mathbf{x}_q) \quad | \quad c_i(\mathbf{x}_q) = l, c_i \in \phi(\mathbf{x}_q) \tag{2.11}$$

Where Ω represents the set of class labels. The choice for weighted majority voting in this work is grounded on the DS literature, as the majority of DS methods are based on majority voting. In particular, we use the weighted version based on competence estimates since the work presented in (Cruz, Sabourin & Cavalcanti (2015a)) demonstrated that a weighed voting scheme leads to improved accuracy. This aggregation rule is interesting since it only requires the predicted labels, which allow aggregating responses of heterogeneous classifiers as well as models that are not well calibrated for predicting reliable probability scores.

2.3.4 Case Study

Case I. To illustrate how the proposed approach works, consider two classifiers (c_1 and c_2) that we wish to combine into an ensemble. c_1 labels all samples as class A (purple), while c_2 labels them as class B (yellow). This implies that, for each unknown sample, there exists a classifier that accurately classifies it. Figure 2.10 illustrates an example of the classification of a query that is close to the class borders. Let us assume that b_i and b_j are strong hyperboxes of classifier 1, and b_t and b_w are strong hyperboxes of classifier c_2 at the given point and are highlighted in bold. The degree of membership of these hyperboxes is: $m_i = 0.7$, $m_j = 0.2$, $m_t = 0.3$, $m_w = 0.2$, and $\mu = 0.9$. Using this information, the competence of each classifier around the data point $\mathbf{x}_q$ is computed (Equation (2.7)), giving $\delta_1(\mathbf{x}_q) = 0.55$ and $\delta_2(\mathbf{x}_q) = 0.75$ for c_1 and c_2, respectively. The threshold parameter $\tau_q = 0.675$ is then calculated based on the membership degree of the strongest hyperbox (Equation (2.9)). This threshold leads to the selection of c_2 only. Therefore, the predicted label of the sample $\mathbf{x}_q$ is B, which is a correct decision. In contrast, the low sample density close to the class boundary can lead to wrong decisions in the KNN-based approaches.

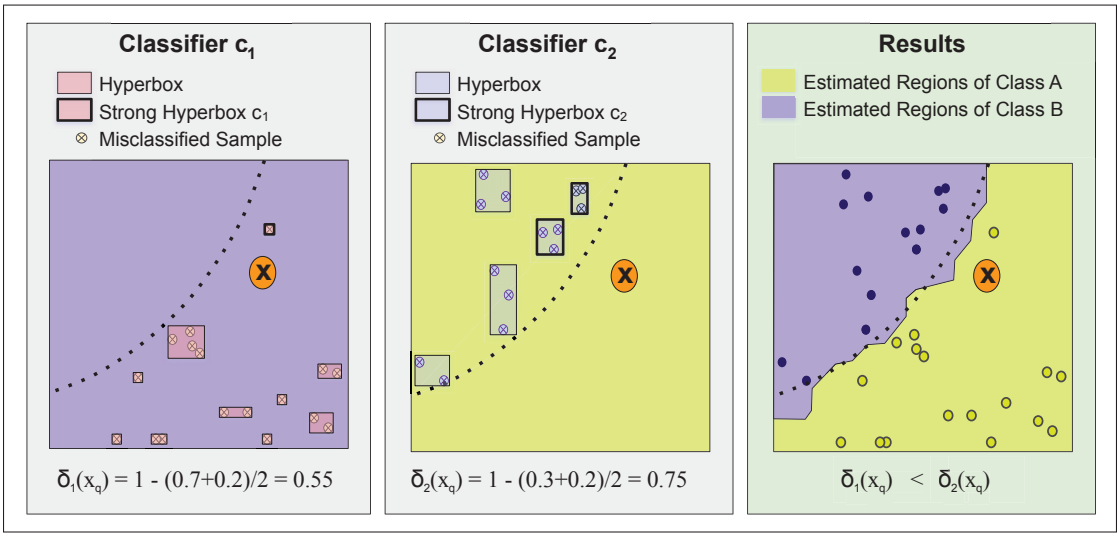


Figure 2.10 Performance of the proposed approach in the boundary regions

Case II - P2 problem example. In a more detailed example, we used the same classifiers (c_1 that labels all samples as class yellow, and c_2 that labels them as purple) to solve the problem P2

(Cruz *et al.* (2015b)) and compared the performance of the proposed framework and KNN-based DES techniques such as KNORA-E, KNORA-U, OLA, DES-KNN, and META-DES (with k=7). The problem P2 (Figure 2.11 (a)) is a binary classification with complex non-linear borders where classes are defined in multiple decision regions determined by polynomial and trigonometric functions (Figure 2.11) (Cruz *et al.* (2017a)). In this problem, both classes have the same prior probability, and there are no overlapping regions between the two classes. Therefore, this dataset can be a good choice for visualizing the modeling behavior of different classification systems.

According to the obtained results, 48 samples of 500 test data were misclassified by all KNN-based approaches, while each query sample is classified correctly by one of the base classifiers. These errors are represented by red star markers in Figure 2.11 (b). Most of these common misclassified samples are located in regions where the local distributions of DSEL samples are not uniform and are more likely errors due to KNN's performance. Thus, showing how relying just on KNN can limit DS systems. On the other hand, the FH-DES framework solved this problem only using 61 hyperboxes and obtained higher accuracy than KNN-based approaches. This framework could correctly classify 21 out of these 48 samples. These samples are shown as green pentagons in Figure 2.11.

This example shows that the proposed framework can address some drawbacks of KNN-based approaches and is more robust against the problems related to the local distributions of data, namely regions with lower sample density as well as local imbalanced class distributions.

Case III - The impact of contraction mechanism in the training process. As mentioned earlier, a preliminary version of FH-DES has been proposed in (Davtalab *et al.* (2022)) used one group of misclassified or correct-classified samples during the training stage to create the hyperboxes set. Therefore, it did not employ *contraction* mechanism or any other alternative to confining the hyperboxes in their desired area as its training process has no knowledge about the right boundaries from the regions of competence and incompetence of a given classifier. Thus, allowing them to grow in unauthorized regions, hindering the system's performance for incremental learning and large datasets.

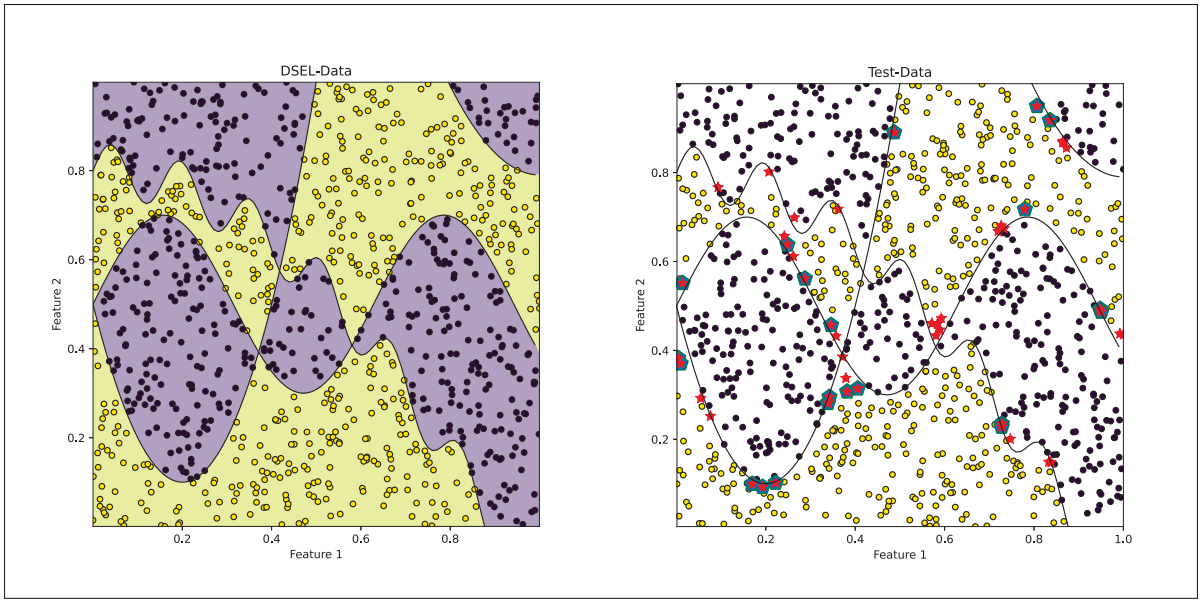


Figure 2.11 Evaluating the proposed framework and KNN-based DES approaches on the P2 problem. a) The DSEL set and classes domains in p2 problem. b) The P2 Test set. The common misclassified samples by KNN-based DES approaches are highlighted in red. Instances with green pentagons are the ones that were corrected by FH-DES

To illustrate the impact of the contraction mechanism, the P2 problem and pool of classifiers composed of two classifiers are considered in this analysis.

In this example, the base classifiers are simple Perceptron linear models trained using 1000 training samples (500 instances for each class). The class domain determined by the classifier c_1 on the P2 problem is shown in Figure 2.12(b), and the performance of c_2 is shown in Figure 2.12(c). In this figure, samples of different classes are shown by small yellow and purple circles. The predicted class domains by the classifiers are highlighted in yellow and purple as well.

Assume three samples $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ that are presented to the system as shown in figures 2.12(d), (e), and (f), respectively. In this example, $\mathbf{x}_1$ and $\mathbf{x}_2$ are misclassified samples of the related classifier. However, $\mathbf{x}_3$ is the correct classified one. Therefore, $\mathbf{x}_1$ and $\mathbf{x}_2$ are used to build the negative hyperbox (es); however, $\mathbf{x}_3$ is neglected. With the arrival of the sample $\mathbf{x}_1$, the first hyperbox is created at the same location (Figure 2.12(d)). When $\mathbf{x}_2$ appears, the hyperbox is expanded to involve $\mathbf{x}_2$ (Figure 2.12(e)). This hyperbox is supposed to represent

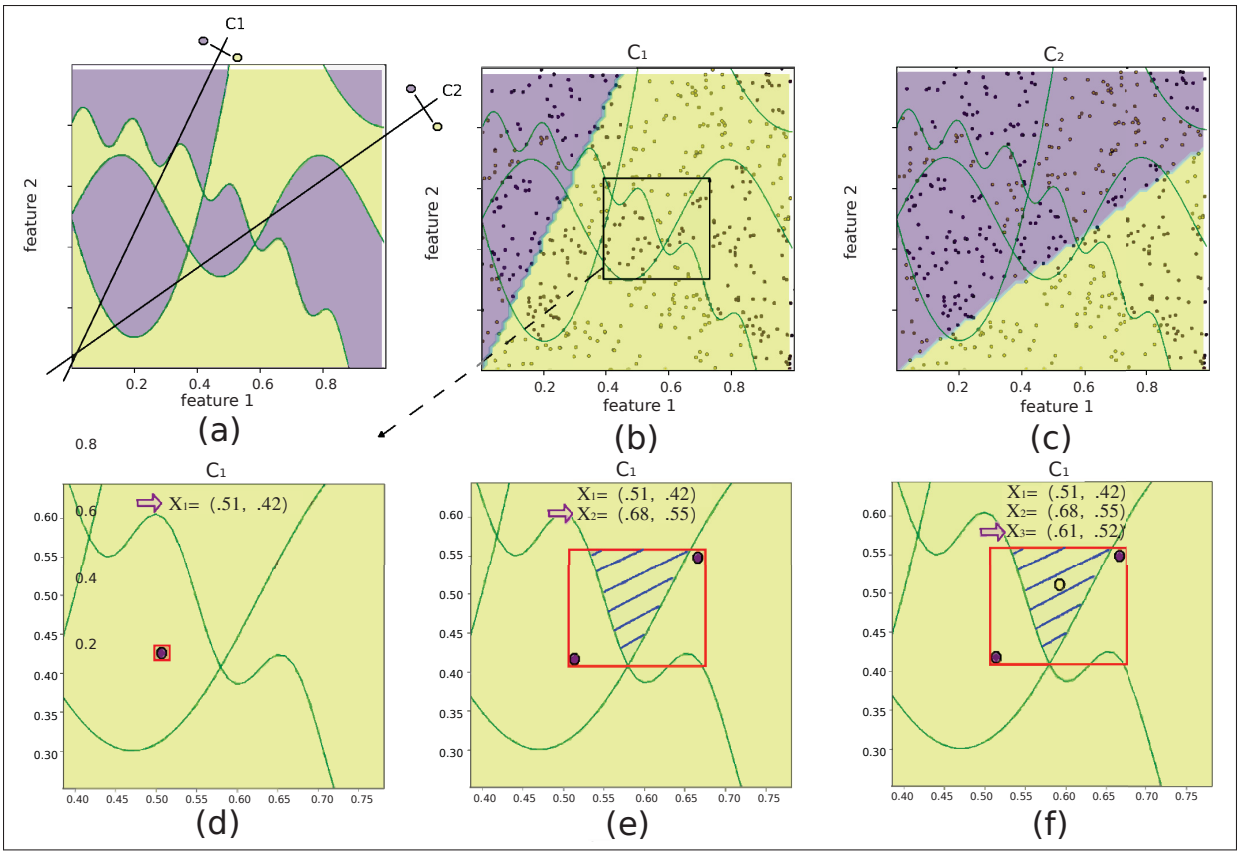


Figure 2.12 (a) Class areas in P2 Problem and the decision boundary of c_1 and c_2, (b) Classes domains determined by classifier c_1, (c) Classes domains determined by classifier c_2, (d) Creating a negative hyperbox by arriving the first sample, (e) Expanding the hyperbox to include $\mathbf{x}_2$, and (f) A correct-classified sample falling inside the negative hyperbox

the incompetence area, however, by the expansion process, it exceeds the competence areas (shown by blue hash). In the third step, $\mathbf{x}_3$ arrives. As shown in Figure 2.12(f), $\mathbf{x}_3$ has fallen into the negative hyperbox, but it has been neglected because it is a correct classified sample. Therefore, without a contraction mechanism, we cannot correct the errors that occurred during the training process, as shown in Figure 2.12(f). To solve this problem, negative hyperboxes should be confined to "incompetence regions". To do so, in this paper, we used misclassified samples to create the negative hyperboxes and also correct-classified samples to confine these hyperboxes in incompetence areas during the training phase. However, only negative hyperboxes of the classifiers are passed to the generalization phase.

In addition, Figure 2.13 illustrates how the proposed framework can solve this problem utilizing the contraction mechanism. In the first row of this example, the same classifiers are shown as in the example in Figure 2.12. In addition, the hyperboxes were created and adjusted with the contraction mechanism. On the left-hand side of the second row, the classification result of the proposed approach without the contraction mechanism is shown. The same result using the contraction mechanism is shown on the right-hand side. As observed, no correct-classified sample has fallen inside the negative hyperboxes, which results in a more accurate system.

2.3.5 Computational and storage complexity

The main steps for selecting the most competent models in DES techniques are defining the region of competence and applying the competence estimation criterion over this region. During the region of competence estimation step, KNN-based DS approaches require computing the distances between the new input $\mathbf{x}_q$ and all other samples belonging to DSEL. Given that this set comprises N examples, such methods require storing all these N examples for inference and computing the distances between the query and all DSEL samples. Hence, it has a computation and storage cost that increases linearly with the DSEL size ($\mathrm{O}(N)$).

Following the computation of regions of competence, these techniques proceed to estimate the competence of each base model within the pool. This competence estimation corresponds to applying a specific criterion over the k nearest instances. As the ensemble comprises a total of M base models, the estimation procedure necessitates $M \times k$ applications of the competence criterion to derive the competence estimates for each base model.

In contrast, in the proposed framework, we eliminate the need to keep DSEL instances as the hyperboxes summarize a whole region of the feature space. Given that e is the total number of hyperboxes (i.e., contains the hyperboxes characterizing the regions of competence or incompetence of all base models) found after the training stage, the total memory cost consumed by the model in the worst-case scenario is $2 \times e$. However, in real-world applications, the number of hyperboxes is usually significantly lower than the number of training instances ($e << N$)

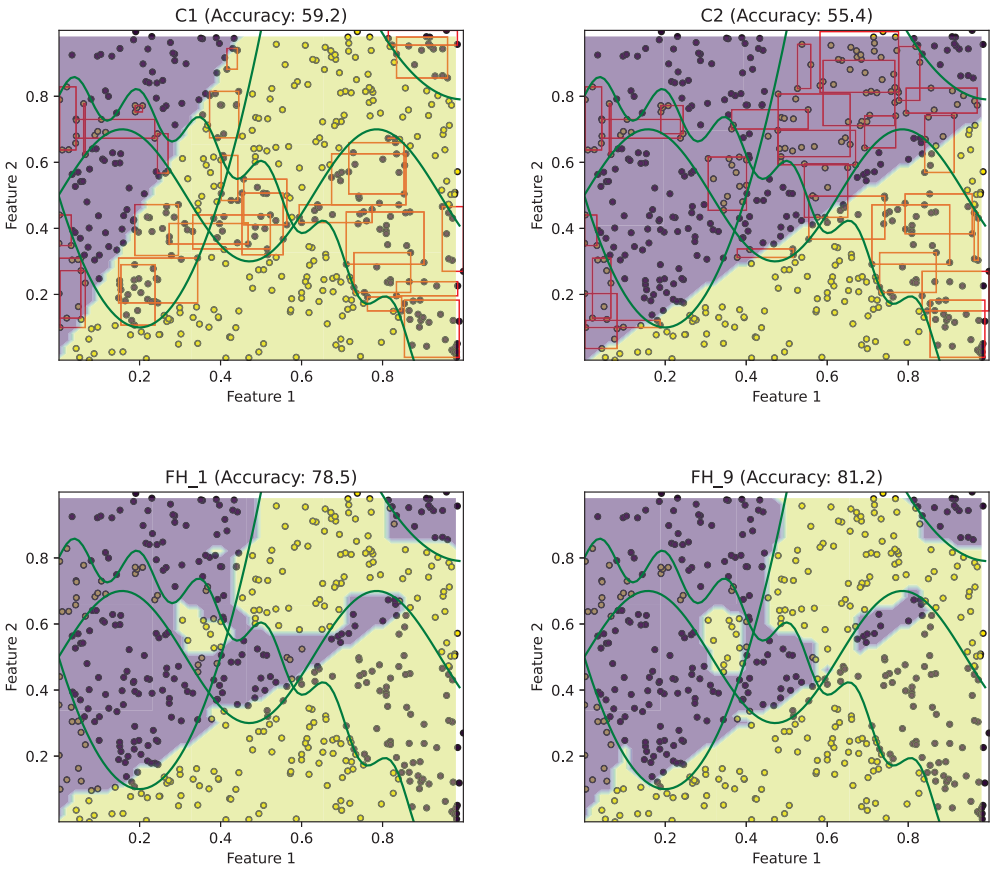


Figure 2.13 Generated hyperboxes using the contraction mechanism on the example of Figure 2.12 (First row) and defined the class domain by the proposed framework without the contraction mechanism (left side of the second row, accuracy = 78.5%) and the same result using the contraction mechanism (right side of the second row, accuracy = 81.2%)

as previously presented in (Davtalab *et al.* (2022)). So that the storage cost does not increase linearly with the dataset size. Moreover, the proposed method estimates the competence of the base experts by computing the membership function to all hyperboxes, which requires a total of e calculation. Our proposed method, thus, has a complexity that increases linearly with the number of hyperboxes generated (O(e)).

Thus, based on the assumption that ($e << N$), the computational complexity of the proposed method is greatly reduced as it does not increase linearly with the dataset size. The whole competence estimation procedure for the proposed method consists of O(e) membership calculations, while for state-of-the-art DES methods such as the META-DES (Cruz *et al.* (2015c)). This principle extends to memory cost considerations as well; while each hyperbox comprises a pair of instances (representing the minimum and maximum corners), the overall count of hyperboxes generated by the system remains significantly lower than the total number of instances present in the dataset. Thus, leading to a much smaller memory footprint.

2.4 Related Work

2.4.1 Dynamic Selection

In DS approaches, classifiers are selected based on their estimated competence level to classify the given unknown sample $\mathbf{x}_q$ is estimated. Then the outputs of selected classifiers are aggregated to label $\mathbf{x}_q$. There are a lot of details within these simple steps that have led to the creation of a wide variety of DS systems (e.g., region of competence definition, selection criterion). Table 2.2 presents state-of-the-art DS approaches and categorizes them according to the main criteria. This section focuses on the main aspects of DS systems and its difference from our proposed framework.

2.4.1.1 Region of Competence (RoC)

Different DS approaches use different ranges of samples to estimate the competence of classifiers. Most state-of-the-art DS approaches define a subset of DSEL samples that are more related to the query sample. For example, KNORA-U (Ko *et al.* (2008)), KNORA-E (Ko *et al.* (2008)), KNOP (Cavalin *et al.* (2013)), META-DES (Cruz *et al.* (2015d)), META-DES-Oracle (Cruz *et al.* (2017a)), DSOC (Brun *et al.* (2016)), DISi (Pereira *et al.* (2018)), DDES (Choi & Lim (2021)), DES-hesitant (Elmi & Eftekhari (2020)), and MLS (Elmi & Eftekhari (2021)), cluster-based DS approaches like (Soares *et al.* (2006)), and Graph-Based approaches (Hou *et al.* (2016); Li *et al.*

Table 2.2 Categorization of state-of-the-art dynamic selection techniques based on the main properties investigated in this paper. OB, TB, and PB stand for Output-Based, Threshold-Based, and Probability-Based selection, respectively. Techniques are ordered based on their publication year

Technique	RoC Definition	Construction Phase	Selection criterion	Selection Type	Year
DCS-Rank (Sabourin *et al.* (1993))	KNN	Generalization	Ranking	OB	1993
OLA (Woods *et al.* (1997))	KNN	Generalization	Accuracy	OB	1997
LCA (Woods *et al.* (1997))	KNN	Generalization	Accuracy	OB	1997
MCB (Giacinto & Roli (2001))	KNN	Generalization	Behavior	OB	2001
MLA (Smits (2002))	KNN	Generalization	Accuracy	OB	2002
DES-Cluster (Soares *et al.* (2006))	Clustering	Training	Accuracy & Diversity	TB	2006
DES-KNN (Soares *et al.* (2006))	KNN	Generalization	Accuracy & Diversity	TB	2006
KNORA-U (Ko *et al.* (2008))	KNN	Generalization	Oracle	TB	2008
KNORA-E (Ko *et al.* (2008))	KNN	Generalization	Oracle	TB	2008
DES-RRC (Woloszynski & Kurzynski (2011))	Potential Function	Generalization	Probabilistic	TB	2011
DES-P (Woloszynski *et al.* (2012))	Potential Function	Generalization	Probabilistic	TB	2012
DES-KL (Woloszynski *et al.* (2012))	Potential Function	Generalization	Probabilistic	TB	2012
KNOP (Cavalin *et al.* (2013))	KNN	Generalization	Behavior	TB	2013
META-DES (Cruz *et al.* (2015d))	KNN	Generalization	Meta-Learning	TB	2015
META-DES.Oracle (Cruz *et al.* (2017a))	KNN	Generalization	Meta-Learning	TB	2017
DSOC (Brun *et al.* (2016))	KNN	Accuracy & Complexity	Generalization	TB	2016
CHADE (Pinto, Soares & Mendes-Moreira (2016a))	Potential Function	Generalization	Meta-Learning	TB	2016
PCC-DES (Narassiguin *et al.* (2017))	Potential Function	Generalization	Meta-Learning	TB	2017
DISi (Pereira *et al.* (2018))	KNN	Generalization	Oracle	TB	2018
DDES (Choi & Lim (2021))	KNN	Generalization	Oracle	TB	2021
DES-hesitant (Elmi & Eftekhari (2020))	KNN	Generalization	Multi criteria	TB	2020
MLS (Elmi & Eftekhari (2021))	Multi technique	Generalization	Multi criteria	TB, OB, PB	2021
DES-ML Elmi *et al.* (2023))	A multi-label classifier	Training	Output of classifier	OB	2023
OLP++ (Souza *et al.* (2023))	Recursive partitioning	Generalization	Output of classifier	OB	2023
FH-DES (proposed)	**Fuzzy Hyperboxes**	**Training**	**Membership Degree**	**TB**	-

(2019)) define a fixed RoC over the query sample that is used to measure the competence of all base classifiers. In some DS methods such as OLA and LCA (Woods *et al.* (1997)), DES-KNN (Soares *et al.* (2006)), KNORA-U and KNORA-E (Ko *et al.* (2008)), RoC defined in feature space.

Other DS techniques work in decision space in which the similarity between the output profile of the query sample and the output profiles of the samples in DSEL is used to calculate the region of competence. Several DS approaches are proposed based on decision space, such as Multiple Classifier Behavior (MCB) (Giacinto & Roli (2001)), k-Nearest Output Profiles (KNOP) (Cavalin *et al.* (2013)) and META-DES (Cruz *et al.* (2015d, 2017a, 2019a)). In a recent work, Souza et al (Souza *et al.* (2023)) proposed a method based on a recursive partitioning algorithm from decision trees to define regions of competencies for high-dimensional datasets.

However, all base classifiers use the same region of competence in these approaches. Moreover, they do not take into account correct-classified (positive) samples and misclassified (negative) samples while defining the region of competence (RoC). While, for the first time in the DS

field, only one group of positive or negative samples is utilized to estimate the competence level of the classifiers (Cruz *et al.* (2018a)). While in the proposed framework, the required regions (competence or incompetence maps) are defined for each individual classifier regarding its misclassified or correct-classified sample. This issue can increase the accuracy of the DS system because a particular competence map is used for each specific classifier to estimate its competence level.

2.4.1.2 Construction Phase

In most state-of-the-art DS approaches methods (Ko *et al.* (2008); Cavalin *et al.* (2013); Cruz *et al.* (2015d); Pereira *et al.* (2018); Choi & Lim (2021); Elmi & Eftekhari (2020, 2021)) the region of competence is defined during the generalization phase. As the majority of DS methods are based on the KNN algorithm for this task, it involves computing distances to all instances in DSEL in order to define this region. From the computational cost point of view, it imposes a significant prediction cost, particularly when dealing with large datasets. Some recent DS approaches like DES-ML (Elmi *et al.* (2023)) and also the proposed framework, the majority of this process is performed during the training phase. In the proposed approach, the competence or incompetence maps are formed during the training phase. This leads to less complexity in the generalization phase compared to current DS approaches. The proposed FH-DES approach also moves the biggest part of its computational cost to the training phase, that is, the hyperbox creation step, requiring only the membership calculation during the inference phase. Thus, we hypothesize it is more efficient for handling large-scale datasets.

2.4.1.3 Selection approach

Regarding the selection approach, dynamic selection techniques are divided into two main groups: Dynamic Classifier Selection (DCS) and Dynamic Ensemble Selection (DES). In DCS techniques, only one is selected to classify the query sample $\mathbf{x}_q$ while DES selects an ensemble of classifiers and then aggregates their outputs. From another perspective and according to the taxonomy proposed in (Elmi & Eftekhari (2021)), DS techniques can be categorized into

three groups: Threshold-Based (TB), Output-Based (OB), and Probability-Based (PB). In the Threshold-Based group, the final classifiers are selected regarding a predefined threshold, KNOP (Cavalin *et al.* (2013)), KNORA-E (Ko *et al.* (2008)), KNORA-U (Ko *et al.* (2008)), and DES-P (Woloszynski *et al.* (2012)) are in this category (Elmi & Eftekhari (2021)). Output-Based approaches select a certain number of the most competent classifiers to form the final ensemble. In this case, if only one classifier is selected, the system will be DCS. MCB (Giacinto & Roli (2001)), OLA (Woods *et al.* (1997)), and LCA (Woods *et al.* (1997)) are some of the DS approaches in this category that selects only the most competent classifier among the pool. In the Probability-Based (PB) approach, the final ensemble of classifiers is selected according to their probability coefficient by a probability tool such as a roulette wheel algorithm. The probability is assigned to each base classifier according to its competence level (Elmi & Eftekhari (2021)).

The proposed approach selects the final ensemble of classifiers based on a threshold-based scheme. However, different than current approaches, the threshold used for classifier selection is adaptative, changing based on each query instance to take into account the estimated competence level of all base classifiers for its classification (Equation 2.9). As such, we expect our method to be more robust to handle the particularities of distinct local regions.

Table 2.3 Categorization of state-of-the-art fuzzy min-max network approaches and the machine learning context they were employed. TB, OB and PB refer to threshold-based, output-based, and probability-based selection schemes, respectively. Methods are ordered based on their publication year

Technique	Use contraction	Membership Function	Context	Year
GFMM (Gabrys & Bargiela (2000))	Yes	Gabrys's Membership Function	Classification	2000
SFMM (Likas (2001))	Yes	Simpson's Membership Function	Reinforcement Learning	2001
DCFMN (Zhang *et al.* (2011))	No	Data Core Membership Function	Classification	2011
M-FMCN (Davtalab, Parchami, Dezfoulian, Mansourizade & Akhtar (2012))	No	Gabrys's Membership Function	Classification	2012
MLF (Davtalab *et al.* (2013))	No	Gabrys's Membership Function	Classification	2014
EFMN (Mohammed & Lim (2014))	Yes	Simpson's Membership Function	Classification	2015
FMM-ETC (Seera, Randhawa & Lim (2018))	Yes	Simpson's Membership Function	Clustering	2018
FMM-BSO (Pourpanah *et al.* (2019))	Yes	Simpson's Membership Function	Rule Extraction	2019
IOL-GFMM (Khuat *et al.* (2020))	No	Gabrys's Membership Function	Classification	2020
IOL_GFMM_HB (Kenger & Ozceylan (2023))	No	Gabrys's Membership Function	Classification	2023
FH-DES (proposed)	**yes**	**Smooth Borders Membership**	**Classifier selection**	-

2.4.2 Fuzzy Min-Max approaches

Fuzzy hyperboxes were introduced to represent domains of classes and clusters in fuzzy min-max neural networks (Simpson (1992); Simpson & Jahns (1993)) and, due to their flexibility, have been used in different machine learning applications contexts (Kenger & Özceylan (2023)) such as clustering, classification, and rule extraction. Table 2.3 reports a summary of hyperbox-based approaches.

Many hyperbox-based approaches have been introduced in recent years, which can be categorized into two primary categories based on whether they utilize the contraction process or not (Khuat *et al.* (2021b)). Most of them, like the original FMM (Simpson (1992)), use the contraction process to address the overlapped areas between the hyperboxes that belong to different classes. These approaches try to improve accuracy by reducing the size of hyperboxes involved in the overlaps. General Fuzzy Min-Max neural network (GFMM) (Gabrys & Bargiela (2000)), Enhanced Fuzzy Min-Max neural network (EFMN) (Mohammed & Lim (2014, 2017)), and improved Fuzzy Min-Max neural network using Ensemble of Clustering Trees (FMM-ECT) (Seera *et al.* (2018)) are some of the FMM approaches that use the contraction process to handle the overlaps. On the other hand, some hyperbox-based approaches use special nodes or mechanisms to handle the overlaps between the hyperboxes. For example, Data-Core-based Fuzzy Min–Max neural network (DCFMN) (Zhang *et al.* (2011)) uses a mechanism to obtain the geometric center of the hyperbox and the data core, modified Fuzzy Min–Max classifier using compensatory neurons (M-FMCN) (Davtalab *et al.* (2012)) uses compensatory neurons, and Multi-Level Fuzzy Min–Max neural network (MLF) (Davtalab *et al.* (2013)) utilizes a multi-layer structure as an alternative for the contraction process and handling the overlaps of each layer by the next layer. In addition, Improved Online Learning algorithm for General Fuzzy Min-Max neural network (IOL-GFMM) (Khuat *et al.* (2020)) is an improved version of GFMM that uses a novel overlap pre-checking process to avoid any overlapped areas between hyperboxes of different classes. The same process is utilized in the Hybrid Model of improved online learning FMM (IOLGFMM_HB) (Kenger & Ozceylan (2023)) this algorithm uses a mixed-integer linear programming (MILP) model to improve the efficiency of hyperboxes generated by the

IOL-GFMM algorithm. The overlap pre-checking process is an efficient method to prevent generating overlap regions. However, the number of hyperboxes increases using this process. To address this drawback in the proposed approach, we use the learning sensitivity parameter (λ) alongside the pre-check process that controls the trade-off between accuracy and performance.

The membership function is another important element of hyperbox-based approaches. Simpson has proposed two membership functions for classification (Simpson (1992)) and clustering (Simpson & Jahns (1993)) applications. However, the membership values in these functions fail to decrease gradually as the input data moves away from the hyperbox, which could result in decreased accuracy. To address this issue, Gabrys and Bargiela have proposed the membership function discussed in Section 1.3. As mentioned earlier, this function possesses sharp corners in its membership levels (Figure 2.5). Special membership functions are used in hyperbox-based approaches that do not use the contraction process. For example, DCFMN (Zhang *et al.* (2011)) uses a novel membership function regarding the geometric center of the hyperbox and the data core. However, such memberships endure a high computational overhead to the system and are not suitable for handling large volumes of data. Therefore, in the proposed approach, we propose a novel membership function with smooth corners (2.9), which is simple and can also mitigate the sharp corners problem from previous approaches.

It is important to mention that there exist other neural network approaches based on fuzzy operators, such as Fuzzy-Artmap (Carpenter *et al.* (1992)), which use prototype nodes. Each node consists of a recognition field that defines the area of the feature space it covers. These prototypes are used to represent different categories and or classes. Fuzzy operators are then used to calculate the match between the prototype nodes and the input samples. Another alternative is using hyperspheres instead of hyperboxes to delimit regions in the input space representing different classes (Kulkarni, Doye & Sontakke (2002); Mahindrakar & Kulkarni (2022)). Nevertheless, the simple geometric structure of hyperboxes based on Min-Max points has some advantages since they are not constrained to be centered around a specific point which leads to more flexibility in representing non-linear and irregular decision boundaries.

Furthermore, fuzzy hyperboxes tend to have a faster training process, especially in cases where complex and overlapping data distributions are involved.

2.5 Experimental protocol

2.5.1 Datasets

In our experimental study, 42 different datasets in a wide variety of areas were used. They were collected from OpenML (Van Rijn *et al.* (2013)), and UCI (Asuncion & Newman (2007)) repositories. Table 2.4 presents the utilized datasets and their specifications. We considered three groups of datasets in order to evaluate different aspects of the proposed approach. The first group consisting of eight datasets was used to tune the proposed approach's hyperparameters following the same hyperparameter tuning methodology as Cruz et al. (Cruz *et al.* (2015d)) (rows 1 to 8). The next 30 datasets (rows 9 to 38) were used in our comparative study to evaluate the FH-DES framework over small and medium-scale problems and compare its performance against the state-of-the-art DES methods. The last 4 datasets (rows 39 to 42) are large-scale datasets utilized to evaluate the scalability of the proposed approaches and incremental learning capabilities. subsectionExperimental setup

2.5.2 Experimental setup

In all experiments, each dataset is randomly divided into three groups. The first group consists of 50% of the data that are used for training, the next 25% for the dynamic selection dataset (DSEL), and the remaining 25% are used for testing. The division is performed by maintaining the prior probabilities of each class. Furthermore, the datasets were normalized using the Min-Max scaling technique (de Amorim, Cavalcanti & Cruz (2023)) with ranges between 0 and 1.

Similar to the experimental protocol used in (Cruz *et al.* (2018a)) and (Davtalab *et al.* (2022)), the pool of classifiers contained 100 Perceptron classifiers that were calibrated using isotonic method (Allikivi & Kull (2019)) by the validation data. These classifiers were generated using

the bagging technique (Breiman (1996)) by the Scikit-learn library (version: 1.0.1). The pool was fixed for all techniques to ensure a fair comparison. Furthermore, each experiment was carried out using 20 replications to obtain the mean and standard deviation results for each dataset.

Table 2.4 Datasets considered in this work and their main characteristics. Rows 1 to 8 represent datasets used for tuning the technique. The 30 datasets used to evaluate the method's performance on small to medium-scale problems are presented in rows 9 to 38. The last group (rows 39 to 42) was used in the large-scale experiment

#	Database	Instances	Features	#	Database	Instances	Features
1	Adult	690	14	22	Laryngeal1	213	16
2	Audit2	776	17	23	Laryngeal3	353	16
3	CTG	2126	21	24	Lithuanian	600	2
4	Cardiotocography	2126	21	25	Liver	345	6
5	Chess	3196	36	26	Mammographic	830	5
6	Credit-screening	690	15	27	Monk2	432	6
7	P2	1000	2	28	Phoneme	5404	5
8	Transfusion	748	4	29	Pima	768	8
9	Audit	771	26	30	Sonar	208	60
10	Banana	1000	2	31	Statlog	1000	20
11	Banknote	1372	4	32	Steel	1941	27
12	Blood	748	4	33	Thyroid	692	16
13	Breast	569	30	34	Vehicle	846	18
14	Car	1728	6	35	Vertebral	310	6
15	Datauser	403	5	36	Voice3	238	10
16	Faults	1941	27	37	Weaning	302	17
17	German	1000	24	38	Wine	178	13
18	Haberman	306	3	39	Sensor	919438	11
19	Heart	270	13	40	ArData	900000	5
20	ILPD	583	10	41	Incidents	2215023	9
21	Ionosphere	351	34	42	Agrawal	1000000	10

The performance of the proposed approach was compared with the state-of-the-art DS techniques, which have been selected based on their performance according to a recent experimental study (Cruz *et al.* (2018a)), different selection criteria (e.g., ranking, behavior, oracle, meta-learning), and also regarding the availability of their implementation. These approaches including KNORA-

U (Ko *et al.* (2008)), KNORA-E (Ko *et al.* (2008)), MCB (Giacinto & Roli (2001)), DES-KNN (Woloszynski *et al.* (2012)), OLA (Woods *et al.* (1997)), Rank (Sabourin *et al.* (1993)), KNOP (Cavalin *et al.* (2013)), and META-DES (Cruz *et al.* (2015d)). All these methods are based on the KNN algorithm to define the region of competence and use different selection criteria to select the final ensemble of classifiers (Cruz, Hafemann, Sabourin & Cavalcanti (2020)). In addition, the Majority Voting (MV), Single Best (SB), and GFMM algorithm are considered as lower-bound DS baselines. The Oracle concept (Kuncheva (2002)) as an upper-bound for DS methods in our experiments. Results of the GFMM algorithm (that uses the hyperboxes for classification directly) are reported in order to investigate whether using the hyperboxes in the DS context is effective compared to its traditional usage.

To evaluate the strategies introduced in subsection 2.3.2, we consider ten configurations of the proposed framework, as listed in Table 2.5. As discussed in Section 2.3, these variants can be set up based on misclassified or correctly classified samples. In the first strategy, variants utilize the incompetence map to estimate the competence of individual classifiers. While in the second strategy, these variants are based on correct classified samples and use a competence map to select the best ensemble of classifiers. When a variant is based on misclassified samples, it is called *negative variant* and is shown by letter *M* e.g., FH_5-M. On the other hand, the letter *C* shows a *positive variants*, which are based on correct classified samples. For example, FH_5-C is the fifth variant in which the hyperboxes are built based on correct classified samples of base classifies. In addition, FH_1-M and FH_1-C represent FH_DES-M and FH_DES-C, respectively, introduced in our preliminary study (Davtalab *et al.* (2022)).

The proposed framework has two main hyperparameters: (μ), which defines a threshold to select the top base classifiers, and Theta (θ), which defines the maximum size of hyperboxes. However, in variants in which the overlap pre-check process is used (FH_2, FH_4, FH_9, and FH_10), the learning sensitivity parameter or Lambda (λ) is utilized instead of θ. As explained in subsection 2.3.2, λ controls the computational complexity of the model and makes a trade-off between accuracy and performance. In this paper, we set λ to 1 to achieve maximum accuracy in small and medium problems. In contrast, in large-scale problems where computational complexity is

Table 2.5 Different variants of the proposed framework

	Selection Strategy		Expansion Criterion		Contraction Strategy	
	Nearest_hyperbox	All_hyperboxes	Theta	Overlap_pre-check	Instance	Hyperbox
FH_1	✓	✗	✓	✗	-	-
FH_2	✓	✗	✗	✓	✓	✗
FH_3	✗	✓	✓	✗	-	-
FH_4	✗	✓	✗	✓	✓	✗
FH_5	✓	✗	✓	✗	✓	✗
FH_6	✓	✗	✓	✗	✗	✓
FH_7	✗	✓	✓	✗	✓	✗
FH_8	✗	✓	✓	✗	✗	✓
FH_9	✗	✓	✓	✓	✓	✗
FH_10	✗	✓	✓	✓	✗	✓

important, we vary the λ value to study how it affects the number of hyperboxes generated in the system and choose a value with a good accuracy vs computational complexity trade-off.

All hyperparameters are set in the range of 0 to 1. To tune these hyperparameters, we use a process similar to that used by Cruz et al. (Cruz *et al.* (2015d)). Thus, the first eight datasets from Table 2.4 (rows 1 to 8), which have not been used in the comparative study, were used during this process to avoid biased estimation. Notably, variants FH_1 and FH_3 do not utilize the contraction process, so they do not need to select any contraction strategy. The tuning experiments were carried out on all proposed variants using the tuning datasets. The best hyperparameter values obtained from this experiment are reported in the Appendix (Subsection 1).

The hyperparameters of the state-of-the-art DES methods are set regarding the original published papers and reported values in (Cruz *et al.* (2018a)) (Table 2.6). Furthermore, all DES methods are publicly available on the DESlib (Cruz *et al.* (2020)) library on GitHub[2].

2.6 Results

In this section, we address the four research questions in different subsections. In subsection 2.6.1, we answer the research question "**RQ1 - Does modeling the misclassified samples lead to higher accuracy and less computational complexity compared to modeling the correctly**

[2] https://github.com/scikit-learn-contrib/DESlib

Table 2.6 Used parameters and their values in experiments

Method	Hyperparameter Name	Value
Perceptron	Maximum iteration	100
	Tolerance	$10e-3$
	Alpha	0.001
	Calibration method	isotonic
Bagging	Number of estimators	100
	bootstrap	True
	max-samples	1.0
KNN-based DS	K	7
META-DES	K_p	5
	h_c	80%

classified ones?". All FH-DES variants are evaluated in this subsection, and their performances are analyzed based on classification accuracy and computational complexity over the 30 datasets. This subsection also discusses the impact of different strategies for expansion and contraction presented in this work. In Subsection 2.6.2, we address the question "**RQ2 - Can the use of fuzzy hyperboxes outperform state-of-the-art DS approaches?**". To do so, the accuracy of the proposed approaches on 30 datasets is compared to the state-of-the-art DS approaches using statistical analyses. Then in Subsection 2.6.3, we address the research questions "**RQ3 - Do hyperboxes decrease the complexity of dynamic selection approaches?**" and "**RQ4 - Can using a contraction process increase the accuracy of the proposed framework in large-scale problems?**". In This subsection, the large-scale datasets listed in Table 2.4 are used to evaluate the time and space complexity of proposed variants compared to KNN-based DS approaches and how the proposed framework behaves in an incremental learning scenario.

2.6.1 Modeling Misclassified vs Correct-classified samples

In the first step, we run all variants in Table 2.5 based on misclassified and correct classified samples over the 30 datasets listed in Table 2.4 (rows 9 to 38). The specific result obtained from all variants per dataset is reported in 2. According to the results of this experiment, variants FH_2-C, FH_4-C, and FH_9-C have the best ranks among positive variants. Similarly, the FH_2-M, FH_4-M, and FH_9-M variants have the best rank and accuracy among negative variants. Variant FH_4-M obtained the highest accuracy among all negative and positive variants.

All these approaches use the overlap pre-checking process to confine the expansion candidate hyperboxes. In addition, FH_10-C and FH_10-M that utilize this technique also have good accuracy. Therefore, we can state that the overlap pre-checking process can improve accuracy in the FH-DES approach. This process prevents the growth of hyperboxes in the unauthorized region compared to other variants. In addition, the pre-check approach cares about all previous samples that have been learned before. All of them are kept inside the hyperboxes during the training process. However, in the traditional contraction approach, some of the learned samples are more likely to be located outside the hyperboxes at the end of the learning phase. These two issues can affect the efficiency of the FH-DES.

To compare each variant's positive and negative versions, we performed a pairwise comparison between each negative variant and its equivalent positive variants using Sign test (Demšar (2006)). This test has two hypotheses that include H_0 and H_1. H_0 is the null hypothesis which means that both techniques obtained statistically equivalent results, and rejection of H_0 means the classification performance obtained by the corresponding DS technique is significantly better than the compared technique. The number of wins, ties, and losses are computed for each technique compared to the baseline. The significance level of this test is determined by the predefined parameter α, which in this paper is set $\alpha = 0.05$ to have 95% confidence. If the number of wins is greater than or equal to a critical value, indicated by n_c, the null hypothesis H_0 is rejected. The critical value is computed using equation 2.12:

$$n_c = \frac{n_{exp}}{2} + z_\alpha \frac{\sqrt{n_{exp}}}{2} \tag{2.12}$$

Where n_{exp} is the total number of experiments and $z_\alpha = 1.645$, for a significance level of $\alpha = 0.05$ (Cruz *et al.* (2017b)). In this test, we have 30 experiments (datasets). Therefore, $n_{exp} = 30$, and the critical value for this number of experiments is $n_c = 19.50$. The average accuracy of each variant's negative and positive versions is calculated on the 30 datasets and compared to each other. Figure 2.14 shows the result of the Sign test. For example, the first

column of this chart shows that the negative version of FH_1 (called FH_1-M) on 22 out of 30 datasets has higher accuracy than FH_1-C.

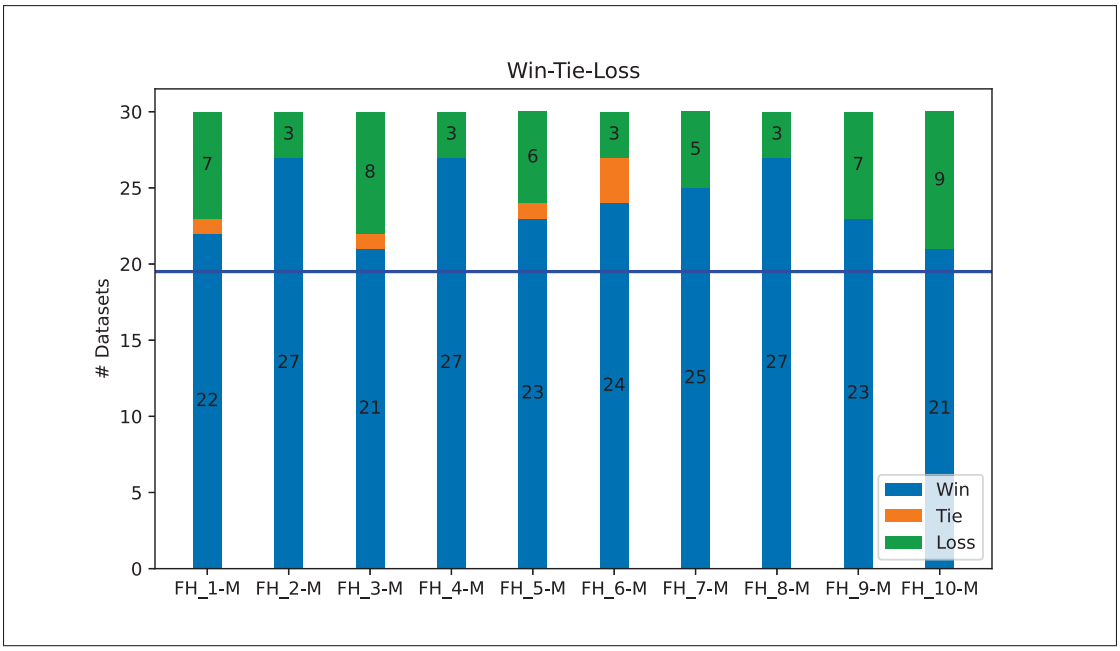


Figure 2.14 Comparing the positive variants and negative variants using the Win-Tie-Loss test on the 30 small datasets. The blue horizontal blue line illustrates the critical value $n_c = 19.5$

The results of the Sign test indicate that negative variants obtain higher accuracy compared to the equivalent positive variants. To analyze the trade-off between performance and the number of hyperboxes, the average accuracy rank and number of hyperboxes generated on the 30 datasets were calculated are presented in Figure 2.15. The higher the accuracy of the algorithm and the fewer hyperboxes it produces (located on the lower left side of the figure), the better the algorithm. The results, shown in Figure 2.15, reveal that FH_4-M has the best rank among the different variants of FH-DES in terms of average accuracy and number of hyperboxes generated, and was selected as the best FH-DES. Its performance is evaluated against other DS approaches later.

This experiment also revealed that not only the accuracy of negative variants is higher than their corresponding positive variant, but most of them also generate fewer hyperboxes than positive

versions. Therefore, answering **RQ1, using hyperboxes generated based on misclassified samples leads to more accurate and lower computational complexity dynamic selection systems.**

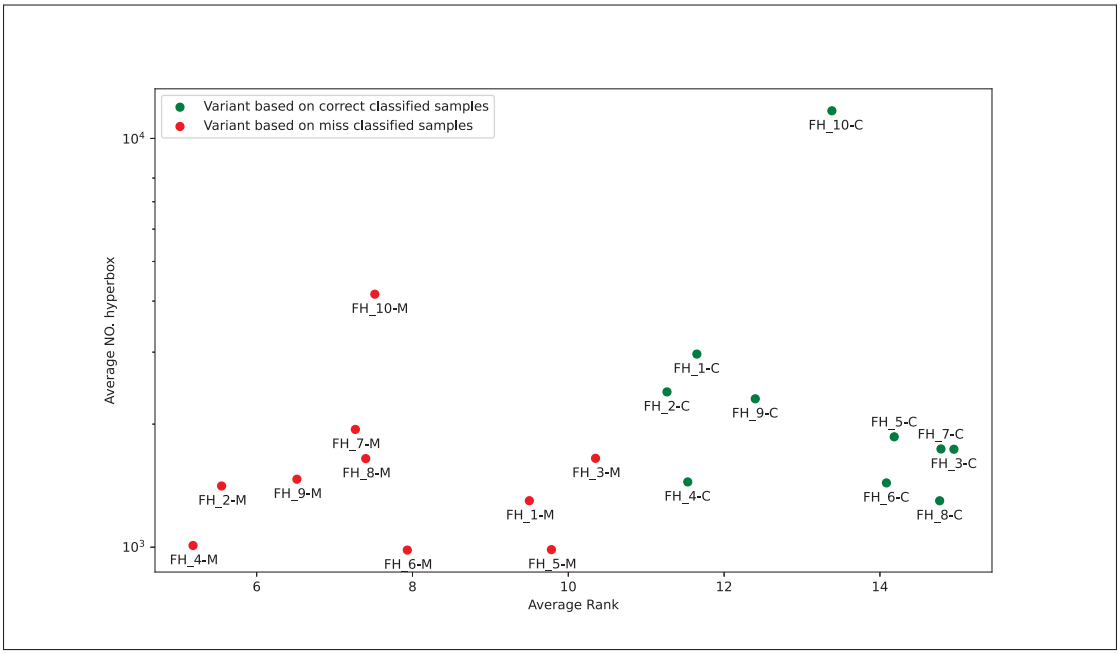


Figure 2.15 Average Ranking of different variants versus the number of generated hyperboxes

2.6.2 State-of-the-art comparison

In Table 2.7, the results of the selected variant (FH_4-M) alongside the baseline methods including *majority voting (MV)*, *single best* (SB), and *Oracle* are reported. The Oracle approach (Kuncheva (2002)) is a conceptual method that selects the base classifier that correctly labels the query sample (if it exists). This concept is known as an upper limit to dynamic selection approaches. Therefore, we do not consider this approach in the ranking of the approaches in Table 2.7. In addition, the methods Majority-Voting (MV) and Single-Best (SB) are considered lower-limit approaches. Furthermore, we have used *general fuzzy min-max neural network classifier (GFMM)* (Gabrys & Bargiela (2000)) as a pure fuzzy hyperbox-based method to show the difference in accuracy with the case where it is used as a dynamic selection method.

Table 2.7 Average accuracy and standard deviation of selected variants in comparison with baseline methods

DataSets	Majority Voting (MV)	Single Best (SB)	GFMM	FH_4-M	Oracle
Audit	96.35(1.46)	97.05(1.17)	95.91(1.38)	**97.07(1.21)**	99.56(0.41)
Banana	86.26(1.7)	89.9(1.81)	**89.98(5.49)**	89.76(2.24)	92.82(1.84)
Banknote	98.72(0.61)	**99.31(0.43)**	95.57(2.4)	99.26(0.44)	99.84(0.27)
Blood	**78.42(1.52)**	77.27(1.13)	70.29(6.2)	77.99(1.54)	89.44(2.26)
Breast	96.78(1.42)	96.75(1.39)	95.0(2.11)	**96.82(1.3)**	99.02(0.64)
Car	70.34(1.07)	74.42(1.48)	77.62(1.61)	**79.46(2.0)**	83.44(0.96)
Datausermodeling	86.83(3.26)	88.47(2.53)	64.36(4.12)	**88.76(2.45)**	99.31(1.0)
Faults	68.73(2.2)	69.28(2.31)	67.21(2.04)	**69.6(2.28)**	91.55(1.06)
German	75.0(2.01)	75.12(1.93)	68.78(2.91)	**75.42(2.33)**	94.68(1.22)
Haberman	75.13(2.44)	75.19(2.43)	66.82(6.69)	**75.39(2.77)**	92.01(2.4)
Heart	**83.31(3.84)**	83.09(4.2)	76.91(4.16)	**83.31(4.24)**	97.43(1.85)
ILPD	71.99(2.64)	**72.6(2.6)**	65.89(4.2)	72.4(2.75)	96.2(1.41)
Ionosphere	87.27(2.2)	87.73(2.66)	**91.82(2.14)**	88.24(2.01)	97.84(1.52)
Laryngeal1	82.5(4.28)	82.69(4.15)	78.89(4.66)	**82.96(4.21)**	96.02(3.0)
Laryngeal3	70.22(3.3)	70.51(3.41)	66.29(4.31)	**71.12(3.39)**	90.06(2.57)
Lithuanian	84.47(1.84)	78.5(10.32)	**91.23(3.69)**	89.17(2.59)	93.67(1.86)
Liver	67.64(4.37)	67.36(4.8)	57.18(4.62)	**68.97(4.48)**	97.82(1.58)
Mammographic	**80.96(2.61)**	70.22(7.11)	72.04(5.76)	79.74(2.67)	92.14(2.85)
Monk2	78.61(3.39)	80.14(3.27)	62.31(3.47)	**87.59(3.37)**	97.27(1.11)
Phoneme	76.72(0.9)	74.93(0.92)	**78.24(1.28)**	77.88(1.04)	88.41(1.83)
Pima	77.06(1.93)	76.64(2.03)	69.77(2.91)	**77.08(2.02)**	93.46(1.34)
Sonar	77.31(5.66)	79.04(6.23)	**86.54(4.9)**	78.08(6.0)	98.46(1.88)
Statlog	75.28(2.05)	75.16(2.26)	69.88(2.36)	**75.48(2.25)**	93.94(1.62)
Steel	69.54(1.66)	70.08(1.89)	66.75(2.47)	**70.53(2.13)**	91.64(1.3)
Thyroid	95.98(1.35)	95.9(1.06)	95.26(1.42)	**96.18(1.1)**	98.64(0.74)
Vehicle	75.38(2.5)	75.4(2.27)	67.5(2.95)	**75.83(1.78)**	96.75(1.1)
Vertebral	83.93(4.1)	83.6(3.4)	77.93(3.94)	**84.4(3.96)**	96.67(2.37)
Voice3	**78.5(3.07)**	78.0(3.48)	70.08(4.9)	**78.5(3.02)**	92.67(1.53)
Weaning	80.46(4.15)	**81.12(4.12)**	77.76(3.0)	81.05(3.96)	97.5(2.12)
Wine	98.22(1.51)	**98.44(1.59)**	96.0(3.03)	**98.44(1.59)**	99.89(0.48)
Ave Rank	2.7	2.45	3.4	**1.45**	-
p-value	**<0.0001**	**<0.0001**	**<0.0001**	**-**	-

The Wilcoxon signed pair rank test, as suggested by (Stapor, Ksieniewicz, Garcia & Wozniak (2021)), was conducted to evaluate whether the results between methods are statistically significant. Since our goal is to compare the performance of the proposed FH_4-M to the baseline, it was considered the control method in the pairwise comparison. The results of the Wilcoxon test (p-values) are presented in the last row of Table 2.7. In all cases, the p-value obtained is lower than 0.0001, indicating that the proposed method obtained statistically superior results compared to all baseline methods.

In addition, we conducted a pairwise comparison between FH_4-M and the baseline approaches using the Sign test (Demšar (2006); Cruz *et al.* (2017a)) to compare the methods from a different perspective. The results are illustrated in Figure 2.16. Each column of this figure demonstrates the number of wins, ties, and losses of FH_4-M against each baseline approach. For instance,

the right bar in the plot shows that FH_4-M achieved 25 wins and 5 losses out of 30 datasets against the GFMM. This analysis further demonstrates that using hyperboxes to select the best ensemble of classifiers in the DS context yields higher average accuracy than when used in the GFMM algorithm as a classification method. Therefore, this study suggests that FMM can be used as a method to estimate the classifier's competence, with promising results. Similar results are observed when the proposed method is compared to standard baseline methods in the ensemble literature.

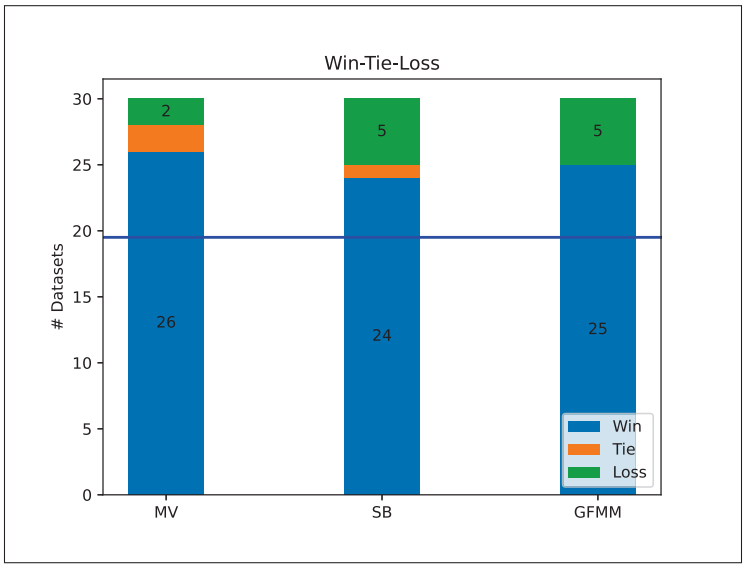


Figure 2.16 Pairwise comparison between the FH_4-M and other DS methods (n_c = 19.5). The number of wins, ties, and losses of FH_4-M are highlighted in blue, orange, and green, respectively

Next, we compare the results of our proposed FH-DES with the state-of-the-art DS approaches. The average accuracy and standard deviation obtained by each approach per dataset is presented in Table 2.8. Our FH_4-M attained the highest average accuracy in nine of the thirty datasets. Additionally, this variant had the best rank and the highest average accuracy among all thirty datasets, when compared to the other DS methods. In addition, the results of the Wilcoxon test, highlighted in the last row of Table 2.8 demonstrate that the proposed method significantly outperforms a few DES techniques (KNORA-U, MCB, KNOP, RANK, OLA) while obtaining

statistically equivalent results to those of the META-DES, DESKNN, and KNORA-E, with an $\alpha = 0.05$.

Table 2.8 Average accuracy and standard deviation of the proposed method and other DS approaches

DataSets	KNORA-U	KNORA-E	MCB	DESKNN	OLA	RANK	KNOP	META-DES	FH_4-M
Audit	96.45(1.5)	97.2(1.13)	97.12(1.25)	96.99(1.17)	**97.44(1.45)**	97.1(1.31)	96.55(1.43)	96.87(1.15)	97.07(1.21)
Banana	87.6(2.01)	91.06(2.39)	**91.22(2.24)**	88.56(2.16)	91.08(1.85)	90.82(1.89)	86.66(1.86)	88.44(2.04)	89.76(2.24)
Banknote	98.86(0.51)	**99.42(0.54)**	99.31(0.65)	99.18(0.49)	99.34(0.56)	99.34(0.56)	98.85(0.54)	99.1(0.62)	99.26(0.44)
Blood	**78.4(1.55)**	76.23(2.35)	77.38(2.25)	77.78(1.7)	77.38(2.24)	76.15(2.38)	78.24(1.49)	77.97(1.7)	77.99(1.54)
Breast	96.82(1.24)	**96.85(1.49)**	96.57(1.56)	96.82(1.46)	96.71(1.78)	96.71(1.78)	**96.85(1.2)**	96.68(1.21)	96.82(1.3)
Car	71.31(1.17)	73.82(1.2)	73.61(1.18)	72.15(1.21)	73.82(1.2)	73.72(1.13)	71.25(0.99)	73.41(1.52)	**79.46(2.0)**
Datausermodeling	88.17(2.49)	**91.34(2.58)**	88.96(2.33)	90.25(2.29)	89.31(3.51)	89.06(3.56)	88.51(2.65)	90.69(2.29)	88.76(2.45)
Faults	69.61(2.27)	69.69(1.4)	68.55(1.72)	**70.5(1.81)**	68.85(1.58)	68.83(1.35)	69.74(2.48)	69.96(1.93)	69.6(2.28)
German	74.94(1.98)	74.14(2.89)	73.52(2.23)	75.14(2.38)	73.44(2.73)	73.42(2.4)	75.22(1.85)	74.82(2.19)	**75.42(2.33)**
Haberman	73.9(3.07)	71.36(4.45)	70.71(4.65)	73.7(3.69)	72.34(4.52)	71.62(4.67)	73.77(2.71)	71.88(4.86)	**75.39(2.77)**
Heart	83.24(3.87)	82.79(4.26)	82.28(3.49)	**83.46(4.28)**	82.28(5.06)	81.69(4.98)	**83.46(3.75)**	82.72(4.16)	83.31(4.24)
ILPD	72.19(2.55)	70.68(3.04)	70.82(2.57)	71.4(1.72)	70.68(2.97)	70.0(2.8)	71.95(2.04)	70.89(2.34)	**72.4(2.75)**
Ionosphere	87.78(2.43)	**88.98(1.87)**	87.16(2.57)	88.52(2.18)	87.39(2.18)	87.5(2.27)	88.01(2.23)	88.01(2.34)	88.24(2.01)
Laryngeal1	82.59(3.72)	82.78(4.19)	81.85(6.07)	82.41(4.16)	80.74(4.32)	81.76(3.76)	82.41(3.99)	82.31(4.08)	**82.96(4.21)**
Laryngeal3	70.84(3.56)	71.4(3.54)	70.34(3.83)	71.57(3.14)	**71.74(4.77)**	71.29(4.31)	70.73(3.38)	70.56(4.65)	71.12(3.39)
Lithuanian	87.1(1.8)	90.97(2.5)	90.83(2.59)	88.7(2.37)	**91.13(2.09)**	90.77(2.43)	86.33(1.95)	88.3(2.14)	89.17(2.59)
Liver	68.62(4.36)	68.16(5.06)	67.64(5.36)	**70.92(4.47)**	68.68(4.98)	67.76(5.26)	69.08(4.25)	68.33(4.23)	68.97(4.48)
Mammographic	**81.11(2.65)**	80.1(3.21)	80.38(3.0)	80.82(2.94)	80.41(2.68)	80.07(3.11)	80.94(2.8)	80.82(3.0)	79.74(2.67)
Monk2	79.4(3.38)	87.04(2.72)	85.74(3.74)	83.29(3.38)	84.54(3.76)	85.28(3.26)	81.3(3.97)	**88.52(4.02)**	87.59(3.37)
Phoneme	77.49(0.83)	**80.05(1.0)**	79.22(1.2)	77.84(0.89)	79.33(1.16)	79.82(1.1)	77.46(0.87)	79.69(1.05)	77.88(1.04)
Pima	77.06(2.19)	76.02(2.01)	74.71(2.48)	76.59(2.48)	75.6(2.48)	75.73(2.51)	**77.29(2.21)**	76.98(2.47)	77.08(2.02)
Sonar	77.4(5.8)	78.85(5.67)	76.92(4.94)	78.27(5.06)	76.63(5.55)	77.21(6.15)	77.4(5.64)	**80.48(6.39)**	78.08(6.0)
Statlog	75.3(2.31)	74.98(2.54)	73.88(2.64)	75.32(2.31)	74.42(2.31)	74.46(2.61)	75.24(2.39)	75.18(2.2)	**75.48(2.25)**
Steel	70.48(1.78)	71.07(1.75)	70.26(1.71)	**71.62(1.91)**	70.52(1.71)	70.13(1.86)	70.61(1.89)	70.78(2.15)	70.53(2.13)
Thyroid	95.98(1.31)	95.92(1.74)	95.78(1.45)	95.75(1.4)	96.16(1.59)	96.01(1.92)	95.95(1.33)	96.04(1.23)	**96.18(1.1)**
Vehicle	75.24(1.77)	75.66(2.36)	73.99(2.27)	75.5(2.09)	74.46(2.26)	74.86(2.41)	75.17(1.94)	74.72(2.04)	**75.83(1.78)**
Vertebral	83.46(5.44)	84.36(3.92)	84.36(3.26)	**85.26(4.99)**	84.04(3.59)	83.97(3.61)	83.91(5.22)	84.1(5.07)	84.4(3.96)
Voice3	**78.75(2.63)**	77.42(3.89)	76.75(3.51)	78.25(2.86)	77.75(3.51)	76.83(3.57)	78.5(2.52)	77.33(3.55)	78.5(3.02)
Weaning	80.86(3.94)	81.45(4.48)	81.32(4.59)	**81.84(4.15)**	80.79(4.98)	81.12(3.8)	80.66(4.14)	80.33(4.35)	81.05(3.96)
Wine	98.0(1.71)	98.11(1.45)	97.11(2.23)	98.33(1.55)	96.33(2.36)	96.33(2.36)	97.89(1.92)	98.11(1.76)	**98.44(1.59)**
Ave Rank	5.2 8	3.85	6.63	3.8	5.5	6.35	5.22	5.03	**3.33**
p-value	**0.0002**	0.4973	**0.0029**	0.4540	**0.0096**	**0.0027**	**0.0006**	0.0641	-

We also present the results of a Sign test between the proposed method and the state-of-the-art ones to provide a more comprehensive understanding of the underlying results (Figure 2.17). The test results show that the variant FH_4-M statistically outperforms the KNORA-U, MBC, OLA, RANK, KNOP, and META-DES approaches. Additionally, FH_4-M had more wins against KNORA-E and DESKNN, though there was no statistical significance according to this test. This confirms the hypothesis that using fuzzy hyperboxes and learning from the base classifiers' mistakes can mitigate the problems associated with defining a proper region of competence in dynamic selection algorithms, leading to higher overall average accuracy. Given these results, we can answer RQ2: yes, FH-DES can outperform state-of-the-art DES techniques.

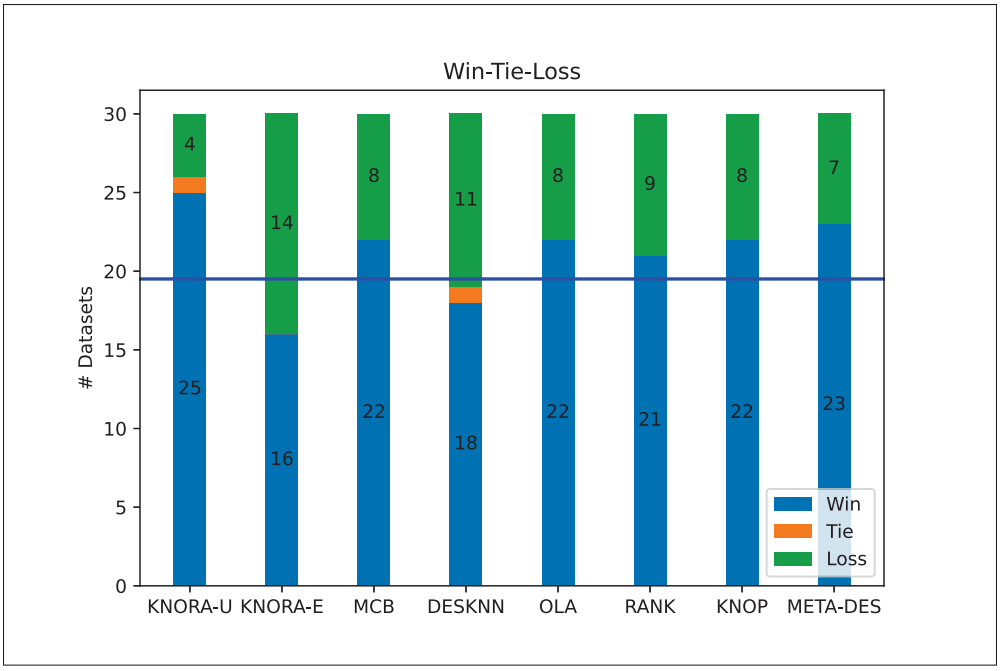


Figure 2.17 Pairwise comparison between the proposed (FH_4-M variant) approach against the state-of-the-art DS approaches. The blue horizontal blue line illustrates the critical value, $n_c = 19.5$

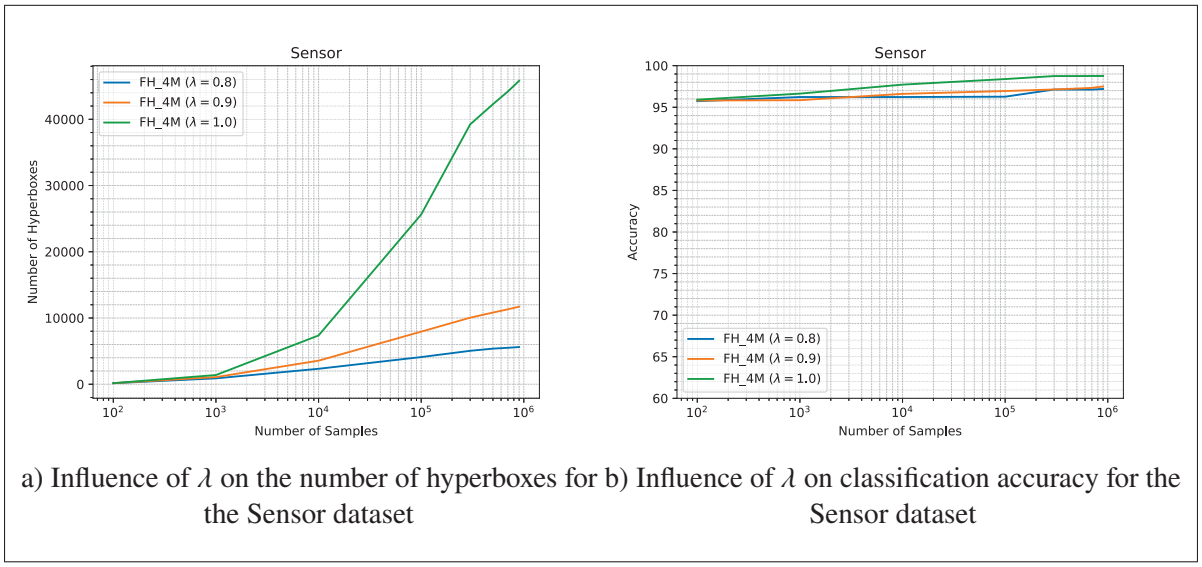
2.6.3 Large-Scale Simulations

To validate the hypothesis that our proposed FH-DES is more computationally efficient than state-of-the-art DS techniques, this section evaluates the efficiency of the selected variant (FH_4-M) using large datasets. In particular, as shown in Section 2.3.5, the storage and computational complexity of the proposed FH-DES depends solely on the number of hyperbox kept by the system. Therefore, in this section, we study the number of hyperboxes generated by the proposed framework for dealing with the large-scale datasets listed in Table 2.4 (rows 39 to 42). To complete this experiment, the DSEL size was varied from 100 to 900,000 to examine its effect on the number of hyperboxes generated and the system's accuracy. Specifically, the following DSEL sizes were utilized: S0: 100, S1: 1000, S2: 10K, S3: 100k, S4: 300k, S5: 500K, S6: 700K, and S7: 900K.

In this step, we tested three configurations with different values of the learning sensitivity hyperparameter, λ (0.8, 0.9, and 1.0). Increasing the value of λ is likely to improve accuracy

and increase the number of generated hyperboxes and overall computational complexity. Setting λ to 1.0 is expected to yield maximum accuracy with high computational complexity, while $\lambda = 0.8$ offers the lightest DS approach with satisfactory accuracy.

The accuracy versus complexity tradeoff of the three λ configurations is shown in Figure 2.18. The figure shows that the number of generated hyperboxes by FH_4M ($\lambda = 1.0$) grows significantly after step S4 (reaching 300K DSEL samples). Meanwhile, (FH_4M ($\lambda = 0.9$) and FH_4M ($\lambda = 0.8$)) generate way less hyperboxes for the same dataset size. This analysis also demonstrates that at step S4, the number of hyperboxes was reduced by 69% when λ was set to 0.8, with a minimal loss in accuracy relative to the configuration with $\lambda = 1.0$. Consequently, λ substantially impacts the system's complexity as it depends solely on the number of hyperboxes generated. Hence, a lower value of λ is preferable for generating fewer hyperboxes in experiments with larger datasets. For this reason, λ was fixed at 0.8 for the remainder of the experiment.



a) Influence of λ on the number of hyperboxes for the Sensor dataset

b) Influence of λ on classification accuracy for the Sensor dataset

Figure 2.18 The influence of learning sensitivity parameter (λ) on accuracy and number of generated hyperboxes. In this figure, FH_4M ($\lambda = 0.8$), FH_4M ($\lambda = 0.9$), and FH_4M ($\lambda = 1.0$) refer to FH_4-M variant with $\lambda = 0.8$, $\lambda = 0.9$, and $\lambda = 1.0$ respectively

In the subsequent phase of this experiment, the performance of the selected variant (FH_4-M) was compared to that of META-DES and FH_1-M, which is the FH-DES variant does not utilize

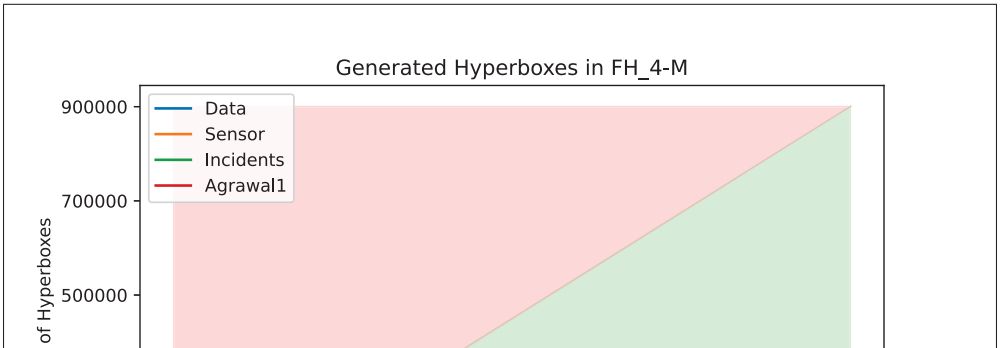
the contraction process (Davtalab *et al.* (2022)). The results obtained per dataset are presented in Table 2.9. The Oracle's accuracy is also indicated as the upper baseline for this analysis. It can be observed that the FH_4-M variant (with $\lambda = 0.8$), which incorporates a contraction process, is more accurate than FH_1-M. Moreover, the average accuracy of FH_4-M increases when additional samples are added to DSEL. Contrarily, the average accuracy of FH_1-M usually decreases with the addition of new samples. Compared to META-DES FH_4-M's present performance is equivalent to those of the META-DES for 3 out of 4 datasets.

In Figure 2.19, we visually depict the number of generated hyperboxes alongside the DSEL sample count, focusing on KNN-based DS methodologies like the META-DES technique. Notably, our proposed method stands out by generating a markedly smaller number of hyperboxes in comparison to the DSEL dataset size. For instance, considering a dataset of 900,000 samples, our approach generated a mere 5,610 hyperboxes for the Sensor dataset, representing less than 1% of the total instances within this dataset. As elaborated in Section 2.3.5, our method necessitates only one membership estimation per hyperbox, resulting in a mere 5,610 membership value estimations. This figure is significantly lower when contrasted with other KNN-based DS techniques that mandate the computation of distances with respect to all data points. Moreover, the storage cost of our proposed method equates to maintaining 11,200 instances in memory for inference—a notably lower overhead compared to the META-DES method, which retains the entire 900,000 examples from the DSEL dataset.

In addition, we can also observe a plateau regarding the number of hyperboxes that are added to the system as the dataset size increases. Thus, we can answer RQ3 confirming that the proposed approach has lower computational complexity than KNN-based DS approaches from the storage and computational perspectives while keeping an equivalent classification performance. Moreover, the results obtained in Table 2.9 demonstrate that the proposed framework can also learn incrementally using the contraction scheme, showing that it is a promising alternative for scaling DS methods to large datasets.

Table 2.9 Evaluating the system performance as the DSEL Size Increases in an Incremental Learning Scenario: FH_1-M Used as the baseline method which does not employ contraction mechanism

DataSets	META-DES	FH_1-M	FH_4-M	Oracle
ArData100	92.12	90.58	**92.73**	94.01
ArData1000	90.78	90.99	**92.77**	94.01
ArData10k	91.06	90.74	**92.77**	94.01
ArData100k	91.74	90.84	**92.78**	94.01
ArData300k	91.99	90.84	**92.87**	94.01
ArData500k	92.07	90.82	**92.87**	94.01
ArData700k	92.06	90.80	**92.87**	94.01
ArData900k	92.06	90.86	**92.90**	94.01
Sensor100	**95.9**	95.90	95.73	99.09
Sensor1000	**97.68**	97.26	96.23	99.09
Sensor10k	**98.21**	97.40	96.23	99.09
Sensor100k	**98.75**	96.61	96.27	99.09
Sensor300k	**98.78**	96.39	97.13	99.09
Sensor500k	**98.84**	96.36	97.13	99.09
Sensor700k	**98.84**	96.36	97.13	99.09
Sensor900k	**98.87**	96.38	97.20	99.09
Incidents100	**87.78**	87.68	87.30	88.06
Incidents1000	**87.86**	83.70	87.30	88.06
Incidents10k	**87.86**	79.84	87.70	88.06
Incidents100k	**87.86**	77.76	87.83	88.06
Incidents300k	**87.86**	77.79	87.84	88.06
Incidents500k	**87.88**	75.85	87.87	88.06
Incidents700k	87.88	72.24	**87.89**	88.06
Incidents900k	87.88	71.95	**87.89**	88.06
Agrawal100	67.02	67.2	**67.27**	75.68
Agrawal1000	**67.52**	67.24	67.33	75.68
Agrawal10k	**69.08**	67.56	67.37	75.68
Agrawal100k	**70.96**	67.76	67.47	75.68
Agrawal300k	**71.68**	67.34	67.50	75.68
Agrawal500k	**72.52**	66.85	67.53	75.68
Agrawal700k	**72.56**	66.28	67.57	75.68
Agrawal900k	**72.63**	66.02	67.60	75.68



2.7 Conclusion

This paper introduced a new dynamic ensemble selection framework based on fuzzy hyperboxes. For each base classifier, hyperboxes are formed based on their correctly classified or misclassified samples to define their ”competencies” and ”incompetencies” areas during the training phase. Together with the application of a fuzzy membership function, the competence map is generated from correctly classified samples and indicates the classifier's competence at each data point in the feature space. On the other hand, the incompetence map is generated from misclassified samples and indicates the regions where the classifier has low accuracy. Therefore, the novel approach presented in this study is the first to use only misclassified samples to select the most competent classifiers in the dynamic selection area.

The experimental study conducted in this research paper addressed four research questions. Firstly, the results showed that using hyperboxes in dynamic selection systems yielded greater accuracy and reduced computational complexity in response to RQ1. Secondly, the findings confirmed that fuzzy hyperboxes outperformed state-of-the-art dynamic ensemble selection (DES) techniques, as addressed in RQ2. In addition, RQ3 was answered positively, demonstrating that hyperboxes substantially reduced complexity compared to KNN-based DS approaches. Lastly, results demonstrate that incorporating a contraction process improved accuracy in large-scale problems, solving the main limitation of the preliminary approach based just on the expansion mechanism. Thus, demonstrating that the proposed framework is a viable solution for scaling DES methods to handle large datasets. On average, only 5k hyperboxes were required to model datasets consisting of 900k samples (50 hyperboxes for each base classifier's misclassifications).

In conclusion, the research presented in this work highlights the potential of FH-DES as a promising solution for addressing several of the main challenges within the domain of dynamic ensemble selection (DES). Future works will explore different hyperbox generation methodologies, including the Nested Generalization Exemplars (NGE) method (Salzberg (2012)) and the exploration of alternative geometric structures such as fuzzy hyperspheres

(Mahindrakar & Kulkarni (2022)). Furthermore, our future investigations will extend to the application of the FH-DES framework in contexts involving data streams and concept drift.

CONCLUSION AND RECOMMENDATIONS

This thesis provides a comprehensive study of Dynamic Selection (DS) approaches and introduces a novel dynamic ensemble selection. Most of the current state-of-the-art DS approaches are based on the KNN algorithm. However, the KNN algorithms have a great sensitivity to the local distribution of the data making it unsuitable for imbalanced classification problems. It also is sensitive to hyperparameters and the optimized hyperparameters may not work correctly in all regions (K-value problem). Moreover, KNN considers only limited information from the feature space. However, the main problem with KNN is a lack of ability to analyze high-dimensional datasets and handle large-scale problems.

The main contribution of this thesis is a new dynamic ensemble selection framework called FH-DES which uses fuzzy hyperboxes to select the best ensemble of classifiers among all available classifiers. The proposed approach aims to move most of the required computations from the generalization phase to the learning phase in order to improve the performance of DS approaches in both terms of accuracy and computational complexity. The proposed approach is capable of using the incompetence maps that are generated only based on misclassified samples. In this way, computational complexity is significantly reduced. Incompetence map indicates the regions where the classifier has low accuracy. According to the literature, the proposed framework is the first DS approach that uses only misclassified samples to select the ensemble of classifiers. According to the experimental results, utilizing the incompetence map significantly increases accuracy in comparison to the cases that use the competence map. In addition, incompetence maps are formed based on fewer hyperboxes and consequently have less computational complexity. Experimental results also show that the proposed approach has the best rank among the compared methods and displayed better accuracy than all other approaches. In some instances, the difference was significant.

Furthermore, a contraction mechanism was proposed in this thesis to improve the incremental learning capability of the framework. According to the obtained results, utilizing the contraction mechanism increases the accuracy system when new learning samples are added to the system. In addition, we can effectively scale up the proposed framework to handle large-scale problems with lower computational complexity in comparison to the current DS systems. While KNN-based approaches need to store and analyze all the DSEL data so they are not scaleable to be used in large-scale classification applications. According to the obtained results, the proposed framework has very light computational complexity in the generalization phase. This approach has approximately 100 times less computational complexity than current DS approaches in some large-scale datasets.

It's worth noting that while our proposed method demonstrates promising outcomes, it does show a slight sensitivity to the order of data. However, this challenge is more nuanced than initially perceived, representing a subtlety rather than a significant obstacle. This issue could be solved using a few replications and aggregating the resulting hyperboxes. Addressing this subtlety will refine the method and ensure its adaptability to various scenarios.

3.1 Future works

The proposed framework has a flexible structure that could be used in different machine learning areas. Looking forward, the following ideas need to be further explored:

- The proposed framework gives us an overall view of the competence of the base classifiers and also the regions that do not work well. So it can be used for the **pool generation** step of dynamic selection to have competent and diverse classifiers in the classifiers pool.
- Hyperbox-based approaches can easily handle high-dimensional problems. Therefore, investigating the efficiency of the proposed approach in **high-dimensional problems** would be interesting.

- Hyperbox-based approaches can learn adaptively. Therefore, exploring their effectiveness in the **online learning** field would be an attractive topic for future work. This capability also allows us to use hyperboxes to detect and handle the concept drift in machine learning problems.
- In hyperbox-based approaches, several samples that are close to each other are represented by a hyperbox. In other words, we can use hyperboxes to **represent the data granularly**, which could decrease the processing complexity. In this way, the learning system will not be sensitive to the local imbalance distribution of samples which is suitable for solving imbalance data. As a result, investigating the efficiency of hyperbox-based systems on imbalanced data would be an interesting topic.

APPENDIX I

CONFERENCE PAPER: DYNAMIC ENSEMBLE SELECTION USING FUZZY HYPERBOXES

Reza Davtalab[1] , Rafael M.O. Cruz[1] , Robert Sabourin[1]

[1] LIVIA, École de technologie supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in International Joint Conference on Neural Networks (IJCNN), July 2022

1. Abstract

Most dynamic ensemble selection (DES) methods utilize the K-Nearest Neighbors (KNN) algorithm to estimate the competence of classifiers in a small region surrounding the query sample. However, KNN is very sensitive to the local distribution of the data. Moreover, it also has a high computational cost as it requires storing the whole data in memory and performing multiple distance calculations during inference. Hence, the dependency on the KNN algorithm ends up limiting the use of DES techniques for large-scale problems. This paper presents a new DES framework based on fuzzy hyperboxes called FH-DES. Each hyperbox can represent a group of samples using only two data points (Min and Max corners). Thus, the hyperbox-based system will have less computational complexity than other dynamic selection methods. In addition, despite the KNN-based approaches, the fuzzy hyperbox is not sensitive to the local data distribution. Therefore, the local distribution of the samples does not affect the system's performance. Furthermore, in this research, for the first time, misclassified samples are used to estimate the competence of the classifiers, which has not been observed in previous fusion approaches. Experimental results demonstrate that the proposed method has high classification accuracy while having a lower complexity when compared with the state-of-the-art dynamic selection methods. The implemented code is available at https://github.com/redavtalab/FH-DES_IJCNN.git.

2. Introduction

Multiple Classifier Systems (MCS) are a good solution for the complex and vast amounts of data that we face today (Cruz *et al.* (2018a); Zyblewski *et al.* (2021)). Different types of MCS have been introduced, but many researchers concluded that Dynamic Selection (DS) could be a better choice for the combination of classifiers (Britto Jr *et al.* (2014); Cruz *et al.* (2018a)). In DS approaches, each given query sample is labeled by an ensemble of base classifiers which are usually selected with regards to their local competence.

Estimation of competence level is a key issue in DS approaches. In this stage, Dynamic Selection Data (DSEL) is used to evaluate the competence level of classifiers. For this purpose, the efficiency of the classifiers in a small region surrounding the query instance on DSEL data is considered as an estimation of the local competence of classifiers (Kuncheva (2014); Britto Jr *et al.* (2014); Cruz *et al.* (2018a)). This region is called Region of Competence (RoC) and in most of the DS approaches, this region is defined either by the K-Nearest Neighbor (KNN) technique applied in the feature space (Cruz *et al.* (2015d); Fernández-Delgado *et al.* (2014); Xiao *et al.* (2016); Krawczyk *et al.* (2018); Cruz *et al.* (2018a); Elmi & Eftekhari (2020)), clustering (Lin *et al.* (2014)), potential functions (Woloszynski & Kurzynski (2009); Woloszynski *et al.* (2012)) or the KNN applied in the decision space (Giacinto & Roli (2001); Cavalin (2012); Batista *et al.* (2012); Nguyen *et al.* (2020)).

KNN-based approaches are more popular; however, they suffer from high complexity in the generalization phase. In this stage, for each query sample, its k nearest neighbors must be found. This means that the distance between the given data point and all samples must be calculated, which endures the system's huge calculation complexity. Clustering-based approaches reduce this complexity by adopting more coarse-grained regions of competence (clusters). They require only require calculating distances to each cluster centroid and then selecting the most competent classifiers according to the nearest cluster. However, the reduction in complexity comes with a significant loss in accuracy compared with KNN-based approaches (Cruz *et al.* (2018a)).

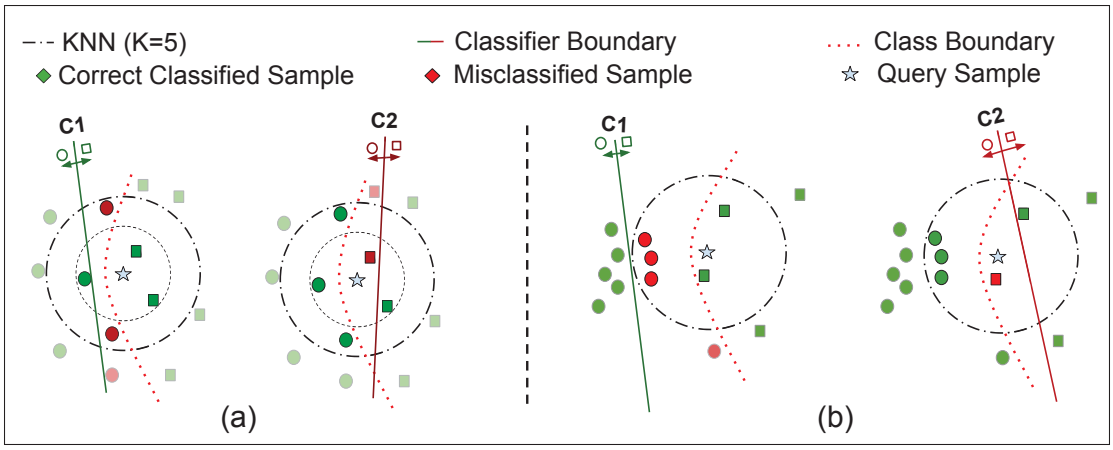


Figure-A I-1 Problems of KNN-based DS approaches. (a) K-value problem, and (b) High sensitivity to the local distribution of data. In both cases, just C1 could correctly classify the query sample while KNN (K=5) selects C2 as a competent classifier

Moreover, the selected K-value may not work correctly in all regions (K-value problem), even if it has been optimized using an optimization process. As shown in Figure I-1-a, K=5 results in selecting the wrong classifier (C2) while K=3 can select the correct one. Techniques based on potential function aim to solve this problem by not having a K-value and considering all data points during the competence estimation. In this case, it considers a potential function that gives higher weights to the samples closer to the query while decreasing as the distance increases (Cruz *et al.* (2018a)). However, its computational complexity is even higher compared to KNN-based approaches as it not only suffers from the high computational cost of calculating the distance between all samples in memory but also needs to aggregate the information of the whole set with the application of the potential function.

Additionally, KNN works based on the Euclidean distance and has great sensitivity to the local distribution of data. Hence, a high degree of overlap in the data may lead to a wrong decision (Figure I-1-b). Finally, KNN just considers the samples of RoC, which contain a limited amount of information. Thus, DS techniques can end up limited to the main problems of the KNN technique, and new ways of estimating the classifier's competence are needed in order to achieve better classification results while reducing the computational cost.

Intuitively, defining and storing each classifier's competence and incompetence areas could increase the labeling speed in the generalization phase. Falling the query sample $\mathbf{x_q}$ into the competence region of c_i means that this classifier is competent to classify $\mathbf{x_q}$. Figure I-2 illustrates the initial idea of this approach to solve the example of Figure I-1-a. In this example,

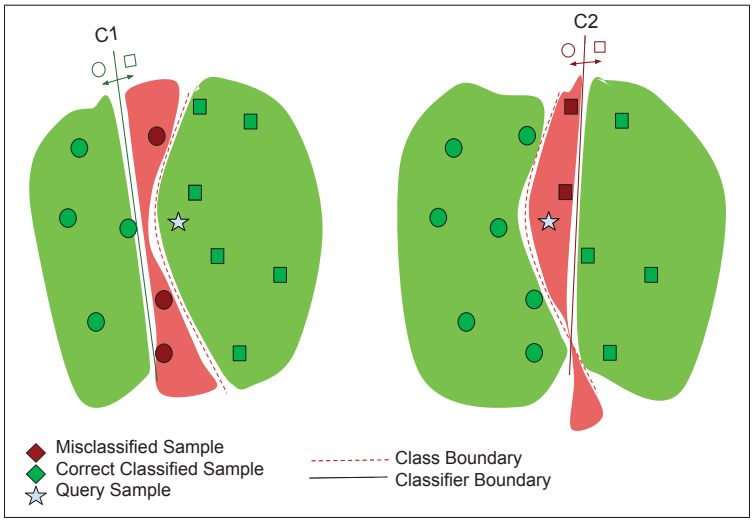


Figure-A I-2 Competence and incompetence areas of classifiers in Figure I-1-a

the competence of C1 is estimated higher than C2, because the query sample has fallen in the green area of C1. However, defining the domain of such areas is not easy and imposes a large computational complexity on the system, unless some simpler structures are used to represent these areas. In a two-dimensional feature space, we can represent these areas using rectangles. Each rectangle could be defined by only two points. Therefore, its computational complexity will not be high if there are an acceptable number of rectangles to represent all training samples.

Hyperbox is a virtual concept that works like these rectangles; however, it is capable of working in high-dimensional spaces. Each hyperbox covers the interior space and a small part of its vicinity. As we move away from the hyperbox, its coverage decreases fuzzily according to a membership function. That is why it is called *fuzzy hyperbox* (Simpson (1992)). The fuzzy aspect of the hyperbox gives us valuable information outside of the hyperbox, and we can estimate how far the query sample is from the competence or incompetence area of the base

classifier. Thus, we can estimate the competence of classifiers even if the query sample falls outside of all hyperboxes. We will discuss hyperboxes in detail in the section 3.

In Figure I-3, fuzzy hyperboxes are used to represent the competence regions in the examples of Figures I-1. As illustrated in Figure I-3-a, the query sample is located outside of all hyperboxes. However, it is located close to the hyperbox of classifier C1 (inside its green area of C1). Therefore, this classifier is considered to be more competent than classifier C2.

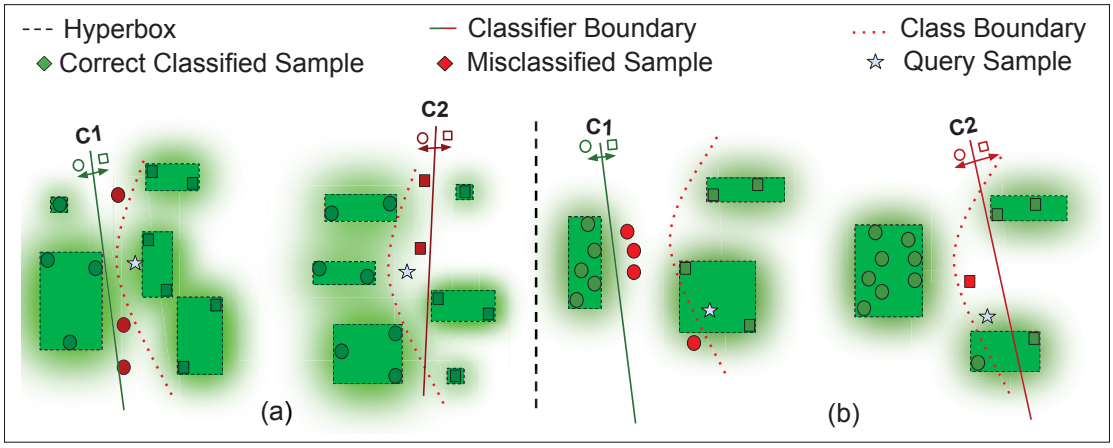


Figure-A I-3 Solving the illustrated problems of KNN in Figure I-1 using Fuzzy Hyperboxes based on correct classified samples that represent competence areas

In summary, this research aims to answer the following research questions: (1) Can the use of fuzzy hyperboxes lead to more accurate dynamic selection approaches? (2) Do the misclassified samples have enough information to estimate the competence of classifiers? (3) Will the use of hyperboxes lead to reduced computational complexity compared to current DS techniques?

The rest of the paper is organized as follows: In Section 3, the background of Fuzzy Hyperboxes is reviewed. The proposed method is discussed in Section 4. Finally, the experimental results and conclusion are discussed in Section 5 and Section 6, respectively.

3. Fuzzy Hyperbox

Hyperbox was introduced by Simpson in 1992 as a building block for Fuzzy Min-Max Neural Networks (FMM) (Simpson (1992); Simpson & Jahns (1993)). Hyperbox is defined by its two corners named *Min (***v***)* and *Max (***w***)* corners.The size and location of the hyperboxes are easily adjustable by changing these two corners. Hyperbox-based learning systems have some features that make them promising tools in machine learning applications: the ability to make soft and hard decisions, scalability, online adaptation, and the ability to model granular data (Khuat *et al.* (2021b)).

3.1 Learning process

The learning process of hyperboxes is a single-pass process in which hyperboxes are formed regarding learning data to cover the needed regions. During this process, for each learning instance **x**, a hyperbox must be found that contains **x** or is expandable enough to contain this sample. Figure I-4 shows how the hyperbox B_j is expanded to involve the sample **x**. In this example, v_{j2} and w_{j1} are changed to expand the hyperbox.

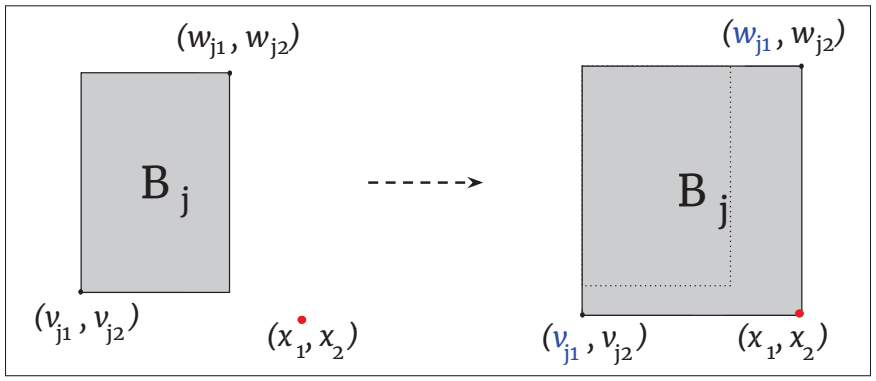


Figure-A I-4 Expansion of hyperbox B_j to involve sample $\mathbf{x}(x_1, x_2)$

The maximum size of hyperboxes is limited by the user-defined parameter θ during the learning process. At the end of this process, if no expandable hyperbox is found, a new hyperbox is created.

Each hyperbox is defined by the following equation.

$$B_j = \{\mathbf{v_j}, \mathbf{w_j}, b_j(\mathbf{x})\} \; \forall \, \mathbf{x} \in I^n \tag{A I-1}$$

In this equation, $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ is a single data point. $\mathbf{w_j} = \{w_{j1}, w_{j2}, ..., w_{jn}\}$ and $\mathbf{v_j} = \{v_{j1}, v_{j2}, ..., v_{jn}\}$ are min and max corners of the hyperbox, respectively. b_j is the membership function of the hyperbox B_j. Also, I^n is n dimensional feature space.

3.2 Membership Function

The membership function of the hyperbox is a crucial part of the fuzzy Min-Max neural network technique. It is utilized to quantify the membership grade of an arbitrary instance to the hyperbox B_j (between 0 and 1). The membership function of a hyperbox is usually defined so that the degree of membership inside the hyperbox B_j is equal to one and decreases when the feature point moves away from the hyperbox.

Many membership functions were proposed for hyperboxes. However, the membership function introduced by Gabrys and Bargiela (Gabrys & Bargiela (2000)) is the most popular membership function among the fuzzy hyperbox's applications (Khuat *et al.* (2021a)). It has a simple structure (only the min and max points), and the membership value monotonically decreases by increasing each side of the hyperbox. This function is defined as follows:

$$\begin{aligned} b_j(\mathbf{x}) \;\; = \;\; & min_{i=1..n}(min([1 - f(x_i - w_{ij}, \gamma_i)], \\ & [1 - f(v_{ij} - x_i, \gamma_i)])). \end{aligned} \tag{A I-2}$$

Where,

$$f(r, \gamma) = \begin{cases} 1 & \text{if } r\gamma > 1 \\ r\gamma & \text{if } 0 \leq r\gamma \leq 1 \\ 0 & \text{if } r\gamma < 0 \end{cases} \tag{A I-3}$$

Where a_i is i_{th} dimension of a sample **x**, and γ is the sensitivity parameter that regulates the rate with which the membership values decrease out of the hyperbox. However, this membership function has sharp corners, which assigns a higher membership to further samples in some cases. Membership levels around the hyperbox and the mentioned problem of this membership function are shown in Figure I-5.

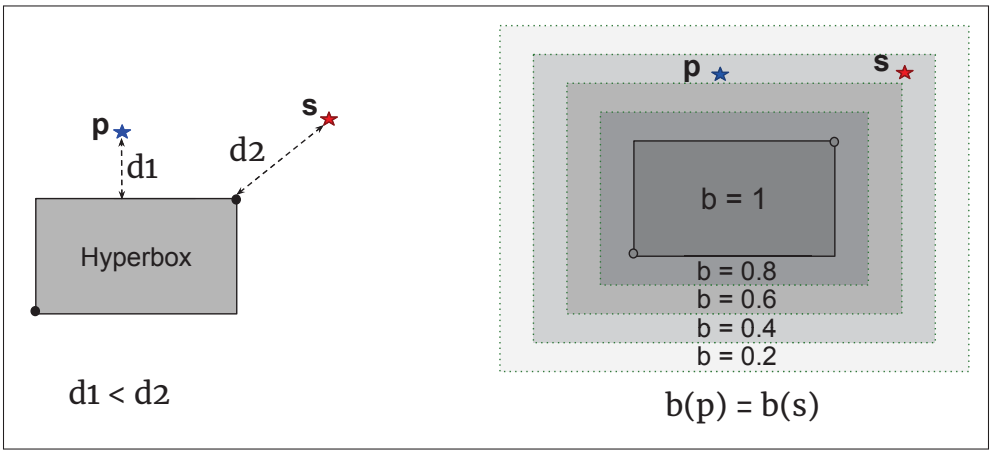


Figure-A I-5 Membership function proposed by Gabrys and Bargiela and its corners problem

As can be observed, this function assigns the same membership value to the points P and S, while P is closer to the hyperbox than S.

4. Proposed Framework (FH-DES)

Here, a novel DS framework based on fuzzy hyperboxes is introduced, called Fuzzy Hyperbox-based Dynamic Ensemble Selection (FH-DES). In this approach, the competence or incompetence areas of classifiers (Figure I-2) are defined by fuzzy hyperboxes.

When hyperboxes are built based on correctly classified examples (Figure I-3), they represent regions where the classifiers work well or competence areas. This approach will be called FH-DES-C in the rest of the paper. In contrast, when hyperboxes are built with misclassified samples, the approach is called FH-DES-M, in which hyperboxes represent areas of incompetence. In

this case, the classifier whose hyperboxes have a lower membership degree will be farther from query sample $\mathbf{x_q}$, so it will be more competent to classify the query sample.

Therefore, the competence of the classifier c_i to classify the given sample $\mathbf{x_q}$ is estimated according to the membership function of hyperboxes that belong to c_i.

As mentioned in section 3, Gabrys's membership function (Gabrys & Bargiela (2000)) has some problems in the corners of membership levels. To fix these problems, we introduce a new membership function with smoother borders (SBM):

$$b_j(\mathbf{x_q}) = (||ReLU(|\mathbf{o_j} - \mathbf{x_q}| - (\mathbf{w_j} - \mathbf{v_j})/2)||_2)^2 \quad \text{(A I-4)}$$

Where $\mathbf{o_j}$ is the center of hyperbox B_j, $\mathbf{v_j}$ and $\mathbf{w_j}$ are min and max corners respectively, $||.||_2$ indicates 2-norms, and $ReLU(\cdot)$ is the Rectified Linear Unit (ReLU) function as below:

$$ReLU(a) = max(0, a) \quad \text{(A I-5)}$$

In Figure I-6, the membership levels of SBM are illustrated. The smooth borders of this function help us to solve the mentioned problem in Figure I-5.

In this method, all necessary calculations to define competence (or incompetence) areas are performed during the training phase and only membership values are calculated during the generalization phase to label the query sample $\mathbf{x_q}$. Therefore, the proposed framework is expected to have less complexity than KNN-based approaches. In addition, the computational complexity of FH-DES-M should be less than FH-DES-C. Because the number of misclassified samples is usually less than the correctly classified samples.

4.1 Training phase

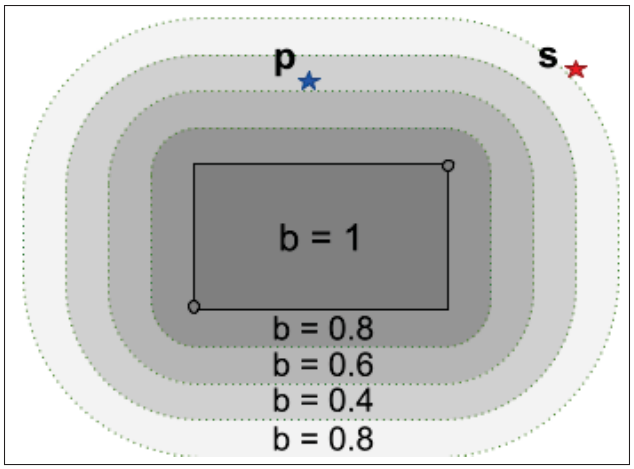


Figure-A I-6 The proposed Smooth-Border membership function (SBM) for FH-DES framework

During the training phase, after generating the pool of classifiers, all the needed hyperboxes are formed according to the performance of the base classifiers (on DSEL data). It consists of the expansion process of FMM (Simpson & Jahns (1993)).

In particular, suppose we want to use the FH-DES-M approach, and $Mset_i$ contains misclassified samples of classifier c_i, all hyperboxes of c_i are built using $Mset_i$ according to the learning process of hyperboxes (Subsection 3.1). The set of hyperboxes, which belongs to the classifiers c_i, is called $Hset_i$. The distribution of hyperboxes depends on the order of the samples within $Mset_i$. Consequently, some hyperboxes can overlap in the feature space; However, it does not affect the system's performance.

Figure I-7 represents the training phase of the proposed approach based on misclassified samples.

As mentioned in Subsection 3.1, the hyperbox creation process for the classifier c_i begins by picking a sample of $Mset_i$ and finding a hyperbox of $Hset_i$ that includes (or can expand to include) the picked sample. If such a hyperbox is not found, a new hyperbox is created at the same point and added to $Hset_i$.

4.2 Generalization phase

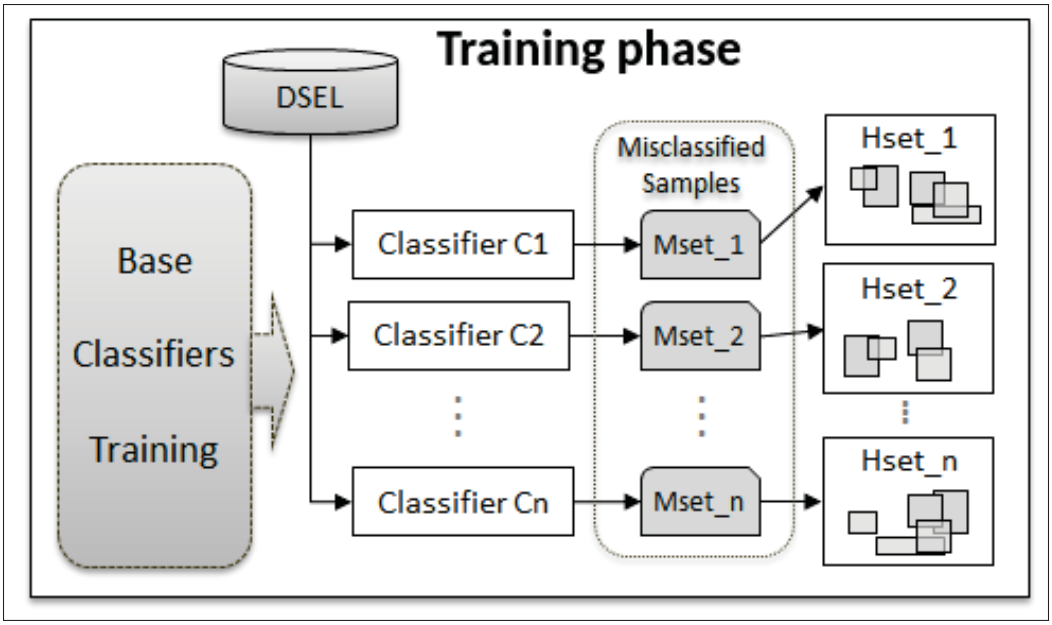


Figure-A I-7 Block Diagram of the training phase of FH-DES based on misclassified samples

During the generalization phase, for each given query sample $\mathbf{x_q}$, the performance of all classifiers is estimated based on their hyperboxes, and the best ensemble of classifiers is selected. Specifically, for each query sample $\mathbf{x_q}$, the competence of the classifier c_i, represented by δ_i, is calculated as follows:

$$\delta_i(\mathbf{x_q}) = (b^{i*} + b^{i+})/2 \tag{A I-6}$$

Here b^{i*} and b^{i+} are the first and second highest membership values among the hyperboxes of c_i. It should be noticed that in the correct-classified version (FH-DES-C), the membership value of hyperboxes is related to the competence of the classifier. While in FH-DES-M, the membership values of hyperboxes represent the incompetence of classifiers. Thus, the calculated competence value should be deducted from 1 ($\delta_i \leftarrow 1 - \delta_i$) in the FH-DES-M version.

In the next step, the ensemble of the most competent classifiers is selected according to a global threshold τ_q. The threshold is defined by Equation A I-7.

$$\tau_q = \mu \times \max_{i=1..M} (\delta_i(\mathbf{x_q})) \tag{A I-7}$$

In this equation, μ is a predefined parameter between 0 and 1. Therefore, the final ensemble of classifiers (ϕ) is formed considering threshold τ_q as below:

$$\phi(\mathbf{x_q}) = \{c_i | \delta_i(\mathbf{x_q}) \geq \tau_q\} \tag{A I-8}$$

Where μ is equal to one, only the most competent classifier(s) is selected. On the contrary, when μ equals zero, all classifiers will be selected. Finally, in the aggregation step, outputs of the selected classifiers are combined with weighted majority voting by associating competence to the classifiers as their weights:

$$\hat{y} = arg \max_{\lambda} \sum_{\forall l \in \lambda} \delta_i(\mathbf{x_q}) \quad | \quad c_i(\mathbf{x_q}) = l, c_i \in \phi(\mathbf{x_q}) \tag{A I-9}$$

Where λ is the set of unique class labels and l represents the label of $\mathbf{x_q}$. The pseudocode of the generalization phase is represented in the algorithm I-1.

Algorithm-A I-1 Labeling process of $\mathbf{x_q}$

Input: $\omega, \mathbf{x_q}$
Output: Predicted label of $\mathbf{x_q}$

1 Calculate the membership value of $\mathbf{x_q}$ for all hyperboxes ;
2 Calculate competence of all classifiers by eq (A I-6) ;
3 Select the ensemble of classifiers by eq (A I-8) ;
4 Aggregate outputs of selected classifiers by eq (A I-9) ;
5 Label the given sample

In summary, unlike previous DS techniques, in this approach, the strength (competence) or weakness (incompetence) of the classifiers is considered to select the final ensemble of the

classifiers, while in current DS approaches, only their strength (competence) is considered as a selection criterion (Cruz *et al.* (2018a)).

In addition, the approach solves some problems of the KNN-based approaches, such as the K-value problem, sensitivity to the local distribution, and limited information. Therefore, we expect the proposed method to have higher accuracy than KNN-based approaches. In addition, since in the proposed framework, only hyperboxes are utilized in the generalization phase (instead of instances), and the framework contains fewer hyperboxes than samples, it is expected that the proposed approach will be faster than KNN-based DS techniques. This is especially true in the FH-DES-M approach, where hyperboxes are formed based on only misclassified samples.

5. Experimental results

In this section, the efficiency of the proposed framework is evaluated using 30 datasets and compared with other DS approaches. To have a better comparison, we used a similar experimental protocol that is used in (Cruz *et al.* (2018a)) and (Cruz *et al.* (2015d)). In this experiment, the pool of classifiers contained 100 perceptrons that were generated using the bagging technique (Breiman (1996)). This pool was fixed for all techniques to ensure a fair comparison. In these experiments, each dataset was randomly divided into 50% training data, 25% the dynamic selection dataset (DSEL), and 25% test data. The division was performed by maintaining the prior probability of each class. To implement different algorithms, DESlib toolkit (version 0.3.5) (Cruz *et al.* (2020)) was used. Furthermore, each experiment was conducted using 20 replications and the mean of the evaluation criteria has been reported. In all experiments, the perceptron of the SciKit-learn library (Version: 1.0.1) is used as a base classifier. Some of the parameters required in these experiments are reported in Table I-1.

The proposed framework has two main hyperparameters, including Theta (θ), which defines the maximum size of hyperboxes, and Mu (μ) which defines a threshold to select top base classifiers. Both hyper-parameters are set in the range 0 to 1. To tune these hyper-parameters, we use a similar process to that used by Cruz et al.(Cruz *et al.* (2015d)). Ten different datasets, which

Table-A I-1 Used parameters and their values in experiments

Method	Parameter Name	Value
Perceptron	Maximum iteration	100
	Tolerance	10e-3
	Alpha	0.001
CalibratedClassifierCV	CV	prefit
	Calibration method	isotonic
Bagging	Number of estimators	100
	bootstrap	True
	max- samples	1.0
KNN-based DS	K	7
META-DES	K_p	5
	h_c	80%

have not been used in the comparative study, were used during this process to avoid biased estimation. The tuning experiments were carried out using the same experimental protocol and the optimal values found were $\theta = 0.27$ and $\mu = 0.99$. Therefore, these values were used in the main experiments.

All simulation details are available in FH-DES's GitHub repository[3].

5.1 Datasets

In our experimental study, 40 different real-world datasets in a wide variety of areas were used. These datasets were selected with different specifications to evaluate different aspects of the proposed approach better. All datasets were collected from OpenML (Van Rijn *et al.* (2013)), UCI (Asuncion & Newman (2007)) repositories, and previous DS research. In Table I-2 the utilized datasets and their specifications are listed. The first ten datasets in this table were used to tune the proposed approach's hyperparameters. The other 30 datasets were used in our comparative study.

[3] https://github.com/redavtalab/FH-DES_IJCNN.git

Table-A I-2 Dataset considered in this work and their main features

#	Database	Instances	Features	#	Database	Instances	Features
1	Adult	690	14	21	Heart	270	13
2	Audit2	776	17	22	ILPD	583	10
3	CTG	2126	21	23	Ionosphere	351	34
4	Cardiotocography	2126	21	24	Laryngeal1	213	16
5	Chess	3196	36	25	Laryngeal3	353	16
6	Credit-screening	690	15	26	Lithuanian	600	2
7	Ecoli	336	7	27	Liver	345	6
8	Glass	214	9	28	Mammographic	830	5
9	P2	1000	2	29	Monk2	432	6
10	Transfusion	748	4	30	Phoneme	5404	5
11	Audit	771	26	31	Pima	768	8
12	Banana	1000	2	32	Sonar	208	60
13	Banknote	1372	4	33	Statlog	1000	20
14	Blood	748	4	34	Steel	1941	27
15	Breast	569	30	35	Thyroid	692	16
16	Car	1728	6	36	Vehicle	846	18
17	Datauser	403	5	37	Vertebral	310	6
18	Faults	1941	27	38	Voice3	238	10
19	German	1000	24	39	Weaning	302	17
20	Haberman	306	3	40	Wine	178	13

5.2 Performance of the proposed framework in different configurations

To determine the best configuration of the proposed framework, we compare their accuracy over the 30 datasets. Thus, we examined the influence of misclassified samples and the smooth corner membership function on the performance of the proposed framework. Therefore, there are four different configurations of the proposed framework. FH-GC and FH-GM approaches are based on the Gabrys membership function, which uses correct-classified and misclassified samples, respectively. Two others utilize the proposed membership function (SBM). This group contains FH-DES-C and FH-DES-M based on correct-classified and misclassified samples, respectively. The classification accuracy of these approaches alongside their standard deviations is reported in Table I-3.

We can see in this table that FH-DES-M achieved the highest accuracy in 13 out of 30 datasets. Additionally, this approach achieved the highest average accuracy and the best average rank in 30 datasets. In addition, using the Gabrys membership function, FH-GM proposed framework achieved the highest accuracy in 11 of 30 datasets.

Table-A I-3 Average accuracy and standard deviation of the proposed method in different configurations

DataSets	FH_GC	FH_GM	FH_DES-C	FH_DES-M
Audit	96.81(0.77)	**96.94(0.8)**	96.87(0.74)	96.87(0.92)
Banana	89.26(2.5)	89.4(2.23)	89.12(2.37)	**89.5(1.68)**
Banknote	99.1(0.57)	**99.52(0.51)**	99.13(0.66)	99.34(0.5)
Blood	**77.57(2.32)**	77.22(2.18)	77.46(1.98)	76.55(2.64)
Breast	96.47(1.52)	**96.82(1.49)**	96.29(1.49)	96.61(1.61)
Car	73.26(1.24)	72.96(1.09)	**74.63(1.08)**	74.11(1.25)
Datausermodeling	87.62(3.66)	91.19(4.03)	87.77(3.93)	**91.29(3.63)**
Faults	69.31(2.1)	69.74(2.28)	70.12(1.84)	**70.38(2.03)**
German	**74.94(2.02)**	74.92(2.28)	74.8(2.0)	74.86(2.27)
Haberman	71.23(3.99)	**71.56(3.78)**	71.36(3.88)	**71.56(3.98)**
Heart	83.01(3.82)	82.43(4.2)	82.79(3.89)	**83.9(4.42)**
ILPD	70.92(3.7)	70.34(3.7)	**71.3(3.1)**	70.65(2.59)
Ionosphere	88.75(2.43)	**88.75(1.52)**	88.35(2.21)	88.13(1.37)
Laryngeal1	81.85(3.69)	81.85(3.29)	81.94(3.25)	**82.5(3.49)**
Laryngeal3	71.8(3.46)	**72.13(4.46)**	71.01(4.15)	71.8(4.08)
Lithuanian	89.3(2.25)	**90.57(2.21)**	89.63(2.08)	90.5(2.32)
Liver	66.72(3.85)	**69.43(4.88)**	67.82(4.77)	69.14(4.34)
Mammographic	78.73(2.87)	**79.18(2.82)**	78.03(3.2)	78.87(2.73)
Monk2	79.03(3.18)	81.81(3.55)	86.2(2.85)	**87.64(3.24)**
Phoneme	77.73(0.91)	**78.13(0.89)**	77.81(0.97)	78.1(0.91)
Pima	75.47(1.91)	75.68(2.41)	74.9(2.46)	**76.28(2.72)**
Sonar	80.29(5.5)	80.48(5.84)	**81.35(6.41)**	79.62(5.42)
Statlog	75.1(2.6)	74.94(2.32)	**75.26(1.91)**	75.08(2.0)
Steel	70.64(1.58)	70.57(1.34)	70.72(1.67)	**71.37(1.33)**
Thyroid	95.92(1.56)	95.84(1.49)	95.87(1.32)	**95.98(1.37)**
Vehicle	74.17(1.83)	74.32(2.17)	74.53(2.1)	**75.05(2.41)**
Vertebral	82.88(3.83)	**84.36(4.28)**	82.44(3.54)	84.04(3.23)
Voice3	**77.58(3.35)**	77.33(3.39)	77.17(3.42)	76.58(3.09)
Weaning	80.72(4.77)	82.24(4.91)	81.25(4.94)	**82.43(4.39)**
Wine	97.67(1.79)	97.33(2.06)	97.11(2.23)	**98.0(1.71)**
Average	81.13	81.6	81.43	**81.89**
Ave Rank	2.93	2.26	2.80	**1.95**

In the next step, statistical analysis is conducted using the post-hoc Bonferroni-Dunn test (Demšar (2006)). This test is applied to compare the ranks achieved by each DS method. The average ranks of different configurations of the proposed framework and the result of the Bonferroni-Dunn post-hoc test are presented in Figure I-8 using the Critical Difference (CD) diagram. The performance of the two DS approaches is significantly different if their difference in average rank is higher than the CD value.

According to the post-hoc test, FH-DES-M is significantly better than FH-GC. And there is no significant difference between the other configurations of the proposed framework. However, FH-DES-M is slightly better than others.

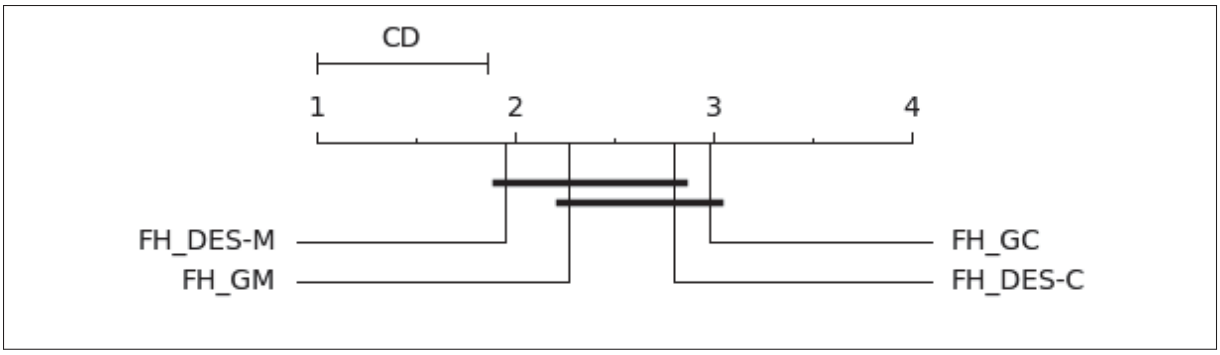


Figure-A I-8 Critical Difference (CD) diagram for different configurations of the proposed framework. The best algorithm is the one presenting the lowest rank. Techniques that are statistically equivalent are connected by a black bar

In the next step, a pairwise comparison is conducted to compare the obtained results of FH-DES-M and FH-GM against the other configurations, based on Sign test (Demšar (2006)). In this test, there are two hypotheses that include H_0 (null hypothesis) and H_1 (alternate hypothesis). Rejection of H_0 means that the performance of the corresponding DS technique is significantly better than the compared technique. The number of wins, ties, and losses for each technique is computed compared to the baseline. The significance level of this test is determined by the predefined parameter α, which in this paper is set $\alpha = 0.05$ to have 95% confidence. If the number of wins is greater than or equal to a critical value, denoted by n_c, the null hypothesis H_0 is rejected. The critical value is computed using equation A I-10:

$$n_c = \frac{n_{exp}}{2} + z_\alpha \frac{\sqrt{n_{exp}}}{2} \tag{A I-10}$$

Where n_{exp} is the total number of experiments and $z_\alpha = 1.645$, for a significance level of $\alpha = 0.05$ (Cruz *et al.* (2017b)). In this test, we have 30 experiments (datasets). Therefore, $n_{exp} = 30$, so for this amount of experiments, the critical value is $n_c = 19.5$. Obtained results for FH-GM and FH-DES-M are represented in Figure I-9 and Figure I-10 respectively.

It can be observed that the proposed framework statistically has good performance using both the Gabrys membership function and the proposed membership function. FH-DES-M is slightly

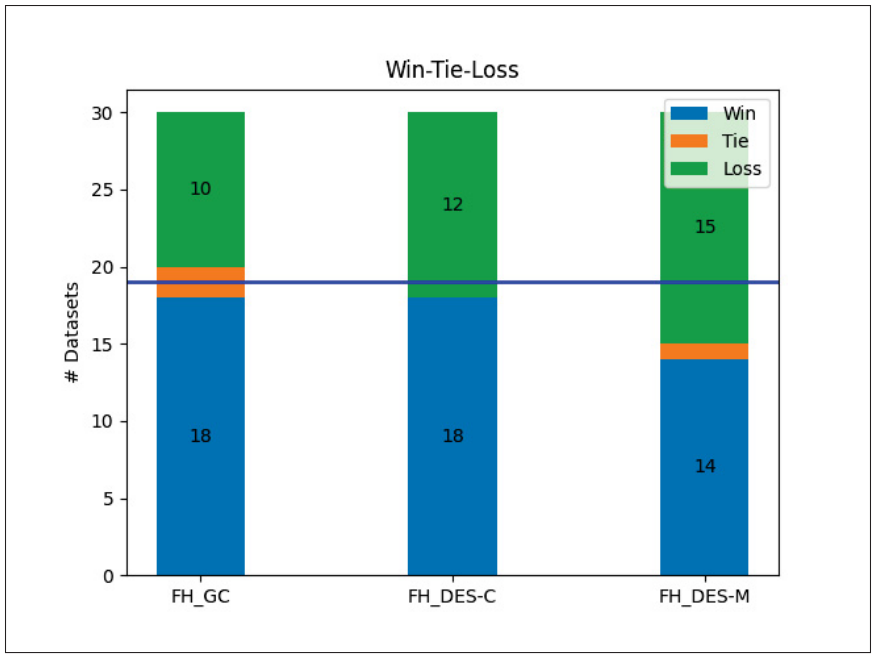


Figure-A I-9 Pairwise comparison between the FH-GM and other configurations of the proposed approach (n_c = 19.5)

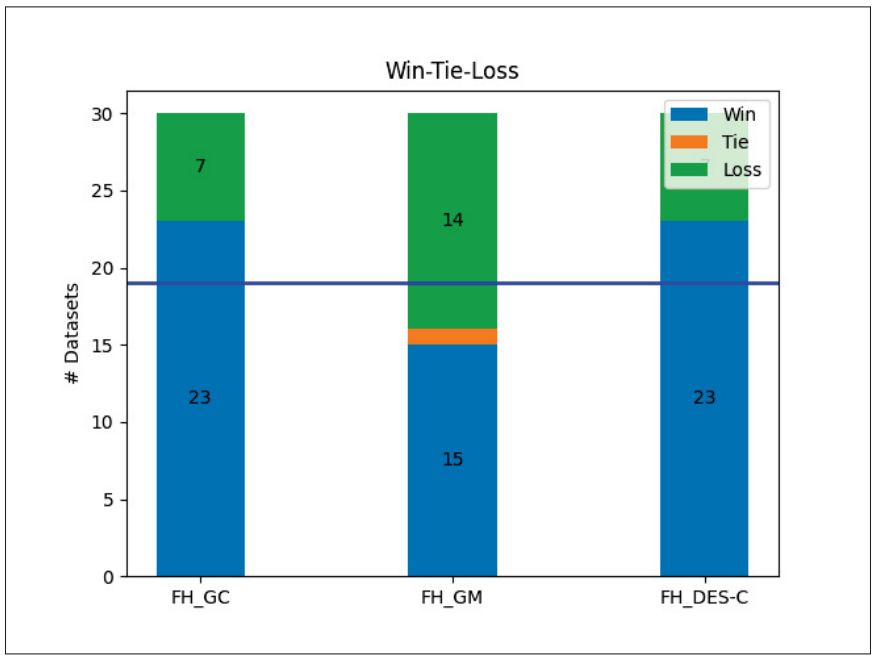


Figure-A I-10 Pairwise comparison between the FH-DES-M and other configurations of the proposed approach (n_c = 19.5)

more accurate than FH-GM, but there is no significant difference. However, these two approaches are significantly better than the correct-classified versions. Therefore, it can be concluded that the misclassified samples contain more helpful information to estimate the competence of classifiers rather than correct-classified ones. The reason is simple; boundary regions are usually challenging in classification applications. Most classification errors occur in these areas, and misclassified samples are usually found there. When we form hyperboxes based on misclassified samples, most of the hyperboxes are formed in these areas. The proximity of hyperboxes to the boundary regions means that the system makes a more precise decision in this region. Thus, it results in higher accuracy during the generalization phase. Furthermore, according to Figures I-10 and I-8, FH-DES-M statistically outperformed FH-GC and FH-DES-C and was slightly better than FH-GM. Therefore, FH-DES-M was selected as the best configuration of the proposed framework and compared with other DS approaches in the next step.

5.3 Comparison with state-of-the-art DS methods

The accuracy and standard deviation of the proposed framework and the state-of-the-art DS approaches are reported in Table I-4. In this table, the Oracle approach (Kuncheva (2002)) is a conceptual method that selects the base classifier which labels the query sample correctly if such a base classifier exists.

We can see in Table I-4 that the proposed approach FH-DES-M achieved the highest average accuracy among all DS approaches, and its average rank is very close to the DESKNN that obtained the lowest rank.

Figure I-11 presents the average ranks of different DS techniques and the result of the Bonferroni-Dunn post-hoc test using a critical difference diagram.

This figure shows that DESKNN and FH-DES-M have the best overall rank among the compared methods, outperforming all DS approaches. However, there is no significant difference between these DES approaches over the mentioned datasets. For a more fine-grained comparison

Table-A I-4 Average accuracy and standard deviation of the proposed method and other DES approaches

DataSets	Oracle	KNORA-E	RANK	KNORA-U	OLA	DESKNN	META-DES	KNOP	FH-DES-M
Audit	99.56(0.5)	96.89(0.73)	96.71(0.74)	96.19(1.3)	**97.02(0.93)**	96.76(0.99)	96.74(1.04)	96.24(1.36)	96.87(0.92)
Banana	93.1(1.93)	90.94(1.79)	91.12(1.92)	87.52(1.9)	**91.48(2.04)**	88.2(1.87)	88.48(1.88)	86.6(1.71)	89.5(1.68)
Banknote	99.87(0.22)	99.43(0.53)	99.43(0.69)	98.82(0.56)	**99.45(0.68)**	99.2(0.47)	99.2(0.54)	98.85(0.55)	99.34(0.5)
Blood	89.6(2.42)	76.74(2.24)	76.66(2.07)	78.24(1.6)	77.62(2.46)	77.67(1.9)	77.99(1.51)	**78.37(1.55)**	76.55(2.64)
Breast	98.99(0.56)	96.61(1.67)	96.15(1.88)	96.78(1.49)	96.12(1.76)	96.71(1.35)	**96.82(1.44)**	96.75(1.37)	96.61(1.61)
Car	83.52(1.06)	73.85(1.24)	73.62(1.27)	71.25(1.1)	**74.48(1.04)**	72.31(0.79)	73.76(0.91)	71.27(1.07)	74.11(1.25)
Datausermodeling	99.06(1.19)	**91.44(2.99)**	89.21(3.82)	88.22(3.15)	89.55(3.32)	90.35(2.75)	90.64(2.77)	88.47(3.15)	91.29(3.63)
Faults	91.54(1.21)	69.97(1.54)	69.12(2.17)	69.6(2.13)	69.42(2.16)	**70.51(2.01)**	70.0(2.08)	69.7(2.15)	70.38(2.03)
German	94.54(1.45)	74.46(2.75)	73.86(2.35)	75.08(1.87)	73.92(2.9)	**75.22(1.96)**	74.94(1.93)	75.04(2.0)	74.86(2.27)
Haberman	92.4(2.06)	70.71(4.32)	70.84(4.2)	73.9(3.15)	71.36(4.24)	74.03(2.6)	72.21(3.3)	**74.16(2.75)**	71.56(3.98)
Heart	97.35(2.01)	82.35(4.26)	81.47(4.42)	**83.9(3.93)**	82.65(4.76)	83.75(3.96)	83.24(3.93)	83.75(3.9)	**83.9(4.42)**
ILPD	95.68(1.62)	70.31(3.04)	69.97(2.88)	**72.09(2.08)**	70.75(3.07)	70.89(1.57)	71.3(2.48)	71.64(1.72)	70.65(2.59)
Ionosphere	98.18(1.32)	**89.32(1.66)**	86.53(2.39)	88.07(2.02)	86.36(2.49)	88.24(2.25)	88.58(2.45)	87.95(2.47)	88.13(1.37)
Laryngeal1	95.28(3.82)	82.41(3.68)	82.13(4.15)	82.41(4.08)	82.04(4.02)	82.5(4.2)	82.13(4.73)	**82.59(4.07)**	82.5(3.49)
Laryngeal3	89.38(2.65)	71.18(3.26)	71.24(4.73)	71.29(3.73)	71.29(4.8)	71.4(3.67)	70.9(3.59)	71.18(3.49)	**71.8(4.08)**
Lithuanian	93.47(1.78)	91.1(2.1)	**91.1(2.45)**	86.9(2.0)	90.9(2.31)	88.6(2.34)	87.93(1.87)	86.37(1.89)	90.5(2.32)
Liver	97.93(1.91)	68.16(4.45)	67.64(3.26)	68.91(4.58)	69.66(3.96)	**70.86(4.23)**	68.74(3.94)	68.85(4.6)	69.14(4.34)
Mammographic	90.29(2.25)	77.81(2.77)	77.55(2.94)	79.04(2.5)	78.51(2.14)	**79.42(2.45)**	78.49(2.38)	79.13(2.55)	78.87(2.73)
Monk2	97.31(1.01)	86.99(3.46)	85.69(3.33)	79.44(3.63)	84.81(3.01)	83.06(3.53)	**88.75(3.99)**	81.3(3.69)	87.64(3.24)
Phoneme	87.75(1.75)	**79.81(1.01)**	79.62(1.03)	77.49(0.92)	79.09(1.03)	77.77(0.81)	79.51(1.05)	77.28(0.89)	78.1(0.91)
Pima	92.97(1.79)	76.12(2.53)	75.42(2.72)	77.03(2.08)	75.49(2.51)	76.54(2.45)	76.59(2.47)	**77.21(2.08)**	76.28(2.72)
Sonar	98.85(1.54)	78.08(5.21)	77.02(4.61)	77.31(5.69)	76.63(5.17)	77.98(5.49)	**80.0(5.28)**	77.4(5.83)	79.62(5.42)
Statlog	94.14(1.53)	74.72(2.42)	74.48(1.97)	75.22(2.33)	74.64(2.2)	75.42(2.08)	**75.82(2.1)**	75.42(2.43)	75.08(2.0)
Steel	91.55(1.44)	70.67(1.45)	69.9(1.94)	70.3(1.81)	70.02(1.68)	**71.65(1.81)**	71.05(1.77)	70.64(1.92)	71.37(1.33)
Thyroid	98.47(0.88)	95.84(1.43)	95.9(1.34)	95.9(1.37)	95.78(1.17)	95.78(1.2)	95.95(1.28)	95.84(1.34)	**95.98(1.37)**
Vehicle	96.89(1.02)	74.98(2.17)	74.83(2.5)	74.79(1.67)	74.25(2.07)	74.55(2.47)	74.81(2.24)	74.5(1.82)	**75.05(2.41)**
Vertebral	95.96(2.93)	83.33(4.05)	83.08(3.88)	82.44(4.27)	83.27(3.35)	**84.1(4.82)**	83.33(3.85)	83.01(4.42)	84.04(3.23)
Voice3	92.67(2.32)	77.0(3.32)	76.92(3.62)	78.58(2.85)	77.08(3.61)	77.83(2.79)	77.83(3.42)	**78.75(2.58)**	76.58(3.09)
Weaning	97.43(2.06)	81.32(4.26)	80.92(4.51)	80.99(4.6)	81.91(4.9)	**82.5(4.25)**	81.18(4.37)	80.92(4.66)	82.43(4.39)
Wine	99.78(0.67)	97.89(1.79)	96.56(3.02)	97.78(1.86)	96.67(2.77)	**98.22(1.66)**	97.78(2.11)	97.89(1.92)	98.0(1.71)
Average	94.78	81.68	81.16	81.18	81.41	81.73	81.82	81.24	**81.89**
Ave Rank	-	3.32	5.1	3.88	4.18	**2.42**	2.85	3.78	2.47

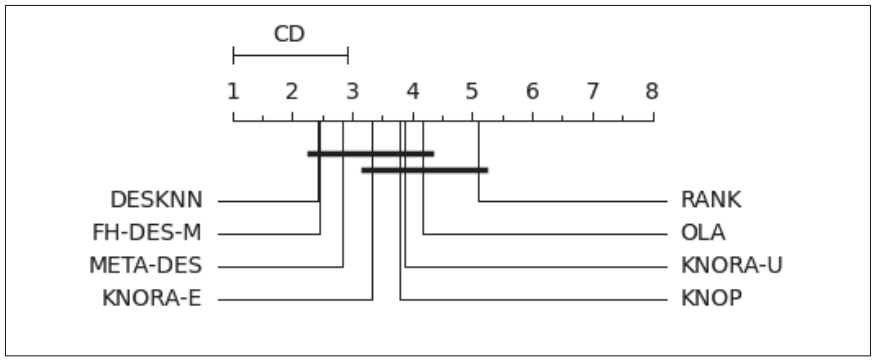


Figure-A I-11 Critical Difference (CD) diagram considering the all compared approaches. The best algorithm is the one presenting the lowest rank and techniques that are statistically equivalent are connected by a black bar

between these techniques, we conducted a pairwise comparison between FH-DES-M and the state-of-the-art DS techniques using the Sign test. The result of this test is shown in Figure I-12.

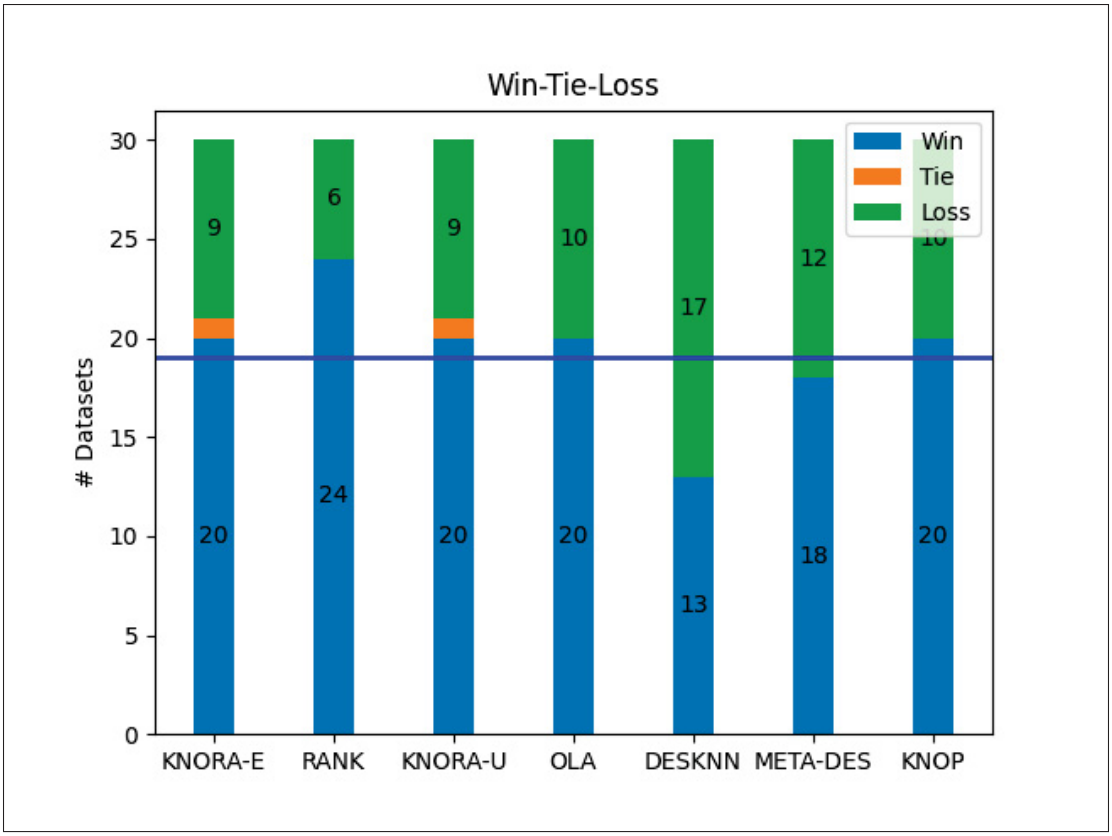


Figure-A I-12 Pairwise comparison between the FH-DES-M and other DS methods (n_c = 19.5)

As we can see in this figure, the proposed approach significantly surpasses five out of seven state-of-the-art DS approaches. And it is slightly better than META-DES. Just the DESKNN approach has a higher number of wins than FH-DES-M (17 losses for the FH-DES-M). But, since the difference is lower than the threshold n_c = 19.5, this difference is not statistically significant.

5.4 Time complexity and Memory Cost

As we discussed earlier, in the KNN-based approaches, to label each query sample, its distance to all DSEL samples is calculated, and then the K nearest samples are determined. If we used big O notation that represents an upper bound to show how run time grows as the input size grows (Knuth (1976)), KNN costs $O(n)$, which n is the number of instances in DSEL. However, the computational complexity in the proposed approach changes to $O(e)$ which e is the number

of hyperboxes. We expect that the number of generated hyperboxes be much smaller than the size of the DSEL data ($e << n$), significantly reducing the computational cost of our system. To validate this hypothesis, we analyze the number of hyperboxes generated by our system considering two large-scale datasets: i) the *Sensor* dataset (Asuncion & Newman (2007)) which contains 928K samples as a binary classification problem. ii) An artificial dataset, namely *ArData* generated with Scikit-learn, which includes five features and two classes. We vary the DSEL size from 1K to 900K to examine the influence of its size on the number of generated hyperboxes. In each step, we added new samples to the DSEL data. In this experiment, two small subsets of the dataset are selected randomly as train and test data so that each of them contains 1000 samples. During all steps of this experiment, there was no overlap between the test, train, and DSEL datasets. The pool of classifiers contained 100 perceptrons, similar to the previous experiments.

Since the complexity of all KNN-based approaches is the same ($O(n)$), during this experiment, we examined only the META-DES approach as a strong and accurate KNN-based approach (Cruz *et al.* (2018a)). In Table I-5, the accuracy obtained using FH-DES-M and META-DES in different sizes of the DSEL data is reported.

This table shows that the proposed method and META-DES have similar accuracy in small data sizes. However, with a larger data size, the accuracy of the KNN-based approach increases. In particular, the accuracy of the META-DES gets close to the oracle's accuracy for the Sensor dataset as the dataset size increases. Because, in large-scale datasets, almost all regions of the feature space are covered by DSEL instances. Thus, sufficient DSEL samples are available to correctly estimate the base classifiers' competencies and select a suitable ensemble of classifiers. However, achieving such accuracy has a large computational complexity. In table I-6, the number of hyperboxes generated by FH-DES-M and the number of DSEL samples are reported.

As presented in table I-6, the number of generated hyperboxes is considerably smaller than the size of the DSEL data ($e << n$). For example, considering the 900K samples study case, the proposed approach generates only 4,127 and 4,392 hyperboxes for the ArData and Sensor

Table-A I-5 The accuracy of the proposed approach and META-DES in different data sizes

DataSets	Oracle	META-DES	FH-DES-M
Data1000	94.01(0.19)	90.78(0.07)	90.99(0.11)
Data10000	94.01(0.19)	91.06(0.14)	90.74(0.13)
Data100000	94.01(0.19)	91.74(0.09)	90.84(0.19)
Data300000	94.01(0.19)	91.99(0.08)	90.84(0.17)
Data500000	94.01(0.19)	92.07(0.15)	90.82(0.09)
Data700000	94.01(0.19)	92.06(0.17)	90.80(0.10)
Data900000	94.01(0.19)	92.06(0.16)	90.86(0.14)
Sensor1000	99.09(0.14)	97.68(0.2)	97.26(0.4)
Sensor10000	99.09(0.14)	98.21(0.19)	97.4(0.17)
Sensor100000	99.09(0.14)	98.75(0.16)	96.61(0.38)
Sensor300000	99.09(0.14)	98.78(0.16)	96.39(0.25)
Sensor500000	99.09(0.14)	98.84(0.15)	96.36(0.23)
Sensor700000	99.09(0.14)	98.84(0.16)	96.36(0.28)
Sensor900000	99.09(0.14)	98.87(0.15)	96.38(0.29)
Average Accuracy	96.74	95.37	93.98

Table-A I-6 Comparing the number of generated Hyperboxes in FH-DES-M and number of DSEL samples

#Sample	# Hyperboxes	
	ArData	Sensor
1,000	530	1,115
10,000	1,313	2,407
100,000	2,679	3,849
300,000	3,457	4,167
500,000	3,818	4,290
700,000	4,026	4,331
900,000	4,127	4,392

datasets, respectively. In addition, we can also observe a plateau in the number of hyperboxes added to the system as the dataset size increases. Between 700k to 900k samples, only 101 and 61 new hyperboxes were added to the system. On the other hand, all 900K samples should be stored and processed for each query sample using KNN-based approaches. Thus, we can confirm that the proposed approach has lower computational complexity than KNN-based DES approaches from the storage and computational perspectives.

6. Conclusion

This paper introduced a novel dynamic ensemble selection framework based on fuzzy hyperbox. In this framework, competence or incompetence areas of classifiers are determined using fuzzy hyperboxes. For each base classifier, hyperboxes are formed based on their correct-classified samples to define the "competencies" and "incompetencies" areas. In addition, for the first time in the dynamic selection area, the misclassified instances were applied to define the incompetence areas. Moreover, this paper also introduced a new membership function to measure the memberships differently with softer boundaries that slightly increase the system's accuracy.

Experimental results demonstrated that utilizing the misclassified samples could significantly increase the accuracy of the proposed DS framework and decrease the computational complexity compared to the correct classified samples. Additionally, the proposed approach based on misclassified samples obtained the highest average accuracy compared to state-of-the-art DS approaches.

Furthermore, the proposed framework also has lower storage and computational complexity when compared to DS techniques based on KNN and potential function models for estimating the regions of competence. According to the experimental results, the proposed method generates about 4k hyperboxes (40 hyperboxes for modeling each base classifier's misclassifications) in datasets containing 900k samples. Thus, the proposed FH-DES can be an excellent alternative for handling large-scale problems with DES approaches. Fuzzy hyperboxes also allow online learning making it a suitable technique for handling streaming data. Future works will investigate the use of FH-DES for dealing with data streams and concept drift.

APPENDIX II

SUPPLEMENTARY MATERIALS OF JOURNAL PAPER

1. Hyperparameter Tuning

As mentioned earlier, the best hyperparameters values of these variants are estimated using a tuning process on the first 10 datasets in Table 2.4. The best values obtained from this process are reported in Table II-1.

Table-A II-1 Best hyperparameter values found for different variants of FH_DES

	FH_1-M	**FH_2-M**	**FH_3-M**	**FH_4-M**	**FH_5-M**	**FH_6-M**	**FH_7-M**	**FH_8-M**	**FH_9-M**	**FH_10-M**
θ	0.27	-	0.5	-	0.7	0.7	0.4	0.5	0.7	0.9
μ	0.99	0.4	0.3	0.1	0.1	0.5	0.4	0.4	0.7	0.7
	FH_1-C	**FH_2-C**	**FH_3-C**	**FH_4-C**	**FH_5-C**	**FH_6-C**	**FH_7-C**	**FH_8-C**	**FH_9-C**	**FH_10-C**
θ	0.27	-	0.8	-	0.8	1.0	0.8	1.0	1.0	0.8
μ	0.99	0.4	0.1	0.4	0.3	0.4	0.1	0.1	0.2	0.6

2. Ablation Study

Obtained accuracy of all positive variants for each specific dataset are reported in Table II-2. Then the results of the negative variants per dataset are presented in Table II-3.

Table-A II-2 Average accuracy and standard deviation of positive variants (based on correct classified samples)

DataSets	FH_1-C	FH_2-C	FH_3-C	FH_4-C	FH_5-C	FH_6-C	FH_7-C	FH_8-C	FH_9-C	FH_10-C
Audit	**96.87(0.74)**	96.09(1.48)	95.91(1.47)	96.04(1.48)	95.93(1.4)	95.98(1.44)	95.91(1.47)	95.88(1.45)	95.91(1.5)	96.11(1.19)
Banana	**89.12(2.37)**	87.3(1.9)	86.24(1.86)	87.22(1.98)	86.74(1.98)	86.18(1.96)	86.44(1.85)	86.28(1.78)	86.92(2.0)	86.16(1.83)
Banknote	**99.13(0.66)**	98.8(0.57)	98.67(0.57)	98.8(0.58)	98.79(0.55)	98.69(0.64)	98.76(0.55)	98.72(0.61)	98.8(0.58)	98.7(0.54)
Blood	77.46(1.98)	78.48(1.53)	78.4(1.33)	78.45(1.61)	78.24(1.62)	78.29(1.36)	78.24(1.47)	78.32(1.33)	**78.5(1.55)**	78.29(1.28)
Breast	96.28(1.49)	**96.61(1.22)**	96.4(1.28)	**96.61(1.26)**	96.5(1.25)	96.5(1.17)	96.36(1.24)	96.47(1.24)	96.54(1.28)	96.4(1.26)
Car	**74.63(1.08)**	73.63(1.21)	70.47(0.96)	73.6(1.29)	71.28(1.15)	71.31(1.19)	70.47(0.96)	70.46(1.06)	72.25(1.2)	70.52(1.12)
Datausermodeling	87.77(3.93)	**88.47(2.68)**	87.57(2.79)	87.67(2.77)	87.77(2.96)	87.82(2.75)	87.72(2.81)	87.62(3.04)	87.67(2.77)	87.57(2.53)
Faults	**70.12(1.84)**	69.15(2.4)	68.99(2.4)	69.03(2.34)	69.0(2.29)	69.06(2.33)	68.99(2.4)	68.98(2.36)	69.0(2.36)	69.06(2.44)
German	74.8(2.0)	74.68(1.9)	74.96(2.15)	**75.16(2.15)**	74.9(2.22)	74.98(2.19)	74.96(2.15)	74.96(2.15)	74.98(2.26)	75.06(2.14)
Haberman	71.36(3.88)	74.48(1.98)	**75.32(2.25)**	74.68(2.3)	74.87(1.89)	74.42(1.65)	75.13(2.33)	75.13(2.11)	74.94(2.22)	75.13(2.44)
Heart	82.79(3.89)	83.16(4.09)	83.01(4.25)	**83.38(4.11)**	83.24(4.25)	83.01(4.47)	83.01(4.25)	83.01(4.25)	83.31(4.17)	82.87(4.22)
ILPD	71.3(3.1)	71.95(2.72)	71.95(2.52)	71.92(2.77)	71.92(2.63)	71.95(2.64)	71.95(2.52)	71.92(2.52)	**71.99(2.71)**	71.71(2.64)
Ionosphere	**88.35(2.21)**	87.9(1.94)	87.27(2.61)	87.61(2.38)	87.44(2.55)	87.39(2.27)	87.27(2.61)	87.44(2.68)	87.61(2.38)	87.56(2.31)
Laryngeal1	**81.94(3.25)**	81.02(4.18)	80.56(4.24)	81.02(4.09)	80.56(4.08)	80.93(4.15)	80.56(4.24)	80.83(4.15)	80.83(4.35)	80.46(4.16)
Laryngeal3	**71.01(4.15)**	70.17(3.96)	69.83(3.52)	70.17(3.77)	70.06(3.66)	69.94(3.77)	69.83(3.52)	69.78(3.57)	69.83(3.68)	70.22(3.72)
Lithuanian	**89.63(2.08)**	86.13(1.69)	85.07(1.6)	86.3(1.74)	85.67(1.74)	85.0(1.71)	85.0(1.64)	84.7(1.77)	85.97(1.47)	85.43(1.77)
Liver	67.82(4.77)	67.93(4.48)	67.36(4.41)	**68.22(4.72)**	67.47(4.61)	67.76(4.25)	67.64(4.35)	67.82(4.47)	67.93(4.55)	67.53(4.25)
Mammographic	78.03(3.2)	80.94(2.78)	80.96(2.65)	81.13(2.57)	**81.15(2.55)**	81.06(2.57)	80.99(2.64)	81.06(2.53)	80.99(2.6)	80.91(2.62)
Monk2	86.2(2.85)	86.39(3.37)	80.79(3.35)	**86.94(3.3))**	82.31(3.14)	82.55(3.3)	80.79(3.35)	80.79(3.35)	81.76(2.69)	81.53(2.53)
Phoneme	**77.81(0.97)**	77.01(0.9)	76.75(0.92)	76.83(0.93)	76.8(0.94)	76.79(0.94)	76.75(0.97)	76.76(0.92)	76.84(0.91)	76.79(0.9)
Pima	74.9(2.46)	76.88(1.88)	76.9(1.73)	76.93(1.75)	76.88(1.68)	76.95(1.91)	**77.03(1.76)**	76.95(1.81)	76.93(1.83)	76.95(1.75)
Sonar	**81.35(6.41)**	76.63(6.0)	76.63(6.5)	76.83(6.19)	76.83(6.45)	76.54(6.19)	76.63(6.5)	76.92(6.29)	77.02(6.3)	77.02(5.97)
Statlog	75.26(1.91)	75.26(2.02)	75.2(2.3)	75.3(2.19)	75.18(2.13)	75.18(2.17)	75.2(2.3)	75.16(2.12)	75.1(2.13)	**75.5(2.47)**
Steel	**70.72(1.67)**	70.06(1.82)	69.86(1.84)	69.86(1.76)	69.94(1.82)	69.85(1.82)	69.86(1.84)	69.85(1.75)	69.89(1.81)	69.95(1.65)
Thyroid	**95.87(1.32)**	95.55(1.33)	95.52(1.29)	95.61(1.19)	95.46(1.26)	95.43(1.29)	95.52(1.29)	95.49(1.23)	95.58(1.24)	95.49(1.31)
Vehicle	74.53(2.1)	**75.52(2.23)**	**75.52(2.23)**	75.42(2.25)	75.47(2.31)	75.42(2.31)	75.5(2.28)	75.35(2.17)	75.38(2.2)	**75.52(2.16)**
Vertebral	82.44(3.54)	84.33(3.74)	84.33(3.74)	84.27(3.86)	84.13(3.98)	84.2(3.71)	84.13(4.02)	84.2(3.66)	84.33(3.84)	**84.47(3.96)**
Voice3	77.17(3.42)	77.42(3.18)	**77.58(2.71)**	77.33(2.91)	77.08(3.37)	77.08(3.07)	**77.58(2.71)**	**77.58(2.91)**	77.33(3.18)	77.0(2.72)
Weaning	**81.25(4.94)**	80.39(4.12)	80.46(4.21)	80.2(4.0)	80.53(4.13)	80.79(4.31)	80.39(4.18)	80.53(3.92)	80.53(4.0)	80.53(4.03)
Wine	97.11(2.23)	97.33(1.94)	97.0(1.9)	97.11(2.0)	96.67(1.92)	**97.67(2.27)**	97.22(1.97)	97.33(1.94)	97.33(1.94)	97.56(2.1)
Average Accuracy	**81.43**	81.32	80.85	81.32	80.96	80.96	80.86	80.88	81.07	80.93
Average Rank	3.73	**2.73**	5.83	3.07	5.12	4.9	5.73	5.72	3.55	4.62

Table-A II-3 Average accuracy and standard deviation of negative variants (based on miss classified samples)

DataSets	FH_1-M	FH_2-M	FH_3-M	FH_4-M	FH_5-M	FH_6-M	FH_7-M	FH_8-M	FH_9-M	FH_10-M
Audit	96.87(0.92)	97.2(1.16)	97.05(1.11)	97.07(1.21)	97.05(1.17)	97.15(1.15)	97.2(1.0)	97.07(1.04)	96.97(1.0)	**97.25(1.1)**
Banana	89.5(1.68)	89.88(1.83)	90.08(2.91)	89.76(2.24)	86.7(1.92)	87.04(2.06)	87.74(2.28)	86.74(2.11)	**91.14(2.49)**	88.5(2.68)
Banknote	99.34(0.5)	99.34(0.41)	99.37(0.45)	99.26(0.44)	99.2(0.41)	99.15(0.49)	99.3(0.45)	99.17(0.47)	99.46(0.46)	**99.48(0.45)**
Blood	76.55(2.64)	77.94(1.65)	76.95(1.23)	77.99(1.54)	78.18(1.43)	78.29(1.48)	78.07(1.77)	**78.45(1.26)**	77.38(1.88)	76.74(1.44)
Breast	96.61(1.61)	96.85(1.12)	**96.92(1.26)**	96.82(1.3)	96.71(1.35)	96.68(1.25)	96.82(1.24)	96.89(1.28)	96.57(1.38)	96.78(1.35)
Car	74.11(1.25)	73.8(1.67)	73.77(1.73)	79.46(2.0)	78.62(1.71)	77.33(1.67)	73.77(1.73)	73.77(1.73)	79.12(2.15)	**83.45(1.42)**
Datausermodeling	**91.29(3.63)**	88.66(2.76)	88.51(2.66)	88.76(2.45)	88.37(2.6)	88.51(2.49)	88.76(2.7)	88.61(2.74)	90.69(2.49)	90.2(2.3)
Faults	**70.38(2.03)**	69.96(2.17)	69.56(2.33)	69.6(2.28)	69.35(2.23)	69.9(2.34)	69.77(2.25)	69.75(2.29)	70.16(2.18)	70.12(2.05)
German	74.86(2.27)	75.06(2.06)	75.1(2.07)	**75.42(2.33)**	75.14(2.04)	75.06(2.17)	75.18(2.04)	75.12(2.06)	74.98(2.19)	75.08(2.28)
Haberman	71.56(3.98)	74.81(2.48)	74.09(2.77)	75.39(2.77)	75.06(2.8)	**75.52(2.14)**	75.06(2.12)	**75.52(2.02)**	74.42(2.43)	73.44(2.61)
Heart	83.09(4.42)	**83.38(4.24)**	83.01(4.37)	83.31(4.24)	83.16(4.22)	83.01(4.52)	82.79(4.26)	82.87(4.44)	82.72(3.86)	82.5(3.78)
ILPD	70.65(2.59)	72.12(2.46)	72.16(2.15)	**72.4(2.75)**	71.92(1.71)	72.05(1.67)	72.09(1.5)	72.12(2.34)	72.26(2.08)	72.12(3.57)
Ionosphere	88.13(1.37)	88.18(1.98)	87.84(2.3)	88.24(2.01)	87.9(2.31)	87.9(2.25)	87.78(2.69)	87.84(2.3)	**88.75(1.83)**	88.3(2.1)
Laryngeal1	82.5(3.49)	82.87(3.74)	82.69(4.07)	82.96(4.21)	82.69(4.51)	82.69(4.27)	82.96(4.04)	83.06(3.81)	83.06(3.98)	**83.43(4.03)**
Laryngeal3	**71.8(4.08)**	71.01(3.55)	70.73(3.23)	71.12(3.39)	70.73(3.38)	71.29(4.29)	71.12(3.2)	70.9(3.41)	71.69(3.8)	71.69(3.01)
Lithuanian	**90.5(2.32)**	89.47(2.56)	75.87(10.44)	89.17(2.59)	86.37(1.95)	86.5(2.28)	87.93(2.63)	86.7(2.2)	90.4(2.54)	71.7(8.35)
Liver	69.14(4.34)	68.74(5.12)	67.01(4.35)	68.97(4.48)	67.76(4.07)	67.13(4.62)	68.68(4.74)	67.87(4.32)	**69.66(3.9)**	69.2(3.24)
Mammographic	78.87(2.73)	79.76(2.62)	70.77(7.45)	79.74(2.67)	72.21(8.79)	**80.43(3.2)**	71.97(6.61)	77.74(5.98)	79.42(2.86)	71.61(6.99)
Monk2	**87.64(3.24)**	82.64(3.38)	79.86(3.22)	87.59(3.37)	80.56(3.08)	80.56(3.08)	81.44(3.36)	79.95(3.26)	80.65(3.5)	81.44(3.7)
Phoneme	78.1(0.91)	78.6(0.87)	74.19(0.94)	77.88(1.04)	76.87(0.93)	76.85(0.92)	77.29(1.08)	76.95(1.0)	**79.12(1.15)**	76.02(1.35)
Pima	76.28(2.72)	77.01(2.19)	76.85(2.32)	77.08(2.02)	77.06(1.9)	**77.34(2.3)**	76.95(2.13)	76.98(2.09)	76.56(2.54)	77.08(2.32)
Sonar	79.62(5.42)	78.85(6.14)	79.33(6.55)	78.08(6.0)	78.75(6.1)	79.71(5.78)	79.71(6.62)	79.62(6.65)	**79.9(5.46)**	79.71(5.66)
Statlog	75.08(2.0)	75.44(2.11)	75.24(2.24)	75.48(2.25)	75.28(2.23)	75.3(2.25)	75.42(2.26)	75.32(2.27)	**75.72(2.36)**	75.42(2.2)
Steel	**71.37(1.33)**	70.58(2.1)	70.24(1.89)	70.53(2.13)	70.17(1.88)	70.71(2.16)	70.44(2.06)	70.3(1.92)	70.93(2.06)	70.78(2.16)
Thyroid	95.98(1.37)	96.18(1.1)	96.04(1.12)	96.18(1.1)	95.98(1.09)	96.27(1.26)	96.13(1.24)	96.13(1.17)	96.21(1.56)	**96.33(1.35)**
Vehicle	75.05(2.41)	75.83(2.23)	**75.87(1.97)**	75.83(1.78)	75.8(2.37)	75.75(2.22)	75.57(1.94)	75.71(2.04)	75.83(2.23)	75.61(2.09)
Vertebral	84.04(3.23)	**85.07(4.33)**	84.4(3.63)	84.4(3.96)	84.0(4.04)	84.6(4.03)	84.73(3.71)	84.93(4.2)	**85.07(3.83)**	83.87(3.29)
Voice3	76.58(3.09)	78.33(3.16)	77.58(3.89)	**78.5(3.02)**	77.58(3.43)	77.25(3.77)	77.83(3.21)	77.92(3.61)	77.0(3.52)	77.83(3.46)
Weaning	**82.43(4.39)**	81.18(3.72)	81.18(4.06)	81.05(3.96)	81.05(4.19)	80.79(3.85)	81.32(3.71)	81.38(4.04)	81.84(4.09)	81.78(3.93)
Wine	98.0(1.71)	98.56(1.45)	98.44(1.59)	98.44(1.59)	98.44(1.59)	98.56(1.27)	**98.67(1.3)**	**98.67(1.3)**	98.0(1.39)	98.11(1.45)
Average	81.89	81.91	80.69	**82.22**	81.29	81.64	81.42	81.47	82.19	81.19
Average Rank	5.15	3.3	6.07	3.27	6.18	4.82	4.55	4.68	**3.13**	3.85

The number of hyperboxes generated by all variants of FH_DES over the 30 datasets is reported in Tables II-4 and II-5 for the positive and negative versions, respectively.

Table-A II-4 Average Number of generated Hyperboxes in positive variants over the 30 small datasets

DataSets	NO. Samples	FH_1-C	FH_2-C	FH_3-C	FH_4-C	FH_5-C	FH_6-C	FH_7-C	FH_8-C	FH_9-C	FH_10-C
Audit	193.0	3640.40	1321.15	2810.60	**643.10**	2876.60	2415.55	2810.60	2136.30	2223.70	18594.70
Banana	250.0	525.80	1093.90	374.10	877.10	300.45	146.85	377.75	**142.75**	877.10	2680.10
Banknote	343.0	668.85	817.05	510.20	489.60	445.65	240.70	570.15	**212.25**	514.00	6001.05
Blood	187.0	420.20	5290.15	379.55	4046.10	370.00	238.40	397.20	**228.90**	4047.30	7011.30
Breast	142.0	1881.65	711.10	902.80	**421.60**	1023.35	623.20	914.45	575.85	723.20	13064.90
Car	432.0	7217.85	5978.35	3952.20	**2906.55**	3908.10	3907.85	3952.20	3964.65	5893.35	30450.35
Datausermodeling	101.0	1651.65	1200.60	646.95	810.95	812.55	219.05	632.10	**195.80**	816.30	5949.00
Faults	485.0	8392.15	4547.60	3577.55	2839.40	4101.10	1477.05	3577.55	**877.25**	3058.00	32700.80
German	250.0	15232.35	4432.15	11284.70	**2196.05**	11748.95	11686.70	11284.70	11213.35	11304.95	18487.40
Haberman	76.0	579.60	1568.15	341.00	1102.45	357.00	235.65	360.50	**228.85**	1101.80	4419.00
Heart	67.0	3264.65	1103.90	2400.65	**686.70**	2445.80	2398.05	2400.65	2350.80	2383.50	5389.15
ILPD	146.0	1305.10	2336.25	754.55	1229.55	757.50	**473.70**	754.55	497.20	1515.55	10228.85
Ionosphere	88.0	2483.80	1037.45	1872.10	**525.40**	1957.00	1533.70	1872.10	1356.55	1445.55	7412.45
Laryngeal1	53.0	1213.80	666.65	767.10	**442.50**	792.80	481.45	767.10	467.70	721.30	4372.20
Laryngeal3	88.0	1035.95	971.50	718.90	655.00	681.25	359.35	719.00	**330.60**	761.95	6327.40
Lithuanian	150.0	478.60	955.00	306.60	720.50	286.60	137.45	284.15	**136.70**	720.45	2214.30
Liver	86.0	597.45	1479.45	321.50	927.75	323.55	228.00	319.25	**224.25**	965.40	4860.80
Mammographic	207.0	1067.25	5102.85	482.40	3053.40	504.80	432.10	482.40	**427.90**	3189.95	16651.00
Monk2	108.0	3358.20	2292.80	2809.80	**1109.45**	2788.35	2788.35	2809.80	2809.80	3132.70	8592.50
Phoneme	1351.0	1051.25	10896.20	481.80	6779.75	604.00	172.15	477.35	**161.60**	6779.40	26842.50
Pima	192.0	1474.85	2709.40	647.85	1638.35	703.85	354.00	621.85	**318.55**	1690.75	13378.70
Sonar	52.0	2230.00	241.35	1083.55	**186.05**	1342.20	731.10	1083.55	566.55	570.25	3843.95
Statlog	250.0	13158.65	4140.20	6746.50	**2137.10**	7956.70	7750.55	6746.50	6524.30	6717.50	18473.45
Steel	485.0	7481.90	4658.05	3425.60	2849.60	3882.25	1494.95	3425.60	**843.40**	3167.90	32865.50
Thyroid	173.0	1901.15	999.05	1128.20	519.15	1167.70	561.65	1128.00	**508.75**	845.70	16412.40
Vehicle	211.0	1738.75	2396.45	863.60	1649.25	921.85	384.95	899.70	**287.15**	1673.10	14542.85
Vertebral	75.0	617.30	930.45	350.85	624.35	346.85	211.45	356.20	**209.65**	680.15	4062.70
Voice3	59.0	1141.05	806.40	783.65	554.25	796.30	552.15	783.65	**493.15**	730.50	4548.25
Weaning	75.0	2047.25	1059.50	729.45	534.80	1022.05	484.75	728.85	**385.00**	649.10	5942.55
Wine	44.0	1177.75	204.15	630.20	**171.85**	648.50	364.80	629.50	290.30	309.70	4241.50
Average	213.97	2967.84	2398.24	1736.15	1444.26	1862.45	1436.19	1738.9	**1298.86**	2307.	11685.39
Average Rank	-	7.97	6.43	4.80	3.83	5.80	2.93	4.83	**1.87**	5.93	10.00

Table-A II-5 Average Number of generated Hyperboxes in negative variants over the 30 small datasets

DataSets	NO. Samples	FH_1-M	FH_2-M	FH_3-M	FH_4-M	FH_5-M	FH_6-M	FH_7-M	FH_8-M	FH_9-M	FH_10-M
Audit	193.0	343.60	230.50	490.65	**202.20**	325.20	325.20	523.40	490.65	341.80	705.35
Banana	250.0	212.45	618.00	241.25	576.75	149.25	**144.50**	316.85	224.80	576.75	2852.20
Banknote	343.0	116.10	234.70	132.75	227.75	**115.20**	**115.20**	189.25	129.45	230.25	553.90
Blood	187.0	265.85	2826.40	350.15	2473.60	**249.25**	249.40	448.05	357.75	2476.05	3022.35
Breast	142.0	190.85	122.95	218.50	**119.70**	170.40	170.35	263.50	217.60	176.90	474.50
Car	432.0	3583.50	3333.15	6228.30	2422.85	**1614.05**	1614.30	6228.30	6228.30	2935.40	12749.65
Datausermodeling	101.0	524.15	587.55	652.50	496.65	388.45	**376.45**	809.30	641.25	552.85	1606.05
Faults	485.0	5355.15	3398.30	7167.00	**2050.30**	3776.40	3776.40	9259.85	7167.00	3801.80	15799.20
German	250.0	5855.15	2463.00	6153.15	**1506.05**	5294.75	5294.75	6245.20	6153.15	5243.40	6512.60
Haberman	76.0	435.20	1072.30	542.15	862.55	344.40	**337.80**	698.50	525.20	904.75	1742.95
Heart	67.0	1127.95	513.60	1161.55	**362.85**	1036.60	1036.60	1247.80	1161.55	1035.10	1310.85
ILPD	146.0	625.20	1288.90	778.20	888.45	**445.05**	**445.05**	1062.75	778.20	972.90	4371.15
Ionosphere	88.0	962.25	484.75	1085.15	**317.75**	841.35	841.35	1182.65	1085.15	780.90	1387.55
Laryngeal1	53.0	189.55	230.85	230.75	191.85	**150.00**	**150.00**	331.50	230.80	208.35	923.80
Laryngeal3	88.0	819.30	689.15	1002.75	**562.60**	671.65	667.70	1234.95	1002.35	783.15	2462.90
Lithuanian	150.0	249.55	572.10	297.40	547.45	208.15	**202.70**	344.40	267.85	537.50	1426.40
Liver	86.0	430.70	1095.10	567.95	780.45	317.95	**314.25**	743.45	548.20	834.95	2926.20
Mammographic	207.0	615.35	2807.55	731.50	2246.60	435.60	**432.55**	920.20	731.30	2359.35	4043.40
Monk2	108.0	1306.10	975.75	1870.70	**661.80**	1171.00	1171.00	2154.35	1870.70	1382.75	2207.50
Phoneme	1351.0	804.55	8241.45	1359.95	5727.40	587.65	**570.35**	2351.70	1401.80	5791.65	18815.45
Pima	192.0	856.75	1548.35	1112.00	1038.85	**657.85**	665.70	1548.95	1090.10	1159.45	4585.05
Sonar	52.0	919.50	112.65	1029.40	**111.55**	759.50	759.35	1181.60	1029.40	651.35	1356.05
Statlog	250.0	5218.10	2408.20	5974.95	**1475.20**	3977.30	3977.30	6164.15	5974.95	3666.30	6526.55
Steel	485.0	5188.45	3420.30	6850.20	**2081.05**	3656.05	3656.05	8556.85	6850.20	3706.40	15634.50
Thyroid	173.0	112.60	185.95	164.75	163.55	**102.85**	**102.85**	236.85	164.75	164.45	764.65
Vehicle	211.0	1169.80	1443.40	1373.25	1126.55	888.90	**880.65**	1854.60	1370.80	1271.90	5748.40
Vertebral	75.0	185.20	441.55	232.70	351.95	134.95	**134.90**	288.25	227.25	354.30	1179.95
Voice3	59.0	557.90	483.85	623.95	**359.05**	525.00	525.00	730.25	623.95	580.25	1351.75
Weaning	75.0	632.40	418.20	725.45	**249.00**	472.35	459.45	973.95	725.15	395.60	1556.55
Wine	44.0	127.90	100.65	142.15	**100.60**	117.40	117.35	154.35	142.15	117.40	171.00
Average	213.97	1299.37	1411.64	1649.7	1009.43	986.15	**983.82**	1941.52	1647.06	1466.46	4158.95
Average Rank	-	4.50	5.47	6.27	3.50	2.57	**2.20**	8.00	5.93	5.53	10.00

3. Results of the Friedman test

3.1 Comparison against the baseline methods

The Friedman test with the Bonferroni-Dunn posthoc test was also used to perform a statistical analysis on the reported results. This test is applied to compare the ranks achieved by each DS

method. The best algorithm is the one that presents the best rank. The performance of the two DS approaches is significantly different if their difference in average rank is greater than the CD value. The techniques that are statistically equivalent are connected by a black bar. The result of this test for the proposed framework and baseline methods is represented in Figure II-1 using the Critical Difference (CD) diagram. According to the results of this test, the selected variant (FH_4) has the best rank compared to the majority voting and the single-best approaches. Also, as we can observe in Figure II-1, the average accuracy of FH_4 is significantly better than the majority-voting and GFMM approaches.

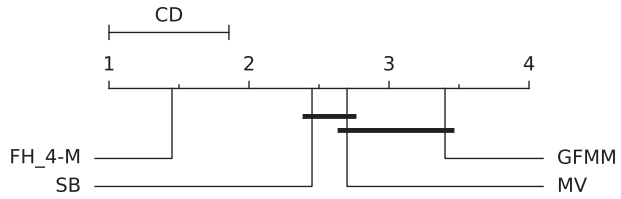


Figure-A II-1 Comparing the selected variants and baseline approaches using the Bonferroni-Dunn post-hoc test. The best algorithm is the one that presents the lowest rank, and the techniques that are statistically equivalent are connected by a black bar.

3.2 Comparison against the state-of-the-art DS methods

We conducted the Friedman test with a post-hoc Bonferroni-Dunn analysis to compare the performance of FH_4-M with the state-of-the-art DS approaches. Figure II-2 presents the results of this test through a critical difference diagram. This figure demonstrates that the proposed FH_4-M has the best rank compared to the state-of-the-art DS approaches. However, based on global analysis, there is no statistically significant difference between the proposed method and some of the other DS approaches across 30 small to medium datasets.

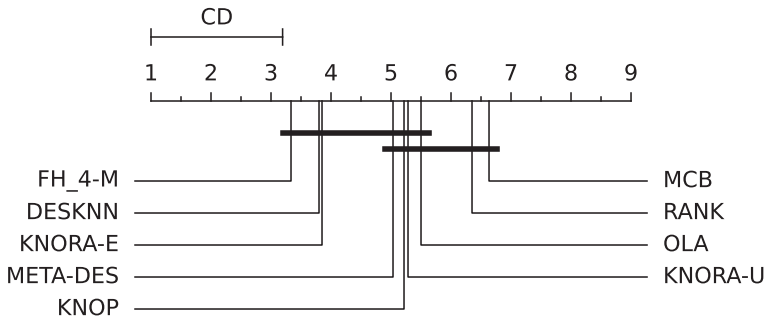


Figure-A II-2 Result of the Bonferroni-Dunn post-hoc test using critical difference diagram including the FH_4-M and other DS approaches. The best algorithm is the one that presents the lowest rank, and the techniques that are statistically equivalent are connected by a black bar.

LIST OF REFERENCES

Ahmed, A. A. & Mohammed, M. F. (2018). SAIRF: A similarity approach for attack intention recognition using fuzzy min-max neural network. *Journal of Computational Science*, 25, 467–473.

Akbulut, D. (2019). Optimization approaches for classification and feature selection using overlapping hyperboxes. Middle East Technical University.

Alhroob, E., Mohammed, M. F., Lim, C. P. & Tao, H. (2019). A Critical Review on Selected Fuzzy Min-Max Neural Networks and Their Significance and Challenges in Pattern Classification. *IEEE Access*, 7, 56129–56146. doi: 10.1109/ACCESS.2019.2911955. Conference Name: IEEE Access.

Allikivi, M.-L. & Kull, M. (2019). Non-parametric bayesian isotonic calibration: Fighting over-confidence in binary classification. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 103–120.

Asuncion, A. & Newman, D. (2007). UCI machine learning repository. Irvine, CA, USA.

Batista, L., Granger, E. & Sabourin, R. (2012). Dynamic selection of generative–discriminative ensembles for off-line signature verification. *Pattern Recognition*, 45(4), 1326–1340. doi: 10.1016/j.patcog.2011.10.011.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123–140.

Britto Jr, A. S., Sabourin, R. & Oliveira, L. E. (2014). Dynamic selection of classifiers—a comprehensive review. *Pattern recognition*, 47(11), 3665–3680.

Brun, A. L., Britto, A. S., Oliveira, L. S., Enembreck, F. & Sabourin, R. (2016). Contribution of data complexity features on dynamic classifier selection. *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 4396–4403.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., Rosen, D. B. et al. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on neural networks*, 3(5), 698–713.

Cavalin, P. R. (2012). LoGID An adaptive framework combining local and global incremental learning for dynamic selection of ensembles of HMMs. *Pattern Recognition*, 13.

Cavalin, P. R., Sabourin, R. & Suen, C. Y. (2013). Dynamic selection approaches for multiple classifier systems. *Neural computing and applications*, 22, 673–688.

Choi, Y.-R. & Lim, D.-J. (2021). DDES: A Distribution-Based Dynamic Ensemble Selection Framework. *IEEE Access*, 9, 40743–40754.

Cruz, R., Sabourin, R. & Cavalcanti, G. (2017a). META-DES.Oracle: Meta-learning and feature selection for dynamic ensemble selection. *Information fusion*, 38, 84–103.

Cruz, R. M., Cavalcanti, G. D. & Ren, T. I. (2011). A method for dynamic ensemble selection based on a filter and an adaptive distance to improve the quality of the regions of competence. *The 2011 International Joint Conference on Neural Networks*, pp. 1126–1133.

Cruz, R. M., Cavalcanti, G. D., Tsang, R. & Sabourin, R. (2013). Feature representation selection based on classifier projection space and oracle analysis. *Expert Systems with Applications*, 40(9), 3813–3827.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2015a). META-DES. H: A dynamic ensemble selection technique using meta-learning and a dynamic weighting approach. *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2015b). A DEEP analysis of the META-DES framework for dynamic selection of ensemble of classifiers. *arXiv preprint arXiv:1509.00825*, 47.

Cruz, R. M., Sabourin, R., Cavalcanti, G. D. & Ren, T. I. (2015c). META-DES: A dynamic ensemble selection framework using meta-learning. *Pattern recognition*, 48(5), 1925–1935.

Cruz, R. M., Sabourin, R., Cavalcanti, G. D. & Ren, T. I. (2015d). META-DES: A dynamic ensemble selection framework using meta-learning. *Pattern recognition*, 48(5), 1925–1935.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2017b). Analyzing different prototype selection techniques for dynamic classifier and ensemble selection. *2017 international joint conference on neural networks (IJCNN)*, pp. 3959–3966.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2018a). Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41, 195–216.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2018b). Prototype selection for dynamic classifier and ensemble selection. *Neural Computing and Applications*, 29, 447–457.

Cruz, R. M., Oliveira, D. V., Cavalcanti, G. D. & Sabourin, R. (2019a). FIRE-DES++: Enhanced online pruning of base classifiers for dynamic ensemble selection. *Pattern Recognition*, 85, 149–160.

Cruz, R. M., Souza, M. A., Sabourin, R. & Cavalcanti, G. D. (2019b). Dynamic ensemble selection and data preprocessing for multi-class imbalance learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 33(11), 1940009.

Cruz, R. M., Hafemann, L. G., Sabourin, R. & Cavalcanti, G. D. (2020). DESlib: A Dynamic ensemble selection library in Python. *The Journal of Machine Learning Research*, 21(1), 283–287.

Dash, S., Shakyawar, S. K., Sharma, M. & Kaushik, S. (2019). Big data in healthcare: management, analysis and future prospects. *Journal of Big Data*, 6(1), 1–25.

Davis, R. I. & Cucu-Grosjean, L. (2019). A survey of probabilistic timing analysis techniques for real-time systems. *LITES: Leibniz Transactions on Embedded Systems*, 1–60.

Davtalab, R., Parchami, M., Dezfoulian, M. H., Mansourizade, M. & Akhtar, B. (2012). M-FMCN: modified fuzzy min-max classifier using compensatory neurons. *Proceedings of the 11th WSEAS international conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, pp. 77–82.

Davtalab, R., Dezfoulian, M. H. & Mansoorizadeh, M. (2013). Multi-level fuzzy min-max neural network classifier. *IEEE Transactions on neural networks and learning systems*, 25(3), 470–482.

Davtalab, R., Cruz, R. M. & Sabourin, R. (2022). Dynamic ensemble selection using fuzzy hyperboxes. *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9.

de Amorim, L. B., Cavalcanti, G. D. & Cruz, R. M. (2023). The choice of scaling technique matters for classification performance. *Applied Soft Computing*, 133, 109924.

de Souto, M. C., Soares, R. G., Santana, A. & Canuto, A. M. (2008). Empirical comparison of dynamic classifier selection methods based on diversity and accuracy for building ensembles. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1480–1487.

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

Dong, X., Yu, Z., Cao, W., Shi, Y. & Ma, Q. (2020). A survey on ensemble learning. *Frontiers of Computer Science*, 14(2), 241–258.

El-Melegy, M. T. & El-Magd, K. M. A. (2019). A Multiple Classifiers System for Automatic Multimodal Brain Tumor Segmentation. *2019 15th International Computer Engineering Conference (ICENCO)*, pp. 110–114.

Elmi, J. & Eftekhari, M. (2020). Dynamic ensemble selection based on hesitant fuzzy multiple criteria decision making. *Soft Computing*, 24, 12241–12253.

Elmi, J. & Eftekhari, M. (2021). Multi-Layer Selector (MLS): Dynamic selection based on filtering some competence measures. *Applied Soft Computing*, 104, 107257.

Elmi, J., Eftekhari, M., Mehrpooya, A. & Ravari, M. R. (2023). A novel framework based on the multi-label classification for dynamic selection of classifiers. *International Journal of Machine Learning and Cybernetics*, 1–18.

Fatemipour, F. & Akbarzadeh-T, M.-R. (2014). A genetic fuzzy linguistic rule based approach for dynamic classifier selection in distributed data enviroments. *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)*, pp. 437–442.

Fatemipour, F., Akbarzadeh-T, M.-R. & Ghasempour, R. (2014). A new fuzzy approach for multi-source decision fusion. *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 2238–2243.

Fernández-Delgado, M., Cernadas, E., Barro, S. & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1), 3133–3181.

Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3-4), 121–136.

Gabrys, B. & Bargiela, A. (2000). General fuzzy min-max neural network for clustering and classification. *IEEE Transactions on neural networks*, 11(3), 769–783.

Garcia, D., Kassa, Y. M., Cuevas, A., Cebrian, M., Moro, E., Rahwan, I. & Cuevas, R. (2018). Analyzing gender inequality through large-scale Facebook advertising data. *Proceedings of the National Academy of Sciences*, 115(27), 6958–6963.

Giacinto, G. & Roli, F. (2001). Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, 3.

Hallur, G. G., Prabhu, S. & Aslekar, A. (2021). Entertainment in Era of AI, Big Data & IoT. *Digital Entertainment*, pp. 87–109.

Hou, C., Xia, Y., Xu, Z. & Sun, J. (2016). Learning classifier competence based on graph for dynamic classifier selection. *2016 12th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD)*, pp. 1164–1168.

Huang, Y. S. & Suen, C. Y. (1995). A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE transactions on pattern analysis and machine intelligence*, 17(1), 90–94.

Jahanjoo, A., Tahan, M. N. & Rashti, M. J. (2017). Accurate fall detection using 3-axis accelerometer sensor and MLF algorithm. *2017 3rd International Conference on Pattern Recognition and Image Analysis (IPRIA)*, pp. 90–95.

Jiao, B., Guo, Y., Gong, D. & Chen, Q. (2022). Dynamic ensemble selection for imbalanced data streams with concept drift. *IEEE Transactions on Neural Networks and Learning Systems*, 1-14.

Kalid, S. N., Ng, K.-H., Tong, G.-K. & Khor, K.-C. (2020). A multiple classifiers system for anomaly detection in credit card data with unbalanced and overlapped classes. *IEEE Access*, 8, 28210–28221.

Kenger, Ö. N. & Özceylan, E. (2023). Fuzzy min-max neural networks: a bibliometric and social network analysis. *Neural Computing and Applications*, 1–31.

Kenger, Ö. N. & Ozceylan, E. (2023). A hybrid approach based on mathematical modeling and improved online learning algorithm for data classification. *Expert Systems with Applications*, 218, 119607.

Khuat, T. T. & Gabrys, B. (2020). A comparative study of general fuzzy min-max neural networks for pattern classification problems. *Neurocomputing*, 386, 110–125.

Khuat, T. T. & Gabrys, B. (2021). Accelerated learning algorithms of general fuzzy min-max neural network using a novel hyperbox selection rule. *Information Sciences*, 547, 887–909.

Khuat, T. T., Ruta, D. & Gabrys, B. (2021a). Hyperbox-based machine learning algorithms: a comprehensive survey. *Soft Computing*, 25(2), 1325–1363.

Khuat, T. T., Ruta, D. & Gabrys, B. (2021b). Hyperbox-based machine learning algorithms: a comprehensive survey. *Soft Computing*, 25(2), 1325–1363.

Khuat, T., Chen, F. & Gabrys, B. (2020). An Improved Online Learning Algorithm for General Fuzzy Min-Max Neural Network. *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9.

Knuth, D. E. (1976). Big omicron and big omega and big theta. *ACM Sigact News*, 8(2), 18–24.

Ko, A. H., Sabourin, R. & Britto Jr, A. S. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern recognition*, 41(5), 1718–1731.

Krawczyk, B., Galar, M., Woźniak, M., Bustince, H. & Herrera, F. (2018). Dynamic ensemble selection for multi-class classification with one-class classifiers. *Pattern Recognition*, 83, 34–51.

Kulkarni, U., Doye, D. & Sontakke, T. (2002). General fuzzy hypersphere neural network. *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, 3, 2369–2374.

Kumar, A. & Prasad, P. S. (2020). Scalable fuzzy rough set reduct computation using fuzzy min–max neural network preprocessing. *IEEE Transactions on Fuzzy Systems*, 28(5), 953–964.

Kumar, S. A., Kumar, A., Bajaj, V. & Singh, G. K. (2019). An improved fuzzy min–max neural network for data classification. *IEEE Transactions on Fuzzy Systems*, 28(9), 1910–1924.

Kuncheva, L. I. (2002). A theoretical study on six classifier fusion strategies. *IEEE Transactions on pattern analysis and machine intelligence*, 24(2), 281–286.

Kuncheva, L. I. (2014). *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.

Kurzynski, M. & Krysmann, M. (2014). Fuzzy inference methods applied to the learning competence measure in dynamic classifier selection. *2014 27th SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 180–187.

Li, D., Wen, G., Li, X. & Cai, X. (2019). Graph-based dynamic ensemble pruning for facial expression recognition. *Applied Intelligence*, 49, 3188–3206.

Likas, A. (2001). Reinforcement learning using the stochastic fuzzy min–max neural network. *Neural Processing Letters*, 13, 213–220.

Lin, C., Chen, W., Qiu, C., Wu, Y., Krishnan, S. & Zou, Q. (2014). LibD3C: Ensemble classifiers with a clustering and dynamic selection strategy. *Neurocomputing*, 123, 424–435. doi: 10.1016/j.neucom.2013.08.004.

Liu, H., Diao, X. & Guo, H. (2019). Nonparametric Hyperbox Granular Computing Classification Algorithms. *Information*, 10(2), 76.

Lu, W., Shan, D., Pedrycz, W., Zhang, L., Yang, J. & Liu, X. (2018). Granular fuzzy modeling for multidimensional numeric data: A layered approach Based on Hyperbox. *IEEE Transactions on Fuzzy Systems*, 27(4), 775–789.

Lu, W., Ma, C., Pedrycz, W. & Yang, J. (2021). Design of Granular Model: A Method Driven by Hyper-Box Iteration Granulation. *IEEE Transactions on Cybernetics*, 2899-2913.

Mahajan, G. & Saini, B. (2020). Educational Data Mining: A state-of-the-art survey on tools and techniques used in EDM. *International Journal of Computer Applications & Information Technology*, 12(1), 310–316.

Mahindrakar, M. & Kulkarni, U. (2022). Unbounded Fuzzy Hypersphere Neural Network Classifier. *Journal of The Institution of Engineers (India): Series B*, 103(4), 1335–1343.

Matlock-Hetzel, S. (1997). Basic Concepts in Item and Test Analysis. 22.

Modgil, S., Gupta, S., Sivarajah, U. & Bhushan, B. (2021). Big data-enabled large-scale group decision making for circular economy: An emerging market context. *Technological Forecasting and Social Change*, 166, 120607.

Mohammed, M. F. & Lim, C. P. (2014). An enhanced fuzzy min–max neural network for pattern classification. *IEEE transactions on neural networks and learning systems*, 26(3), 417–429.

Mohammed, M. F. & Lim, C. P. (2017). A new hyperbox selection rule and a pruning strategy for the enhanced fuzzy min–max neural network. *Neural networks*, 86, 69–79.

Nandedkar, A. V. & Biswas, P. K. (2007). A Fuzzy Min-Max Neural Network Classifier With Compensatory Neuron Architecture. *IEEE Transactions on Neural Networks*, 18(1), 42–54. doi: 10.1109/TNN.2006.882811.

Narassiguin, A., Elghazel, H. & Aussem, A. (2017). Dynamic ensemble selection with probabilistic classifier chains. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 169–186.

Nguyen, T. T., Luong, A. V., Van Nguyen, T. M., Ha, T. S., Liew, A. W.-C. & McCall, J. (2019). Simultaneous meta-data and meta-classifier selection in multiple classifier system. *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 39–46. doi: 10.1145/3321707.3321770.

Nguyen, T. T., Luong, A. V., Dang, M. T., Liew, A. W.-C. & McCall, J. (2020). Ensemble Selection based on Classifier Prediction Confidence. *Pattern Recognition*, 100, 107104. doi: 10.1016/j.patcog.2019.107104.

Nozari, H. A., Nazeri, S., Banadaki, H. D. & Castaldi, P. (2018). Model-free fault detection and isolation of a benchmark process control system based on multiple classifiers techniques—A comparative study. *Control Engineering Practice*, 73, 134–148.

Oliveira, D. V., Cavalcanti, G. D. & Sabourin, R. (2017). Online pruning of base classifiers for Dynamic Ensemble Selection. *Pattern Recognition*, 72, 44–58. doi: 10.1016/j.patcog.2017.06.030.

Pereira, M., Britto, A., Oliveira, L. & Sabourin, R. (2018). Dynamic ensemble selection by K-nearest local Oracles with Discrimination Index. *2018 IEEE 30th International conference on tools with artificial intelligence (ICTAI)*, pp. 765–771.

Pinto, F., Soares, C. & Mendes-Moreira, J. (2016a). Chade: Metalearning with classifier chains for dynamic combination of classifiers. *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pp. 410–425.

Pinto, F., Soares, C. & Mendes-Moreira, J. (2016b). CHADE: Metalearning with Classifier Chains for Dynamic Combination of Classifiers. 16.

Porto, A. & Gomide, F. (2022). Evolving hyperbox fuzzy modeling. *Evolving Systems*, 13(3), 423–434.

Pourpanah, F., Lim, C. P., Wang, X., Tan, C. J., Seera, M. & Shi, Y. (2019). A hybrid model of fuzzy min–max and brain storm optimization for feature selection and data classification. *Neurocomputing*, 333, 440–451.

Read, J., Pfahringer, B., Holmes, G. & Frank, E. (2011). Classifier chains for multi-label classification. *Machine learning*, 85(3), 333.

Rey-del Castillo, P. & Cardeñosa, J. (2012). Fuzzy min–max neural networks for categorical data: application to missing data imputation. *Neural Computing and Applications*, 21(6), 1349–1362.

Ruta, D. & Gabrys, B. (2005). Classifier selection for majority voting. *Information fusion*, 6(1), 63–81.

Sabourin, M., Mitiche, A., Thomas, D. & Nagy, G. (1993). Classifier combination for hand-printed digit recognition. *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 163–166.

Salzberg, S. L. (2012). *Learning with nested generalized exemplars*. Springer Science & Business Media.

Seera, M., Randhawa, K. & Lim, C. P. (2018). Improving the fuzzy min–max neural network performance with an ensemble of clustering trees. *Neurocomputing*, 275, 1744–1751.

Simpson, P. K. (1992). Fuzzy Min—MaX Neural Networks—Part 1: Classification. *IEEE Transactions on Neural Networks*, 3(5), 776–786.

Simpson, P. K. & Jahns, G. (1993). Fuzzy min-max neural networks for function approximation. *IEEE International Conference on Neural Networks*, pp. 1967–1972.

Smits, P. C. (2002). Multiple classifier systems for supervised remote sensing image classification based on dynamic classifier selection. *IEEE Transactions on Geoscience and Remote Sensing*, 40(4), 801–813.

Soares, R. G., Santana, A., Canuto, A. M. & de Souto, M. C. P. (2006). Using accuracy and diversity to select classifiers to build ensembles. 1310–1316.

Souza, M. A., Sabourin, R., Cavalcanti, G. D. & Cruz, R. M. (2023). OLP++: An online local classifier for high dimensional data. *Information Fusion*, 90, 120–137.

Stapor, K., Ksieniewicz, P., Garcia, S. & Wozniak, M. (2021). How to design the fair experimental classifier evaluation. *Applied Soft Computing*, 104, 107219.

Trajdos, P. & Kurzynski, M. (2016). A dynamic model of classifier competence based on the local fuzzy confusion matrix and the random reference classifier. *International Journal of Applied Mathematics and Computer Science*, 26(1), 175–189. doi: 10.1515/amcs-2016-0012.

Trajdos, P. & Kurzynski, M. (2018). A correction method of a binary classifier applied to multi-label pairwise models. *International journal of neural systems*, 28(09), 1750062.

Trajdos, P. & Kurzynski, M. (2020). A Correction Method of a Base Classifier Applied to Imbalanced Data Classification. *International Conference on Computational Science*, pp. 88–102.

Van Rijn, J. N., Bischl, B., Torgo, L., Gao, B., Umaashankar, V., Fischer, S., Winter, P., Wiswedel, B., Berthold, M. R. & Vanschoren, J. (2013). OpenML: A collaborative science platform. *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pp. 645–649.

Vijayanand, R., Devaraj, D. & Kannapiran, B. (2018). Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. *Computers & Security*, 77, 304–314.

Waghmare, J. M. & Kulkarni, U. V. (2019). Unbounded Recurrent Fuzzy Min-Max Neural Network for Pattern Classification. *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. doi: 10.1109/IJCNN.2019.8852310.

Walmsley, F. N., Cavalcanti, G. D., Sabourin, R. & Cruz, R. M. (2022). An investigation into the effects of label noise on Dynamic Selection algorithms. *Information Fusion*, 80, 104–120.

Wang, Y., Zhang, J. & Yan, W. (2023). An Enhanced Dynamic Ensemble Selection Classifier for Imbalance Classification With Application to China Corporation Bond Default Prediction. *IEEE Access*, 11, 32082–32094.

Woloszynski, T. & Kurzynski, M. (2009). On a New Measure of Classifier Competence Applied to the Design of Multiclassifier Systems. *Image Analysis and Processing – ICIAP 2009*, pp. 995–1004.

Woloszynski, T. & Kurzynski, M. (2011). A probabilistic model of classifier competence for dynamic ensemble selection. *Pattern Recognition*, 44(10-11), 2656–2668.

Woloszynski, T., Kurzynski, M., Podsiadlo, P. & Stachowiak, G. W. (2012). A measure of competence based on random classification for dynamic ensemble selection. *Information Fusion*, 13(3), 207–213.

Woods, K., Kegelmeyer, W. P. & Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE transactions on pattern analysis and machine intelligence*, 19(4), 405–410.

Xiao, H., Xiao, Z. & Wang, Y. (2016). Ensemble classification based on supervised clustering for credit scoring. *Applied Soft Computing*, 43, 73–86.

Xue, L., Huang, W. & Wang, J. (2020). Ranking-Based Fuzzy Min-Max Classification Neural Network. *Web Information Systems and Applications*, 12432, 352–364. Retrieved from: http://link.springer.com/10.1007/978-3-030-60029-7_33.

Zhang, H., Liu, J., Ma, D. & Wang, Z. (2011). Data-core-based fuzzy min–max neural network for pattern classification. *IEEE transactions on neural networks*, 22(12), 2339–2352.

Zhang, S., Cheng, D., Deng, Z., Zong, M. & Deng, X. (2018). A novel kNN algorithm with data-driven k parameter computation. *Pattern Recognition Letters*, 109, 44–54.

Zyblewski, P., Sabourin, R. & Woźniak, M. (2021). Preprocessed dynamic classifier ensemble selection for highly imbalanced drifted data streams. *Information Fusion*, 66, 138–154.