UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIENCIA DA COMPUTAÇÃO

MARIA LUIZA NASCIMENTO RODRIGUES

An Evaluation of Dynamic Selection robustness in noisy environments for Activity Recognition in Smart Homes

Recife
2020

MARIA LUIZA NASCIMENTO RODRIGUES

An Evaluation of Dynamic Selection robustness in noisy environments for Activity Recognition in Smart Homes

A M.Sc. Dissertation presented to the Center for Informatics of Federal University of Pernambuco in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Field: Computational Intelligence

Advisor: Prof. Dr. George Darmiton da Cunha Cavalcanti

Recife
2020

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

R696e Rodrigues, Maria Luiza Nascimento
An evaluation of dynamic selection robustness in noisy environments for activity recognition in smart homes / Maria Luiza Nascimento Rodrigues. – 2020.
78 f.: il., fig., tab.

Orientador: George Darmiton da Cunha Cavalcanti.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2020.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Sistemas de classificação múltipla. I. Cavalcanti, George Darmiton da Cunha (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2020 - 188

Maria Luiza Nascimento Rodrigues

“An Evaluation of Dynamic Selection robustness in noisy environments for Activity Recognition in Smart Homes”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 13/08/2020.

BANCA EXAMINADORA

Prof. Dr. George Darmiton da Cunha Cavalcanti
Centro de Informática / UFPE
(Orientador)

Prof. Dr. Rafael Menelau Oliveira e Cruz
École de Technologie Supérieure

Prof. Dr. Bruno Almeida Pimentel
Instituto de Computação / UFAL

To God, the Creator.

ACKNOWLEDGEMENTS

To all who contributed to this dissertation, I would like to express some acknowledgments. To God, who with your love and mercy provided me the capacity and strength to do this project. Without him, I would be nothing.

To my advisor, George Darmiton, I would like to thank you. Your guidance and teachings aided me to reach the end of this moment in my life. Despite my difficulties, you believed in me. Thanks a lot.

I thank my family for the unconditional love and support during this time. Even with my fears and anxiety, they have always stood by my side, encouraging me to do my best.

To my friends, my sincere thanks. You were always there when I needed it most.

Thank you.

ABSTRACT

Smart homes can be defined as environments monitored by sensors that capture information executed in it. These sensors are responsible for measure the temperature of a room, the number of times a switch has been turned on, and so on. However, the data obtained in these scenarios may vary during or after the capture process. These variations are defined as noise and affect the interpretation of the data. Given the information obtained from the environment, machine learning techniques can use this knowledge to identify the activities and predict future ones. This area of learning is named Activity Recognition. In recent studies, the Random Forest presented consistent results in Activity Recognition problems in noisy-free environments. To identify which techniques can be used in noisy scenarios, this dissertation evaluated the use of Multiple Classifier Systems in comparison to Random Forest. The proposal is to investigate how these techniques perform on real-world data sets for activity recognition considering six noise levels: 0% to 50%, which refers to a randomly changing in the label activities. Experimental results have shown that the Dynamic Selection techniques are adequate to handle noisy environments presenting stable results as the noise level increases. The performance of OLA and MCB was significantly better than Random Forest even with the 50% noise level.

Keywords: Multiple Classifier Systems. Imbalanced Data. Dynamic Selection. Activity Recognition.

RESUMO

Casas inteligentes podem ser definidas como ambientes monitorados por sensores que capturam as informações nele executadas. Esses sensores são responsáveis por medir a temperatura de uma sala, o número de vezes que um interruptor foi ligado e assim por diante. No entanto, os dados obtidos nesses cenários podem variar durante ou após o processo de captura. Essas variações são definidas como ruído e afetam a interpretação dos dados. Dadas as informações obtidas do ambiente, as técnicas de aprendizado de máquina podem usar esse conhecimento para identificar as atividades executadas e prever as futuras. Essa área de aprendizado é denominada Reconhecimento de Atividade. Recentemente, a Random Forest apresentou resultados consistentes em problemas de reconhecimento de atividade em ambientes sem ruído. Para identificar quais técnicas podem ser usadas em cenários ruidosos para residências inteligentes, esta dissertação avaliou o uso de sistemas de múltiplos classificadores em comparação com o desempenho obtido pela Random Forest. A proposta é investigar o desempenho dessas técnicas em conjuntos de dados do mundo real para reconhecimento de atividades, considerando seis níveis de ruído: 0% a 50%. Resultados experimentais mostraram que as técnicas de seleção dinâmica são adequadas para lidar com ambientes ruidosos, apresentando resultados estáveis à medida que o nível de ruído aumenta. O desempenho do OLA e MCB foi significativamente melhor que o Random Forest, mesmo com o nível de ruído de 50%.

Palavras-chaves: Multiple Classifier Systems. Imbalanced Data. Dynamic Selection. Activity Recognition.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ADLs	Activities of Daily Living
AR	Activity Recognition
DCS	Dynamic Classifier Selection
DES	Dynamic Ensemble Selection
DS	Dynamic Selection
EoC	Ensemble of Classifiers
HAR	Human Activity Recognition
HMM	Hidden Markov Model
IoT	Internet of Things
IR	Imbalance Ratio
KNN	K Nearest Neighbors
LCA	Local Class Accuracy
MCB	Multiple Classifier Behavior
MCS	Multiple Classifier Systems
NBC	Naive Bayes Classification
OLA	Overall Local Accuracy
RoC	Region of Competence
SGH	Self-Generating Hyperplanes
SS	Static Selection
SVM	Support Vector Machines

CONTENTS

1 INTRODUCTION

The advancement of sensor technology and its easy access, coupled with powerful data mining approaches, have contributed to the growth of the term Internet of Things (IoT) becoming quite popular over the years. IoT refers to the use of sensing, coupled with connectivity and intelligent algorithms to make decisions. In the mid-1990s, Mark Weiser was the forerunner of what we know as Ubiquitous Computing (WEISER, 1993): The use of any device, and its services, to establish a connection with other devices and services.

According to Weiser, “ordinary computers such as desktops and laptops are a transitional step towards achieving the real potential of information technology." For Weiser, in the future, machines will be seamlessly integrated into the world in the most diverse shapes and sizes, resulting in the disappearance of more in-depth technologies (WEISER, 1991).

In (MATTERN; FLOERKEMEIER, 2010), Mattern and Floerkemeier state that ordinary daily objects will be intelligent. The objects will be capable of sensing the context it is inserted in, making assumptions, and making decisions to improve the quality of life, its performance, and energy consumption. Thus, it is possible to save time, energy, and money.

IoT can be applied to health-care, transportation, agriculture, education, and many other sectors (AL-FUQAHA et al., 2015). One of those applications is smart homes. Smart Homes are residences that possess a good amount of intelligence in ordinary objects called smart objects. The smart objects can recognize some actions performed by the residents in the sensing environment, as well as communicate with other smart objects. In that way, they can be used to facilitate the resident's life, such as helping to cook, drink, take medicine for living an independent functional life. These self-care actions are known as Activities of Daily Living (ADLs) (KRISHNAN; COOK, 2014). The aging population, the cost of health-care, and the opportunity of improving life's quality of individuals with disabilities were some of the reasons that motivated studies to predict ADLs.

Activity Recognition (AR) aims to identify the resident's context, making activities assumptions through digital devices in Smart Homes. Using the sensors' information, methods of activity recognition monitor the resident's functional status in real-time. Thus, it is possible to find patterns in the resident's routine and allows the automation of some tasks.

The literature reports the use of machine learning for solving the activity recognition challenge (KRISHNAN; COOK, 2014). In recent years, different machine learning algorithms have been used as the key process to find patterns and predict future activities in smart environments.

1.1 MOTIVATION

The data quality during the learning process is essential for the good performance of the algorithms. Unreliable data lead learning to extract incoherent and dystopian information in the introduced scenario. Thus, ensuring that the algorithms are capable of handling changes in the environment guarantees the stability of learning. Unfortunately, data extracted from the real world can change during the process of capturing, processing, or labeling activities, and such changes are defined as noise.

In this manner, the main objective of this project is to evaluate the Dynamic Classifier Selection (DCS) algorithms in activity recognition problems in Smart homes as the noise increases randomly in the training data set. Noise, in our environment, corresponds to a changing on label, activity, defined in the training set. A study related to performance and robustness was conducted for these techniques, and the results found were compared to Random Forest: the most recent approach used to activity recognition problem as detailed in (MINOR; DOPPA; COOK, 2017).

1.2 OVERVIEW OF THE PROPOSAL

The data generated in the smart homes corresponds to a sequence of sensor events along time (COOK et al., 2013). The chunks from the sensor events, according to predefined size, are the activity windows. In each of those windows, the labels could be randomly changed. The changing in the labels is defined as noise.

In our project is proposed an evaluation of ensemble techniques in comparison with the Random Forest's results when the noise level increases. It aims to analyze the impact of the noise insertion during the training phase and describes the pipeline most suitable to this environment, considering the Multiple Classifier System approach. This work focuses only on evaluating the Dynamic Selection (DS) techniques performance since this approach has been shown that surpass the results achieved by Static Selection (JR; SABOURIN; OLIVEIRA, 2014).

Based on this, it was evaluated in two of the MCS stages: Generation and Selection. In the Generation stage, each classifier is trained over sub-regions of feature space to become a specialist. To select the most appropriate technique, the abstract model Oracle was assessed (KUNCHEVA, 2002). In the Selection stage, it was evaluated the performance of DCS techniques as the noise increases in comparison to results obtained by Random Forest.

1.3 RESEARCH METHODOLOGY

In this work, the proposal is to evaluate the performance considering the Smart home noisy environments. A noisy environment can be generated when data capture or storage is corrupted. In CASAS data sets, a specialist defines the label for each activity performed. During the labeling process, some information could be lost or misinterpreted by him. In our scenario, we considered a noisy environment when the corresponding activity was randomly changed in the training set.

The Generation, and Selection phases were experimented, considering the noise insertion on the training phase. That way, the pipeline is compared, and the best is pointed.

The contributions of this work are related below:

- It evaluates robustness to noise in smart environments;
- It discusses the use of pool generation methods in noisy environments;
- It reviews the use of the Dynamic Selection of classifiers in activity recognition;
- It presents a comparison between dynamic selection techniques and one of the best classification model, Random Forest (MINOR; DOPPA; COOK, 2017).

1.4 DISSERTATION STRUCTURE

This dissertation is organized as follows. Chapter 2 presents the background. The MCS main concepts are introduced. In Chapter 3, the Activity Recognition related works are presented. The proposed method is approached in Chapter 4. In this chapter, process steps are presented into two blocks: Training and test phases. Chapter 5 conducts the experiments and evaluates the results according to the methodology previously determined. Lastly, the main points presented in this dissertation are summarized in Chapter 6. The conclusions derived from the experimental results are summarized, and this work's contributions are outlined.

2 BACKGROUND

2.1 INTRODUCTION

Deciding which model will be used in a problem is a tough question. Due to the variety of techniques and their approaches to deal with a problem, select one of those methods depends not only on the problem description but also on the performance achieved during the test/validation phase. According to (WOLPERT; MACREADY, 1997), there is not a single machine learning technique suitable for all kinds of problems. This theorem is called “No Free Lunch Theorem." Therefore, since there is no ideal classifier, an alternative approach is to use a combination of multiple classifiers to perform the classification task. In (WOŹNIAK; GRAÑA; CORCHADO, 2014), the MCS is presented: An approach that aims, given a set of classifiers, to combine several of them based on their competences. The main objective is to outperform the result achieved by a single classifier.

In (ALKOOT; KITTLER, 1999), the performance of MCS techniques have shown to outperform single classifier models. The use of these techniques has shown wide diffusion in solving different problems, such as Music genre classification (ALMEIDA et al., 2012), Handwriting recognition (KO; SABOURIN; BRITTO, 2008), and others.

This chapter proposes to introduce the main concepts of MCS techniques used in our evaluation. In Section 2.2, an overview of the MCS phases: Generation, Selection, and Aggregation - is demonstrated. The Oracle model is also presented in this section. In turn, Section 2.3 presents a brief explanation of the DS approaches adopted since the project is focused on the use of these techniques.

2.2 OVERVIEW

Figure 1 describes the three phases that compound the MCS: Generation, Selection, and Integration. According to (JR; SABOURIN; OLIVEIRA, 2014), the Generation phase is responsible for, given a set Γ, train it to generate a pool of accurate and diverse classifiers, C. In the Selection phase, the most competent classifier, or ensemble, is selected from the pool C. This ensemble of classifiers (EoC) C', where $C' \subset C$ is validated by the validation set ν. The selection phase is optional in the pipeline because some MCS algorithms don't use it. Finally, in the Integration phase, after the classification task performed by the classifiers in C', the outputs are combined to give an outcome.

There are several approaches to all stages briefly described above. This section is focused on present the mains concepts regarding each phase.

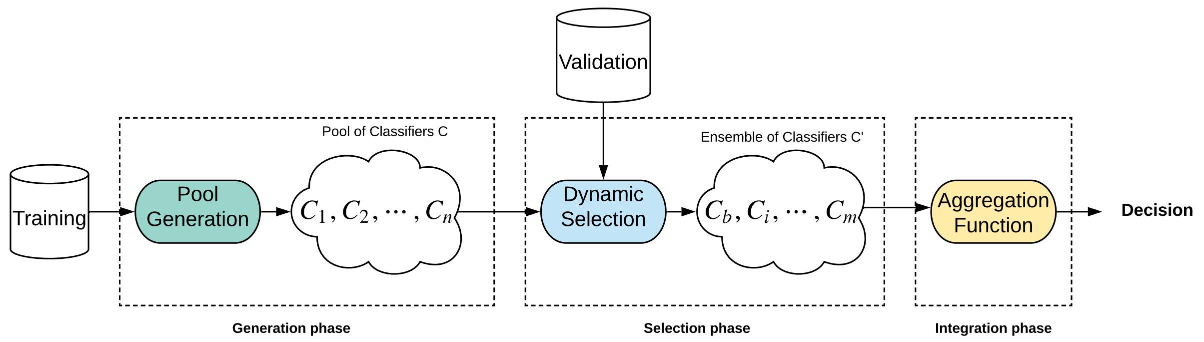


Figure 1 – Stages of a MCS approach. In the first phase, a pool of m classifiers $C = \{c_1, c_2, \cdots, c_m\}$ is generated through a training set Γ. In the selection phase, an EoC, $C' \in C$ is selected by Dynamic Selection techniques using the validation set ν. Lastly, the Integration is responsible for fuses the classifiers' outputs to make a final decision adopted from (CRUZ; SABOURIN; CAVALCANTI, 2018)

2.2.1 Generation

The aim of the Generation phase is to create an accurate and diverse pool of m classifiers, $C = \{c_1, c_2, \cdots, c_m\}$ by the training set Γ. Since the Γ is composed of different base classifiers, the main objective of this phase is to create a pool of classifiers. This pool is nominated diverse when each classifier presents uncorrelated errors between the other ones (SOARES et al., 2006). In other words, a pool that contains classifiers specialized in different portions of feature space is named diverse. Related to the size of the pool, since each kind of problem has its difficulties, there is no formula to define the ensemble size, which means that it is a no-trivial task.

As stated in (ROKACH, 2009), there is four-way to reach diversity in an ensemble:

1. **Manipulating the inducer:** Using different approaches and/or parameters during the training phase;
2. **Manipulation the training samples:** Segmenting the training set in different chunks or a different portion of features for each base classifier;
3. **Manipulating the target attribute representation:** Instead of using a difficult classifier to solve a classification task, combine the multiple different classifiers to induce the representation of target attributes;
4. **Use different classifier models or hybrid ensembles.**: Using different techniques to mix how the classifiers interpret the feature space.

Regarding the methods used to generate the pool of classifiers, three techniques were adopted in this proposed method: Bagging (BREIMAN, 1996), AdaBoost (FREUND; SCHAPIRE, 1997), and Self-Generating Hyperplanes (SOUZA et al., 2019b).

2.2.1.1 Bagging

Bagging is an acronym for *Bootstrap Aggregating*. The purpose of this method is, considering a bootstrap replication of the training set Γ, create an ensemble of classifiers accurate and diverse. Then, the output of classifiers is combined with the Majority Vote (BREIMAN, 1996). Ideally, it is necessary to use different training sets to guarantee diversity in the ensemble. Let $\Gamma = \{t_1, t_2, \cdots, t_n\}$, the training set with n labeled examples. To create different subsets of the training set, a chunk L is created from Γ by sampling with replacement. These samples will be used to train individual classifiers. In other words, each classifier will be trained with a random selection of training set features.

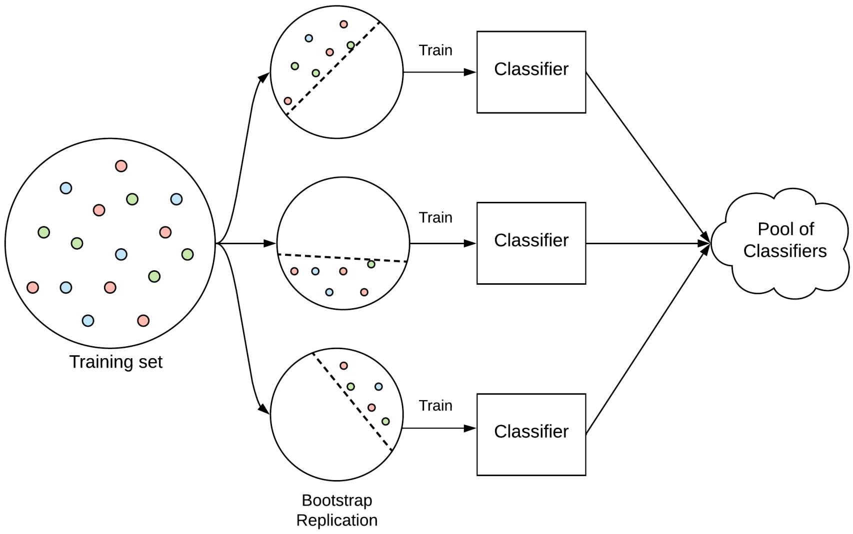


Figure 2 – Bagging method. A training set is randomly distributed by bootstrap replication. Then, each sample is used to train a classifier that will compose a pool of classifiers.

2.2.1.2 AdaBoost

Boosting is defined as the “general problem of producing a very accurate prediction rule by combining rough and moderately inaccurate rules-of-thumb" (FREUND; SCHAPIRE, 1997). In other words, Boosting adds, incrementally, the classifiers in a pool of classifiers.

The Boosting technique, during the training process, considers an error uniform distribution across all learners. The feature classification, named as weight, is used in the next classifier training. The instances misclassified at step $t-1$ defines the weights in the training of the next classifier in step t. Adjusting the weights leads the learners to focus on difficult instances until the propagation error to be null or reaches the threshold defined. This methodology is AdaBoost, *Adaptive Boosting*.

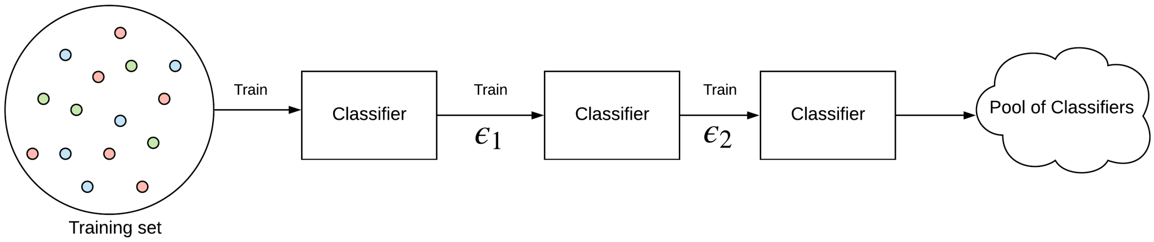


Figure 3 – AdaBoost method. During the training phase, the training errors ϵ are propagated to the next classifier training.

2.2.1.3 Self-Generating Hyperplanes

The Self-Generating Hyperplanes (SGH) method aims to achieve an Oracle accuracy rate of 100%. The Oracle is an important concept related to MCS and considers if at least one classifier can correctly classify a single test sample. The Oracle refers to upper limit for DCS methods or a Dynamic Ensemble Selection (DES) if consider the Majority Voting as combination scheme. So, it is used to compare the results achieved by Dynamic Selection techniques (KUNCHEVA, 2002).

The SGH defines the centroids for each class, and, for each pair of farther centroids, a hyperplane c_m equidistant to them is defined. After, the c_m is evaluated over the training set, and the instances correctly predicted are removed from training data. c_m is added into the ensemble, and a new iteration starts until the training set is empty. Therefore, the SGH ends when all the instances are correctly predicted by at least one classifier C (SOUZA et al., 2019b).

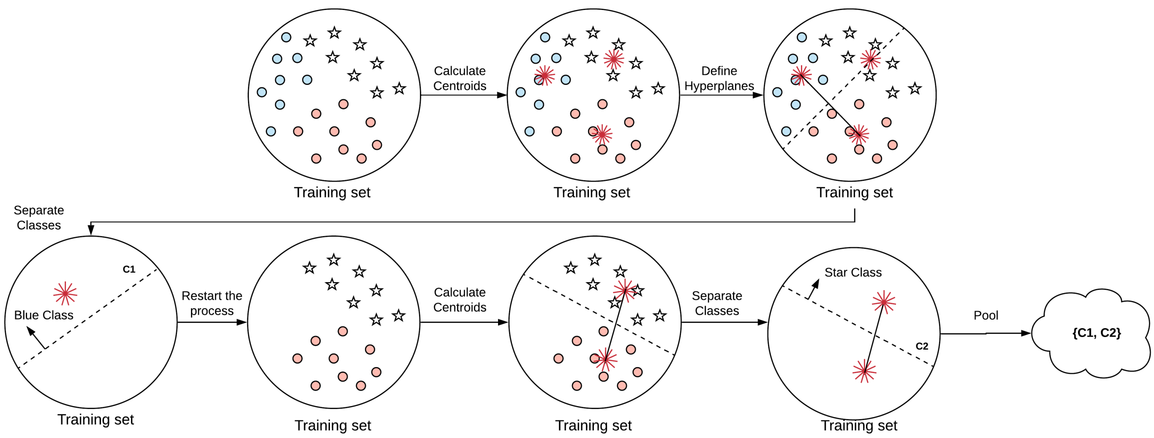


Figure 4 – SGH method. Generation of perceptrons. At the first iteration, the centroids for each pair of classes are calculated. After that, the hyperplane equally distant between the centroids are defined. Lastly, the perceptron able to correctly classify the classes is generated.

2.2.2 Selection

The second phase in the MCS process is Selection. In the Selection phase, the most competent classifier, or ensemble of classifiers, C' from the pool C is selected to perform the classification task. In some generation methods, the classifiers don't need to be selected in the pool. Thus, the Selection phase is not mandatory.

In Selection, two types of approaches are used: Static and Dynamic Selection (JR; SABOURIN; OLIVEIRA, 2014).

For the Static Selection (SS) approach, the selection is performed considering all instances of the training set according to the criterion estimated in the validation dataset. Then, the EoC, or classifier, selected is used to predict throughout all the unknown query x_q samples. In contrast, the Dynamic Selection selects the classifier, or ensemble, for each new query x_q, and, because of this, the DS considers each classifier from the ensemble as an expert in a specific region of the feature space. Figure 5 shows the differences between the two approaches.

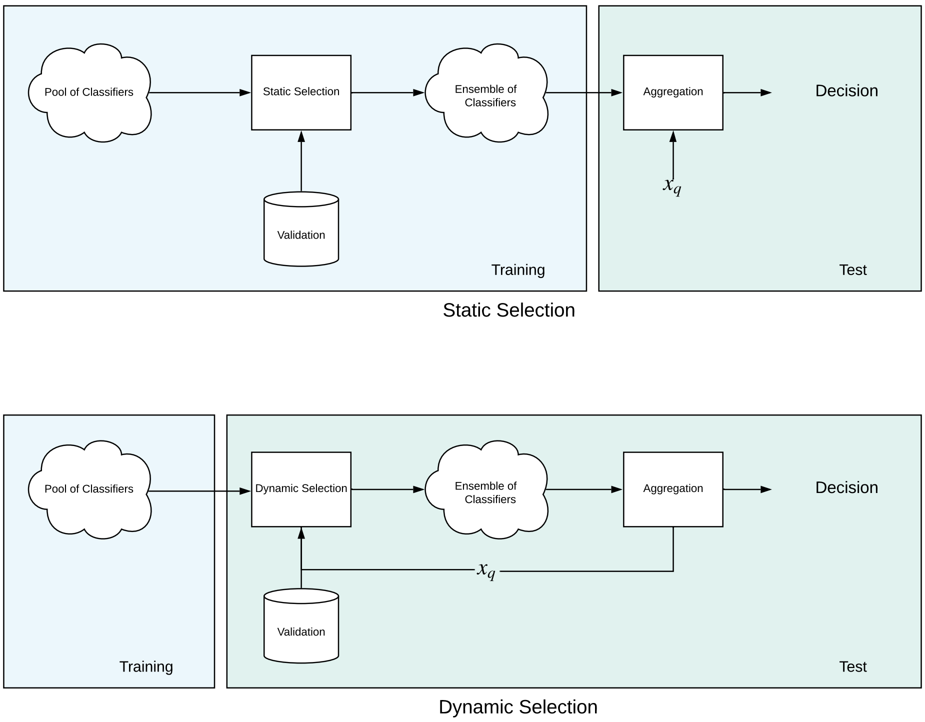


Figure 5 – Static and Dynamic Selection approaches. In the Static approach, all the validation set is considered to selects a classifier, or ensemble, to predict the test samples. Dynamic Selection, in turn, considers each query of the validation set to select a classifier, or ensemble, based on some criterion, such accuracy and/or diversity.

The Dynamic Selection technique selects the classifiers most competent for each new query sample. Because of this, it is acceptable that its results outperform the static selection (KO; SABOURIN; BRITTO, 2008). Thus, in this work, we will focus on DS approach.

2.2.3 Integration

Since the Selection phase can result in more than one classifier, it is necessary to combine its outputs. Fuse the outcome of the selected classifiers considering a combination rule is a responsibility of the integration phase. In this phase, the aggregation of the classifiers' outputs can be non-trainable, trainable, or based on dynamic weighting (CRUZ; SABOURIN; CAVALCANTI, 2018).

Regarding the non-trainable combiners, the techniques like the Majority, Sum, and Product voting have the fixed combination rule (KITTLER et al., 1998). The commonly used approach is the Majority Vote. In this approach, the outputs are combined based on the frequency of the labels in the classifier predictions. The outcome of the system is the most voted class.

2.3 DYNAMIC SELECTION TECHNIQUES

The Dynamic Selection aims to select the classifier, or ensemble, most competent to the unknown test sample. Regarding the DS approach, the following steps are required to classify a new query x_q (CRUZ; SABOURIN; CAVALCANTI, 2018):

1. Define a Region of Competence: A local region around the query x_q used to calculate the classifier competence level; This region is composed of the neighbors' instances of the test sample.

2. Determine the criteria used to calculate the competence level of each classifier: Accuracy, probability, etc.

3. Determine if the selection will be DCS, which chooses only one classifier, or DES, which selects an ensemble.

Firstly, to select the classifier able to predict the query sample, x_q, a local region is obtained. This region around the query sample is called the Region of Competence, RoC. The validation set is an input to K Nearest Neighbors (KNN), a method that selects the k samples nearby the x_q to delimit the region. Thus, the competence of the given classifier, according to the criteria previously specified, is calculated in the Region of Competence (RoC). At the end, based on the competences scores, the classifier, or ensemble, is selected to perform through the test samples.

According to (CRUZ; SABOURIN; CAVALCANTI, 2016; LIMA; SERGIO; LUDERMIR, 2014; GIACINTO; DIDACI, 2004), the definition of the Region of Competence has a huge impact on the DS performance, and its defined in four approaches (CRUZ; SABOURIN; CAVALCANTI, 2018):

- **Clustering:** In this approach, the validation dataset is clustered to define the Region of Competence. The competence of each classifier is estimated to all clusters, and the distance between the unknown sample and the cluster's centroid is calculated, during the test phase, to evaluate the competence of each classifier. Lastly, the classifier competence is measured to the test samples belonging to the nearest clusters. This approach is faster than KNN because it performs only in the centroids instead of each sample.

- **K-Nearest Neighbors:** Considers the KNN method to define the Region of Competence in the validation set. Since the number of neighbors, K, is established, the competence level is calculated using the criteria set to samples whose is a part of the RoC. Since this approach considers the neighbors, it is more precise to define the local region even has the computational cost higher than the Clustering method.

- **Potential Function Model:** Despite the other approaches, the whole validation set is used to calculate the competence level for each classifier. In this approach, each data point in the validation set has the weight measured by the Euclidian distance between the unknown sample using the Potential Function Model, commonly Gaussian potential function. As an advantage, this methodology discards the necessity to define a RoC, although the computational cost involved.

- **Decision Space:** Inspired by (HUANG; SUEN, 1995), this methodology uses classifier's predictions as information to define the "Decision Space." In this technique, the test and training samples are transformed into output profiles, where corresponds to the decision profit by the classifiers. Then, the RoC is defined by the instances whose output profile is closer to the query sample x_q.

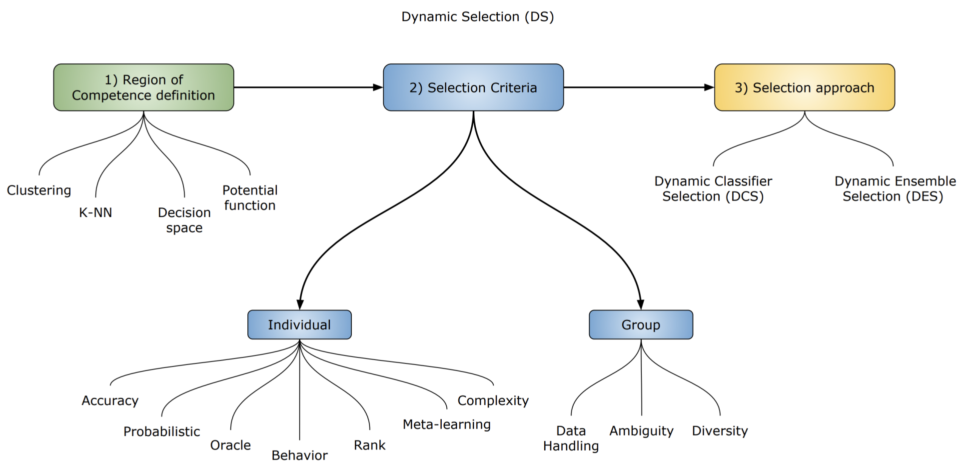


Figure 6 – Taxonomy of Dynamic Selection based on the proposed taxonomy by (CRUZ; SABOURIN; CAVALCANTI, 2018).

The criteria used to evaluate the performance of a classifier can vary according to different approaches, such Accuracy (WOODS; KEGELMEYER; BOWYER, 1997), Ranking

(SABOURIN et al., 1993), Probabilistic (GIACINTO; ROLI, 1999), Behavior (GIACINTO; ROLI, 2001) and so on. In the next section, some of these techniques were detailed.

2.3.1 Dynamic Classifier Selection

The proposed method, posterior described, focuses the evaluation of the performance of the DS techniques and Random Forest in noisy environments.

The DCS techniques used in this work are described below. Its definition of the Region of Competence is based on the KNN approach.

2.3.1.1 Overall Local Accuracy

According to (WOODS; KEGELMEYER; BOWYER, 1997), the OLA considers the number of instances in the Region of Competence correctly predicted as the competence level, σ_i, as stated in Equation 2.1. The base classifier, c_i, which has the highest percentage of the hits is selected to predict the sample x_q.

$$\sigma_i = \frac{1}{K} \sum_{k=1}^{K} P(w_l | x_k \in w_l, c_i), \tag{2.1}$$

where w_l refers to the output class assigned by the classifier c_i to the neighbors x_k.

2.3.1.2 Local Classifier Accuracy

Despite the resemblance with OLA, the LCA (WOODS; KEGELMEYER; BOWYER, 1997) calculates the competence level, σ_i, based on the labels assigned by the classifiers to the instances of validation set. This approach considers the class attributed to the test sample, x_q, and estimate the percentage of the neighbors in the RoC were classified by c_i as belonging to the same label, w_l, and were correctly classified. As the OLA, the classifier with the highest competence level is selected.

$$\sigma_i = \frac{\sum_{x_k \in w_l} P(w_l | w_l, c_i)}{\sum_{k=1}^{K} P(w_l | w_l, c_i)} \tag{2.2}$$

2.3.1.3 Modified Classifier Ranking

In the Modified Classifier Ranking method (SABOURIN et al., 1993), the competence level, σ_i, is estimated based on the instances correctly labeled in RoC successively. This percentage determines the rank of the classifier, and the one who has the highest competence level, the number of consecutive samples correctly predicted, will be selected to perform in the query x_q.

2.3.1.4 Multiple Classifier Behavior

The Multiple Classifier Behavior (MCB) technique (GIACINTO; ROLI, 2001) is based on (HUANG; SUEN, 1995), and the local accuracy method. In this approach, the competence level, σ_i, is calculated considering the output profile similarity, α_i, between the instances in the Region of Competence and the instance x_q. As established, the instances which have the similarity below the predefined threshold will be removed from the RoC. Then, the competence level of classifiers is computed based on the percentage of examples correctly predicted. The classifier with the highest competence, c_i, will be selected to predict x_q.

$$\alpha_i = \frac{1}{M} \sum_{i=1}^{M} T(x_q, x_j) \tag{2.3}$$

$$T(x_i, x_j) = \begin{cases} 1, \text{if } c(x_i) = c(x_j) \\ 0, otherwise \end{cases} \tag{2.4}$$

, where the x_q is the sample to be predicted, and the $x_j \in$ RoC.

2.3.2 Dynamic Ensemble Selection

The DES techniques evaluated in our proposed method are described in the next topics. Similarly to DCS methods, the Region of Competence is based on the KNN approach.

2.3.2.1 KNORA-Elimination

Let the Region of Competence ω_j. The classifiers who correctly predicted all the samples in the RoC, are selected to compose the Ensemble of Classifiers, C'. If none of the base classifiers were selected, then the Region of Competence is reduced, and the selection phase is restarted.

2.3.2.2 KNORA-Union

In contrast to KNORA-Elimination, the KNORA-Union selects the classifiers which accurately labeled, at least, one of the samples in the Region of Competence ω_j. In this approach, the number of votes that a classifier has is correspondent to the number of samples correctly predicted in the RoC.

2.3.2.3 K-Nearest Output Profiles (KNOP)

Similarly to KNORA-Union, KNOP selects the classifiers which correctly predicted, at least, one of the profiles in the set ϕ_j. Contrasting KNORA-Union that works on feature space, KNOP chooses based on decision space. The output profiles are calculated for each sample, and the similarity between them and the validation space is measured. Then, the results are stored in the set ϕ_j.

2.3.2.4 DES-KNN

Initially, the Region of Competence ω_j is computed. Based on the samples in ω_j, the classifiers are decreasing ordered by accuracy and increasing order by diversity. Here, the Double Fault is the measure used to calculate the diversity. Then, the N most accurate and J most diverse classifiers are selected to the EoC, where J and N are previously defined and $J \leq N$.

2.3.2.5 DES-P

Considering the Region of Competence ω_j, the base classifiers' local accuracy is calculated for each sample in ω_j. Thus, the classifier competence is measured as the difference between its accuracy and the performance of the random classifier, that is, the classification model that randomly chooses a class with equal probabilities.

The performance of the Random Classifier is calculated as follows:

$$RC = \frac{1}{L}, \tag{2.5}$$

where L means the number of classes.

So, the competence level can be measured according to the following equation:

$$\alpha_{i,j} = P(c_i|\omega_j) - \frac{1}{L}, \tag{2.6}$$

2.3.2.6 METADES

In this algorithm, the selection phase can be considered as a meta-problem, which uses the behavior of each classifier as the criteria to select it. Instead of using the features itself, it is calculated the meta-features. Then, during the meta-training phase, the meta-classifier is trained using the meta-features. Thus, the meta-classifier predicts if a classifier is competent enough to classify a given input.

3 RELATED WORKS

In this section, Activity Recognition related works are presented. The architecture proposed for capturing the signals and the system-design is detailed in Section 3.1. After, in Section 3.2, the use of machine learning techniques to predict the activities are described. Finally, the recent discoveries in the Activity Recognition problem are depicted.

3.1 CASAS: AN ARCHITECTURE OVERVIEW

The Activities of Daily Living Recognition is one of the most interesting aspects of Human Recognition problems. This process is complex and provides information to assist the aging population as well as mechanisms to help them to have an independent life. Various researchers have proposed approaches to solve it.

In 2013, (COOK et al., 2013) proposed an architecture to smart homes denominated CASAS. The CASAS project changed the way data are captured and stored. Their architecture utilizes a ZigBee wireless mesh that communicates directly with hardware components and captures information from different regions of the home. The middleware provides services, such as adding timestamps to events, assigning universally unique identifiers (UUIDs), and persisting sensor state. The residences are treated as intelligent agents to perceive the state of the residents and grab the surrounding sensor information. Figure 7 presents a sensor layout for CASAS datasets.

Figure 7 – Sensor layout for seven CASAS smart homes testbeds: The sensor distribution depends on the number of residents and the type of activities performed.

Initially, the activity recognition was proposed using simple approaches, such as Hidden Markov Model (HMM), and Naive Bayes Classification (NBC). Further, in (KRISHNAN; COOK, 2014), the Support Vector Machine (SVM) method was applied for real-time activity recognition. The articles showed in Table 2 focus on four aspects: i) Ubiquitous sensors, which indicates the use of sensors along with the environment; ii) Noise, which indicates if a dataset has divergence from data captured; iii) AR, which indicates if it is related to Activity Recognition; iv) Ensemble, which indicates the combination of classifiers to predict the activities. The Summary column in Table 2 presents an overview of the related project. The articles presented will be discussed in the sections below.

Method	Ubiquitous sensors	Noise	AR	Ensemble	Summary
Mozer (MOZER, 1998)	✓				Adaption of environment by Neural Networks
Essa (ESSA, 2000)	✓				Ubiquitous sensors distributed
Hong et al. (HONG et al., 2009)	✓		✓		Combines sensor's information to activity consensus
Intille et al. (INTILLE et al., 2010)	✓				Observational apartment by sensoring systems
Chen et al. (CHEN; NUGENT; WANG, 2012)	✓		✓		Proposed ontology to activity recognition based on home characteristics
Krishnan and Cook (KRISHNAN; COOK, 2014)	✓		✓		Support Vector Machines (SVM) method for predict activities on sensor-window
Lu lu et al. (LU; QINGLING; YI-JU, 2017)			✓		Knowledge-driven approach to real-time
Aminikhanghahi and Cook (AMINIKHANGHAHI; COOK, 2019)	✓		✓	✓	Activity-segmentation detection
Irvine et al. (IRVINE et al., 2019)	✓		✓	✓	Homogeneous ensemble neural network approach to Human Activity Recognition
Myagmar et al. (Myagmar; Li; Kimura, 2020)	✓		✓	✓	Heterogeneous transfer learning approach to Activity Recognition
Andrea Sanabria and Juan Ye (SANABRIA; YE, 2020)	✓		✓		Unsupervised Domain adaptation technique for Activity Recognition
Proposed Method	✓	✓	✓	✓	Dynamic Selection of Perceptrons

Table 2 – Related works.

3.2 ACTIVITY RECOGNITION APPROACHES

Mozer (MOZER, 1998) built ACHE, Adaptive Control of Home Environment, which is a domestic laboratory equipped with around 75 sensors to provide information related to environment status. The approach uses Neural networks to adapt the equipment according to the activities that will be performed. These predictions could assist in daily tasks, such as to define when the heater will be turned on, for instance. Despite the approach modeling the adaptive control in regular activities, in a real-life scenario, it is well-known that the activities are irregularly performed. Thus, this could be a roadblock to predict the next actions.

Essa (ESSA, 2000) through a residential laboratory, The Aware Home, in the Activity Monitor project, evaluates the acceptance of aging people to health-care monitoring technologies. Eight attendants wore health technologies, such as cardiac monitor, thermometer and others, for two weeks in the research residences, where five of them indicated to would like to continue using it. The results suggested that the efforts may focus on spreading the benefits to monitor health in older people and the usage of its. As detailed, in this proposal, an important aspect of building a smart environment is to explore easily accessible and pervasive computing services. However, the sensors used here are narrow to audio and video capture, which means that interpretation could represent a limitation.

Hong et al. (HONG et al., 2009) have proposed to use information handling techniques, mainly the Dempster–Shafer theory of evidence and the Equally Weighted Sum operator to achieve an activity consensus on ADLs prediction. The approach proposes an Evidential Neural Network to represent the hierarchy of inferring context-aware activities based on sensors' data since the information extracted from them could be unreliable due to corruption, interference, or faults. Although this behavior is more common in real databases, in this project, we decided on the use of simulated noisy data. Thus, the results achieved might not reflect the behavior in a real smart home environment.

Lastly, Intille et al. (INTILLE et al., 2010) proposed an observed apartment where the condo rooms were equipped with many sensors to develop applications that aid people in controlling the environment, save resources, and monitoring activity. The PlaceLab was designed with sensors integrated into the architectural aesthetic. The attendants also used wearable sensors to integrate with the environment due to its proposal architecture could be subject to biases and to be poor at capturing chains of causality.

3.3 RECENT DISCOVERIES

In 2012, Liming Chen et al. (CHEN; NUGENT; WANG, 2012) have proposed a knowledge-driven approach to real-time activities. A generic system architecture was proposed for the approach based on analyzes of smart home characteristics. Then, an ontology-based recognition process was presented. The approach to activity recognition is built upon Description Logic theories and reasoning mechanisms. The system has been tested and validated in both real-world and simulated activity scenarios.

In 2014, Narayanan Krishnan and Diane J. Cook. in (KRISHNAN; COOK, 2014) defined the environments as smart agents, where the residents status and the rooms of the house are perceived by sensors, and modified by the controllers' usage to improve the comfort, security and/or to help in activities performed by the residents. In this project, combining mutual information based weighting of sensor events and adding past contextual information into the feature presented the best results to predict the activities.

In 2017, Lu lu et al. (LU; QING-LING; YI-JU, 2017) proposed an approach to assisting living and health-care using wearable technologies. The method extracts the values captured by sensors and, using the Beta Process Hidden Markov Model, defines latent features to train the SVM. According to the results, a suggestion to obtain better results is, previously, extract the latent features, and then train the intelligent algorithms.

In 2019, Samaneh Aminikhanghahi and Diane J. Cook (AMINIKHANGHAHI; COOK, 2019) adopted activity recognition detection based on the segmentation model to improve the robustness of the machine learning techniques. According to this paper, segment behavior-drift sensor data in real-time improves the accuracy of its by providing information about activity transitions and, thus, insights on activity start/end times and duration. Therefore, the approach proposed by Aminikhanghahi considered the imbalanced data set arising from the time distribution in time.

Also in 2019, Irvine et al. (IRVINE et al., 2019) proposed a homogeneous ensemble neural network approach to Human Activity Recognition (HAR) in smart environments. The activities monitored in this article are related to the ones performed by the inhabitants. In other words, it's an action taken by the resident, such as 'Prepare dinner", 'Wash Dishes" and so on. The experiments demonstrated that as the longer activity is, there are fewer conflicts between the base models leading to the increase in the performance before the conflict resolution, where occurs when more than one model chosen the main class output. The Neural Network Ensemble proposed surpasses the results of two non-parametric benchmarks: KNN and SVM. However, the approach proposed did not consider a feature selection to determine the optimal subspace features.

In 2020, Myagmar et al. (Myagmar; Li; Kimura, 2020) proposed a heterogeneous transfer learning algorithm called Heterogeneous Daily Living Activity Learning (HDLAL) by applying Maximum Mean Discrepancy, and Principal Component Analysis in smart environments to predict Activity Recognition. The approach is used to derive the domain feature space from other domains. Then, Random Forest, an ensemble classifier, is used to train this new feature space.

Lastly, in 2020, Andrea Sanabria and Juan Ye (SANABRIA; YE, 2020) proposed an Unsupervised Domain adaptation technique for Activity Recognition named UDAR. The main purpose of this article is, considering that there is insufficient labeled data related to activities performed in smart environments, an approach capable of recognizing activities in an unknown dataset through unsupervised techniques using transferring learning. The results have shown that their approach is consistent and outperforms the state-of-art domain techniques results found in the literature. However, this approach is limited to the sensors' position. To achieve better results, the setting of sensors should be similar, and semantically comprehensible to the source during the mapping. Otherwise, the sensors need to be remapped.

4 PROPOSED METHOD

4.1 INTRODUCTION

In the related works, we explored the usage of machine learning techniques to Activity Recognition in smart environments. However, in most cases, the context is noise-free, which means that the data captured is as expected. Hence, to evaluate the performance of DS techniques to activity recognition in a noisy environment, we proposed the following experiment: the idea is to use this data modeling approach in a noisy scenario, where the window label changes randomly as the noise level up. Figure 8 shows an overview of the proposed activity recognition process that is divided into two phases: training and test.

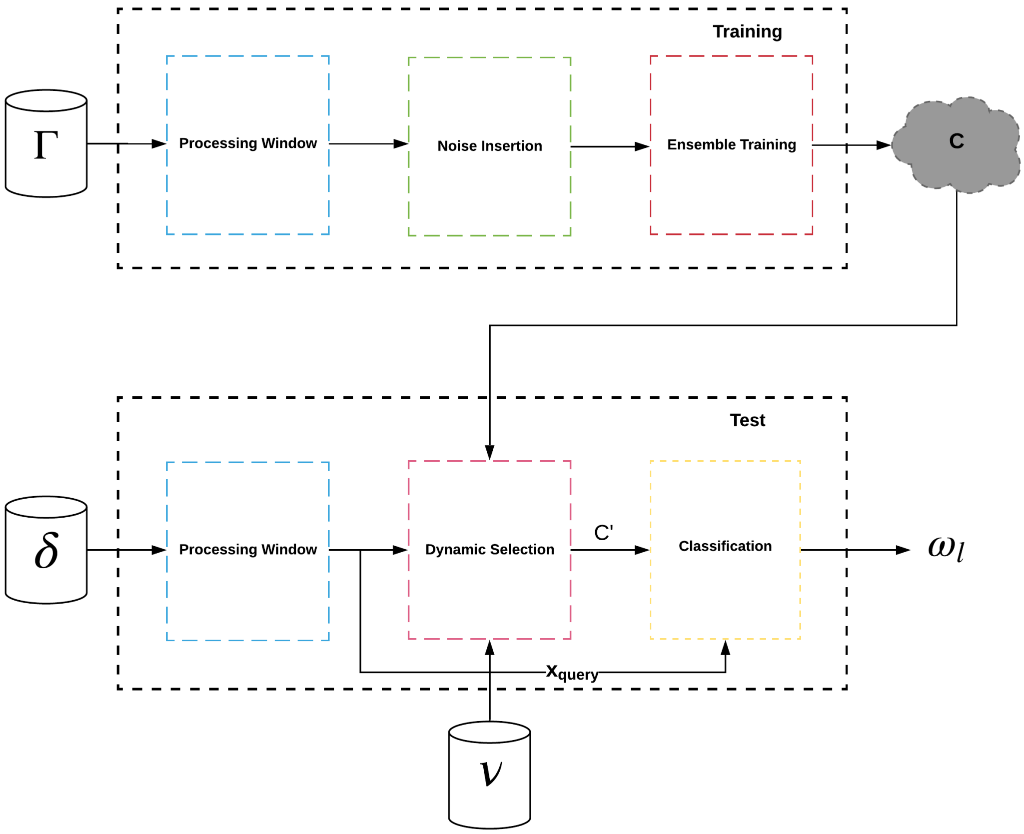


Figure 8 – The proposed method overview. Γ corresponds to the set of sequence of sensor events, C is the Pool of Classifiers generated, δ is the test data set, x_{query} is the query sample, C' is a subset of the Pool of Classifiers C, and the ω_l is the outcome of the sample x_{query}. In the first phase, the C is obtained from the training phase. Then, in the test phase, the C' is selected according to the approach evaluated to labeling as ω_l the x_{query} and validated by validation set ν.

In order to simplify the understanding of the following sections, the notation used in our approach is described below:

C: the Pool of Classifiers.

C': the most competent classifier, or ensemble. $C' \subset C$.

D_{tr}: the training set.

δ: the test dataset.

Γ: the training dataset.

q: the number of windows in W.

S: the set of sensors events.

t: the number of vectors in X.

X: the set of feature vectors.

x_j: the jth feature vector in X.

x_{query}: the query example.

ω: the class label for the query.

W: the set of windows.

w_i: the ith instance of window in W.

Section 4.2 presents the training phase that is responsible for the data transformation, and for the training of the Pool of Classifiers. Section 4.3 details the Dynamic Selection step responsible for selecting the most competent classifier, or ensemble, and labeling the instance.

4.2 TRAINING PHASE

Figure 9 shows the training phase that is divided into three main steps:

1. Processing Window: The raw sensor data is transformed into labeled windows, in which the labels correspond to the activities.

2. Noise Insertion: Random noise is added to the data.

3. Ensemble Training: A pool of classifiers C containing m trained classifiers is generated.

In this phase, the Processing Window step defined in (KRISHNAN; COOK, 2014) is responsible for, given a sequence of sensor events Γ, dividing it into activity windows and then, extract features, such as the sensor's activation frequency, for instance. Afterward, in the Noise Insertion step, the labels of the generated windows are randomly changed. Finally, the Ensemble Training trains the pool of classifiers C. In the next sections, these steps are detailed.

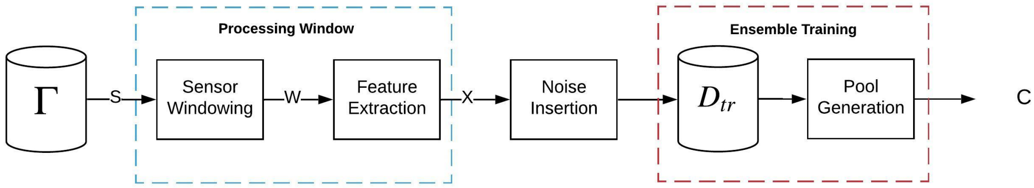


Figure 9 – Training phase. Γ corresponds to the raw data and S to set of sensors extracted from Γ. W refers to the Activity windows, and X to the Bag of Sensors. D_{tr} is the training set used during the Ensemble Training to generate the Pool of Classifiers, C.

4.2.1 Processing Window

According to (CHEN; DAS; COOK, 2010), evaluating the previous sensor events is an approach to comprehending the sensor context in an environment. In the Processing Window, the sequence of sensor events S is divided into small blocks of data called windows, denoted by W. This process consists of two steps: Sensor Windowing and Feature Extraction. Figure 10 shows the complete process.

According to (COOK; KRISHNAN, 2015), the Sensor Windowing step is defined as follows: A process that divides the sensor data stream into windows, W. The number of events that compose a window is called window size. In Figure 10, two windows, w_1 and w_2, are generated in the Sensor Windowing phase considering the window size as 4 sensors events.

Defining the best window size is a difficult task because, if we assign a small window, the window might contain no labeled events. Otherwise, a large window may include multiple distinct activities (KRISHNAN; COOK, 2014).

Once the set of windows, $W = \{w_1, w_2, \cdots, w_q\}$ was defined in the Sensor Windowing step, the next step is to extract the features that correspond to its content. The q windows are used to extract the q feature vectors $X = \{x_1, x_2, ..., x_q\}$ in this phase. In Figure 10, each vector is composed of 8 features extracted from windows.

Each feature vector x_i is composed of two types of elements: time and sensors. The time-based features are composed of three arguments extracted regarding the time of the

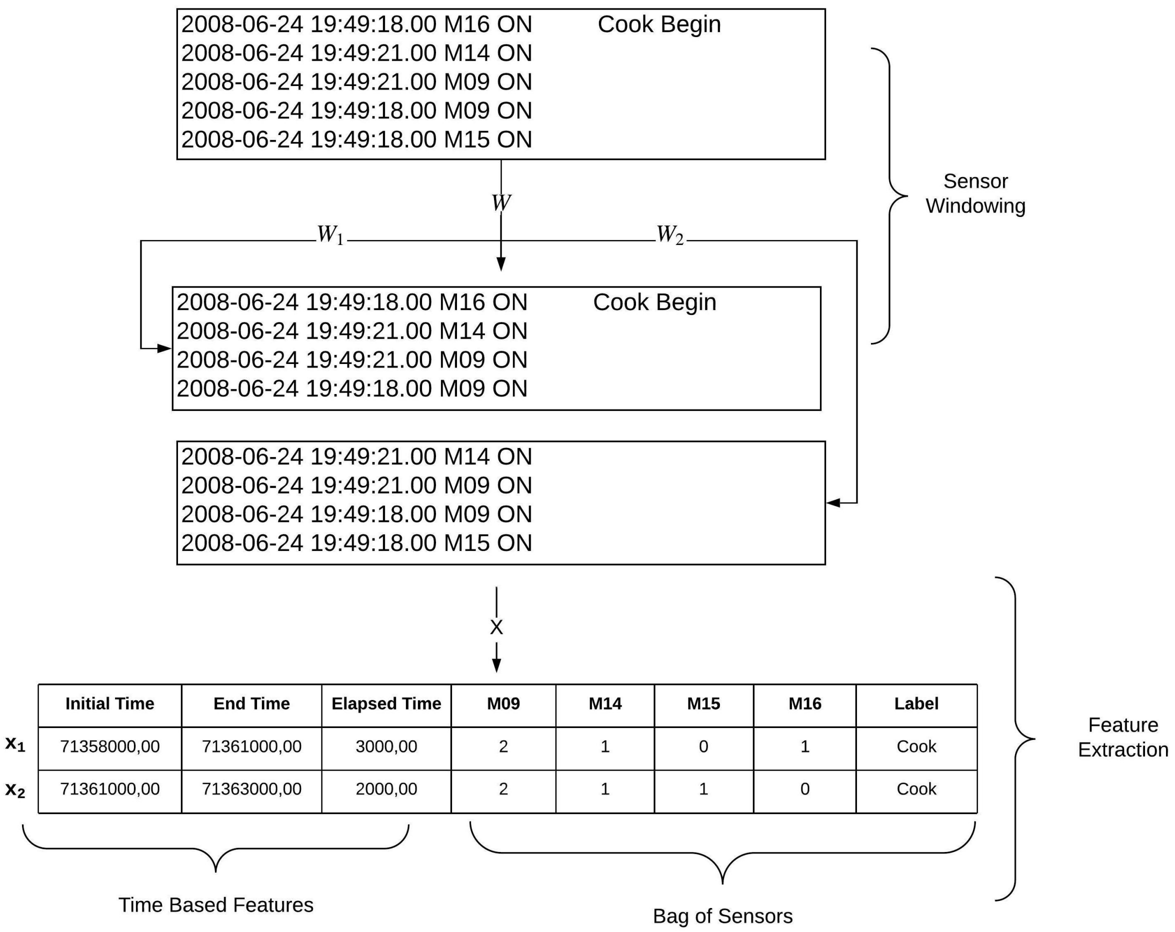


Figure 10 – Processing Window. An example of a transformation of sensors events into a Bag of Sensors.

sensor events:

- Time of the first sensor event in the window.
- Time of the last sensor event in the window.
- The difference between them (duration of the window).

The Bag of sensors is analogous to the Bag of words, a technique commonly used in Text mining (SEBASTIANI, 2002). In Bag of words, a text can be represented as a set of words and their frequency that appears in the document. Similarly, in Bag of Sensors, for each sensor in the dataset, the frequency of the sensor's activation defines the window, in which the value of sensor status can be categorical (ON/OFF) or numeric. Therefore, the dimensionality of the feature vectors depends on the number of sensors in the dataset. For instance, for a dataset with 25 sensors, the total number of features is 28 (25 sensors plus 3 time-based features).

Lastly, each window needs to be labeled. The activity label is defined based on the last sensor event within the window. If the last sensor event of the window has the class label defined, this will be the window label. Otherwise, the window label corresponds to the “begin-end” block the window is in. For example, in Figure 10, the window label is “Cook". In cases where the class is not explicitly determined, the window is labeled as “Other”. In well-known real-world data sets, most of the windows do not correspond to the known activities. For that reason, it is essential to incorporate the class “Other”. (KRISHNAN; COOK, 2014).

4.2.2 Noise Insertion

In real-world environments, the data available is not always flawless. In some cases, the data is not reliable due to the inconsistency during its capture. For instance, when one of the sensors has malfunction producing unexpected results or when the specialists mislabeling the classes because they are not familiar with the pattern presented. Formally, a data in which the information obtained has divergence from the expected is named Noisy data (Wu, 2007). Handling noisy data is still a challenge to machine learning approaches since it may affect the results of the models.

Noise insertion is used to verify how the classifier behaves in noisy environments. The generation of noisy data could be performed in different ways. (ZHU; WU, 2004) describes two approaches widely adopted: Attribute and Class label noise insertion. In Attribute noise insertion, the errors are introduced in the information captured by sensors, for example, or by missing information. On the other hand, the Class label noise insertion can be performed changing the label defined to a data set (NETTLETON; ORRIOLS-PUIG; FORNELLS, 2010).

In this work, an amount of the training examples is randomly selected to have their labels changed to another class to simulate a noisy environment. Six noise levels are evaluated in our proposed method, from 0% to 50%. Noise-free data, 0%, means that the data was not changed up, while 50% corresponds to, at maximum, half of the labels randomly mutated.

As the insertion of noise increases, new labels are randomly assigned to the classes. Each noise level corresponds to a new dataset to be trained generated from the noise-free environment, 0%. For instance, 10% refers to a dataset created from 0%. Due to the randomness, new classes may correspond to the current ones, which does not indicate a change. It is possible to observe this behavior when changing from 10 % to 20 % in Figure 11, and the class label does not change. Therefore, the 50% can be related to, at maximum, the change of half of the labels.

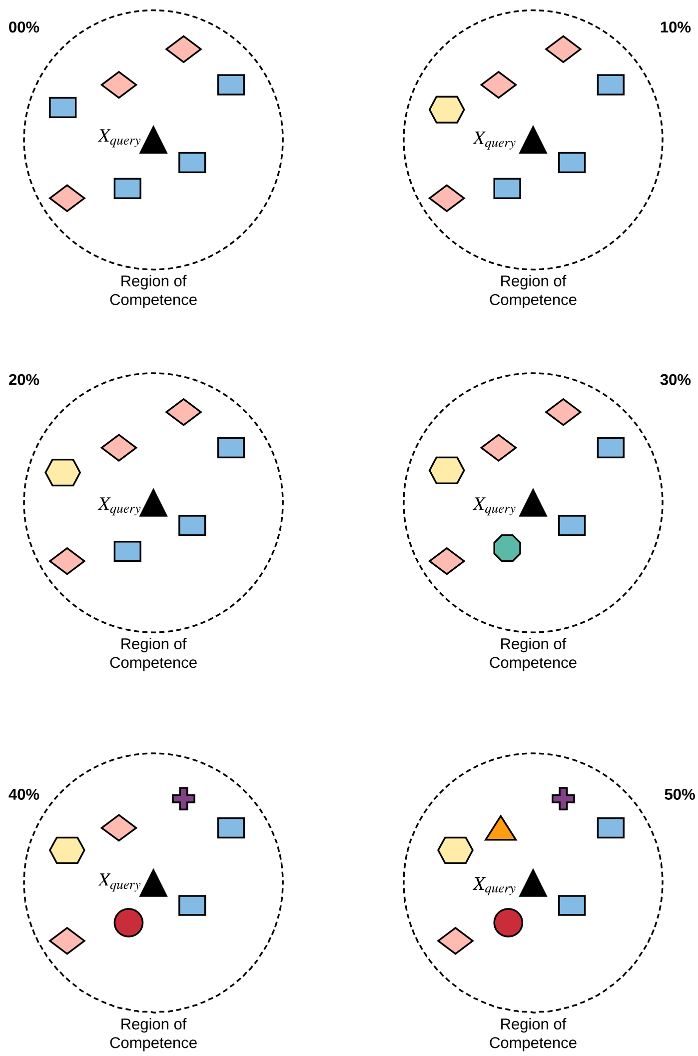


Figure 11 – A snapshot of the noise insertion in the data set for six noise levels. 0% refers to noise-free environments, and the 50% to a scenario where the, at maximum, half of the labels were changed.

4.2.3 Ensemble Training

After the Noise Insertion step, the data is used to train the classifiers. In the Ensemble Training step, the proposed method has several examples in D_{tr}, the training set. To train different classifiers, some methods of pool generation are evaluated to obtain the pool of m classifiers $C = \{c_1, c_2, ..., c_m\}$. Thus, in our project, three ensemble generation techniques are evaluated: Bagging (BREIMAN, 1996), AdaBoost (FREUND; SCHAPIRE, 1997), and Self-Generating Hyperplanes (SOUZA et al., 2019a).

In the Bagging approach, the pool of classifiers is generated using bootstraps. Bootstraps are replicated random samples of the dataset, where a tuple (class, instance) may appear repeated or not at all in one of those. The usage of this approach is recommended in cases where the predictors have high variance. These predictors are named unstable, which means when it has small changes in the input, there are large changes in the predictor. However, Bagging does not improve stable methods (BREIMAN, 1996).

In the Boosting approach, (SCHAPIRE, 1990) demonstrates the possibility of generating a strong learner from weak ones. One of the approaches used to assert the proposition is AdaBoost. In a nutshell, AdaBoost uses, in a pipeline of classifiers, the learning error of the previous training set as weight to generate a new training set that will be used to train the current classifier. In this manner, this algorithm can lead to overfitting. (RATSCH; ONODA; MüLLER, 2001), (QUINLAN, 1996), and (GROVE; SCHUURMANS, 1998) mention that AdaBoost has a lower performance than individual classifiers in the presence of noise.

The Self-Generating Hyperplanes is an alternative method for generating the pool of classifiers. Instead of randomly selecting the instances to train a subset of classifiers, the SGH sets a classifier able to equally divide the feature-space. The premise of SGH is to achieve an Oracle accuracy rate of 100% on the training set. Therefore, the number of classifiers in the pool is deterministic, which means that for input data, the number of classifiers selected to compose the pool is the same in all executions. Regarding selecting a hyperplane equally distant, it may not work well in boundary cases, where the instances are outliers, for instance.

4.3 TEST PHASE

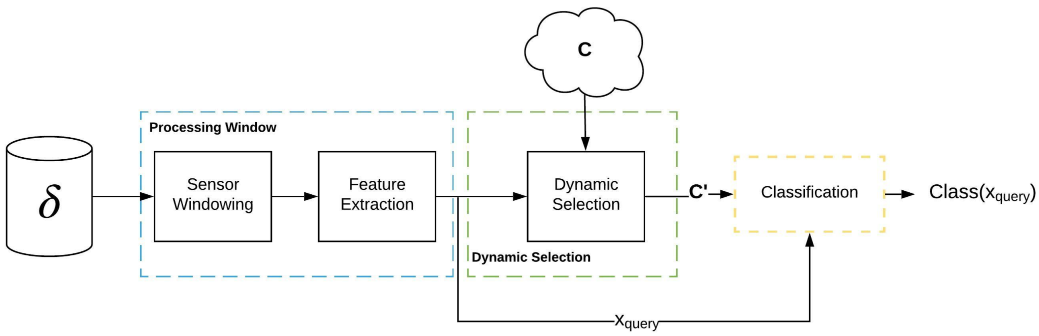


Figure 12 – Test phase. In this phase, the Processing Window is also executed in the Test set, δ. After that, the most competent classifier, or ensemble, C' is selected from the Pool of Classifiers C, based on the approach evaluated. Lastly, the outcome correspondent to the label of x_{query} is defined.

Figure 12 shows the data flow during the Test phase. Similarly to the Training phase, the data from the test set, δ, are transformed into a set of feature vectors in the Processing Window. After the Feature Extraction, the query example x_{query} is presented to be classified by the Dynamic Selection module. In this step, the most competent classifier, or ensemble, is selected from the pool C to classify the x_{query}.

5 EXPERIMENTS AND DISCUSSION

5.1 INTRODUCTION

In Chapter 2, the background of this project was presented, focusing on pool generation techniques and the DS approaches. Besides the well-known methods, such as Bagging, the SGH method was introduced as another method in the comparison and evaluated. In Chapter 3, the related works in Activity Recognition were presented to contextualize what it is researching in the academy to handle this problem. In Chapter 4, an architecture to evaluate the performance of DS approaches in noisy environments was presented. A random label change occurs in six noise levels to simulate the noisy environment.

In this section, we describe the parameters of our experiment: the data sets, machine learning techniques parameters, and the other experimental configurations. Afterward, the results are exhibited and discussed.

This chapter is organized as follows: In Section 5.2, the Experimental Protocol is described. Section 5.3 goes deeper into the evaluation of the proposed method in different configurations. Also, in this section, it is presented a discussion about DS techniques performance.

5.2 EXPERIMENTAL PROTOCOL

5.2.1 Data sets

Five data sets from CASAS project[1] are used in the experiments: HH103, HH124, HH129, Kyoto2008, and Kyoto2009. These databases are composed of activities, such as “Sleep”, “Brushing Teeth”, among others, that are being performed in smart homes and captured by sensors. Table 3 describes the main characteristics of the datasets.

Table 3 – CASAS dataset description. The number of residents and sensors, raw sensor data, distinct activities, the percentage of majority and minority class, and the imbalance ratio for the five datasets.

Dataset	# Residents	# Features	# Examples	# Classes	Largest class %	Smallest class %	IR
HH103	1	58	133713	30	16.45	0.01	1645.0
HH124	1	83	60790	23	79.22	0.01	7922.0
HH129	1	65	173000	33	33.61	0.02	1681.0
Kyoto2008	2	54	16736	5	52.17	1.33	39.0
Kyoto2009	2	74	110401	16	23.84	0.16	149.0

The datasets adopted in this project consist of sensor events. Let $S = \{s_1, s_2, \cdots, s_n\}$ be a sequence of n sensor events composed of a calendar date, time of the day, sensor

[1] Available at http://casas.wsu.edu

name and sensor status. Figure 13 shows a fragment of one of the datasets used in this work. The data represents calendar date, composed of year, month, and day; time of the day, which consists of an hour, minute, second and millisecond; sensor's identification, and value in which can be nominal, ON/OFF, or numerical. A block that delimits the beginning and end of activity defines the activity label of sensor events. Otherwise, they are unlabeled. For instance, in Figure 13, we can see the beginning of the "sleep" activity.

```
2011-06-15 00:03:09.817697 LS006 0
2011-06-15 00:17:44.211833 LS005 4
2011-06-15 00:23:09.72767 LS006 1
2011-06-15 00:32:12.027662 BATV103 3180
2011-06-15 00:37:44.063746 LS005 5
2011-06-15 00:43:09.615905 LS006 0
2011-06-15 00:47:43.992647 LS005 4
2011-06-15 01:03:09.498286 LS006 1
2011-06-15 01:07:43.887796 LS005 5
2011-06-15 01:17:43.846326 LS005 4
2011-06-15 01:23:09.380385 LS006 0
2011-06-15 01:27:43.776285 LS005 5
2011-06-15 01:37:43.727408 LS005 4
2011-06-15 01:43:09.302783 LS006 1
2011-06-15 01:43:14.277702 M015 ON Sleep="begin"
2011-06-15 01:43:17.068053 M015 OFF
2011-06-15 01:47:43.659496 LS005 5
2011-06-15 01:53:09.245733 LS006 0
2011-06-15 01:57:43.601974 LS005 4
2011-06-15 02:01:31.402581 M015 ON
```

Figure 13 – Fragment of a sequence of sensor events extracted from smart homes from the CASAS dataset.

Figure 14 shows the floor plan and sensor layout for the apartments. During six months, the data was collected in the flats while the residents lived there and performed their regular daily routines.

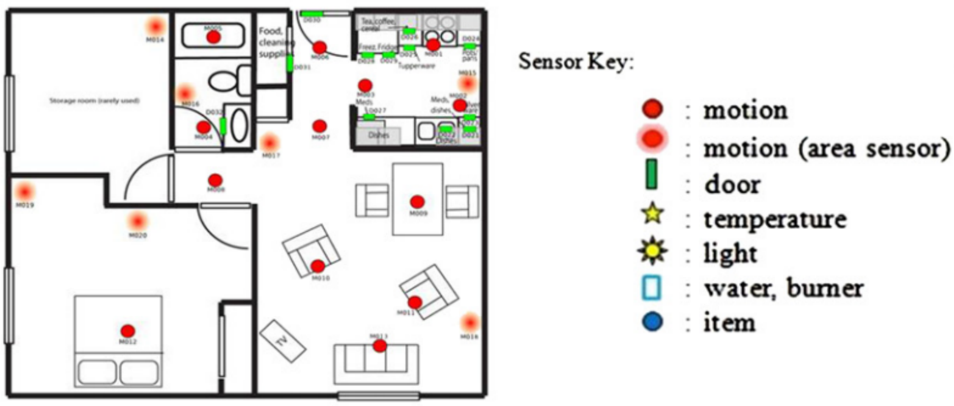


Figure 14 – Example of CASAS environment (CHEN; DAS; COOK, 2010). The datasets are composed of the sensors' information captured in a certain period.

In Table 3, the Activity Recognition datasets present a considerable degree of class imbalance, named Imbalance Ratio (IR). This feature refers to the ratio between the number of examples in the majority class and the number of samples in the minority class (ORRIOLS-PUIG; BERNADO-MANSILLA, 2009).

As detailed in Table 3, some datasets presents a high rate of IR. This phenomenon occurs due to some activities that are more regular and/or longer than others. For instance, the "Eat" activity usually takes 0.5-1 hours a day for a resident while "Sleep" takes 6-8 hours approximately.

5.2.2 Parameters setting

The proposed method requires a configuration to perform the evaluation. In the Training phase, it is necessary to define the Window size, the noise level to be inserted, and the pool generation technique. Lastly, in the Test phase, the DS approach, the region of competence size, and the aggregation method should be defined.

Window sizes between 5 and 30 sensor events are commonly adopted (COOK; KRISHNAN, 2015). As the objective of this work is not to optimize or identify the window size, no investigation was performed with this purpose. Thus, in our experiments, the window size chosen was 30, as defined in (KRISHNAN; COOK, 2014).

Regarding the noise insertion, it was performed six noise levels: 0% to 50%, increasing by 10% per level. Due to this approach, the original dataset produces six new ones. Thereby, the experiments were carried out using five datasets with six noise levels each one, totalizing 30 datasets.

The pool of classifiers used in the experiments was generated according to (CRUZ et al., 2015). Let C a pool of 100 Perceptrons. In our proposed method, three generation techniques was evaluated: Bagging (BREIMAN, 1996), AdaBoost (FREUND; SCHAPIRE, 1997), and SGH. In this latter approach, the pool's size is determined automatically.

The DS techniques consider the Region of Competence to select the most competent classifier, or ensemble, to predict the class of the query example x_{query}. The data set used to define the RoC is called D_{SEL}, and in our experiment is the same as D_{train}, the training data set. The size of the neighborhood in our methodology was established by the 7-nearest neighbors of x_{query}, as described in (CRUZ; CAVALCANTI; REN, 2011).

Since the Random Forest parameters used in the (ALBERDI et al., 2018) was not provided, the Grid Search was used to establish its configuration in our experiments. In this method, the grid evaluates the possible parameters, and the best setting is chosen as the final configuration. As each noise level represents a different data set, the grid search, in our approach, defines the number of trees for each fold on each noise level on each data set.

5.2.3 Evaluation

The cross-validation technique is commonly used to evaluate the performance of a method. In our proposed approach, the data sets were split in 5-folds: 4-folds to training the classifiers, and 1-fold for testing.

Some evaluation metrics were adopted in our project: For each fold and noise rate, the F-measure, and Multi-Label F-measure were calculated.

F-measure is suggested in the literature to integrate precision and recall as an average, which assigns equal importance for Recall and Precision. Therefore, F-measure is defined as follows:

$$\textit{F-Measure} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5.1}$$

Considering that F-measure is, commonly, used to two-class learning problems, the Multi-label F-measure can be used as an variation of F-measure to multi-class imbalance problems. MFM is defined for multi-class problems as follows:

$$MFM = \frac{\sum_{i=1} F - measure_i}{m} \tag{5.2}$$

where, i is the index of the class considered positive.

These metrics are computed for each class, and the mean value is taken among the activity classes. Due to the high degree of imbalance of the data sets, all techniques were evaluated considering the Micro approach: The average of each class contribution is used to compute the metric and exposes the effect of noise insertion. Besides these metrics, the Confusion Matrix was also calculated to understand the behavior of the algorithms for each class. For more details, see the attached Appendix D. It is essential to point out that the activities labeled as “Other" were not considered in these calculations.

Finally, the mean among the folds of the same noise rate is calculated to compute the average performance of the machine learning algorithm.

5.3 RESULTS AND DISCUSSION

The experiments conducted in this proposed method consist of an evaluation and analysis of the impact of increasing noise level in Activity Recognition problems. Considering that the parameters were previously defined, the next sections will describe the steps over the test set, and the results achieved in each one. Moreover, a deep evaluation of the OLA, and LCA behavior are presented.

5.3.1 Comparative Study

In this section, to understand the behavior of the DS approach to handle noisy environments, an evaluation to define the most suitable pool generation technique, and further, identify the DS method appropriate was performed. The choice of static methods, Bagging, and AdaBoost, is related to results achieved in diverse kinds of problems, as stated in (CRUZ et al., 2015). Thus, the comparative study evaluates the SGH method, which presents an alternative for the generation of a pool of classifiers. Recent studies showed the benefits of OLP, which uses the SGH as generation method, on imbalanced data sets, as reported in (SOUZA et al., 2019a).

The performance of pool generation techniques using the Oracle model is evaluated in Section 5.3.1.1. Afterwards, Section 5.3.1.2 presents the DCS performance compared to Random Forest using the SGH approach to generate the pool of classifiers. Lastly, the statistical analysis is performed in Section 5.3.1.3.

5.3.1.1 Oracle Evaluation

An inspection of the pool of classifiers techniques was conducted considering the abstract Oracle model in six noise rates to evaluate the effect of the generation phase. In short, the purpose of this study is to define the most suitable generation method for noisy environments in the Activity Recognition problem.

As previously defined, the Oracle model seeks to find a pool of classifiers capable of correctly predicting all instances in the training set. Otherwise, the classifiers are randomly inserted in the pool, and, because of this, Oracle rates occasionally do not reach 100%. Therefore, as the Oracle is closest to the maximum, the pool is more diverse and accurate.

Hence, the objective of this study is to correlate the pool generation techniques with the Oracle rate. The approach which achieves the best results is selected to generate the pool. For this, considering the HH124 database, which has the highest imbalance ratio, an evaluation of the performance of the abstract model was made as the noise level increased for each generation method.

The following generation methods are chosen to evaluate the performance of these experiments: Bagging, AdaBoost, and SGH. For the baseline methods, a pool composed of 100 Perceptrons was defined. According to its definition, the SGH selects the Perceptrons needed to achieve the 100% Oracle accuracy rate. Thus, the number of Perceptrons on the pool of classifiers is defined during the training process. Figure 15 presents pool size defined by SGH for each data set considering the noise insertion.

Figure 16 shows that the Oracle presents an accuracy near to 100% when the SGH is the generation method in test sets. However, despite high results, as noise increases, the performance of other techniques is affected, showing a decline. Thus, due to the results, the

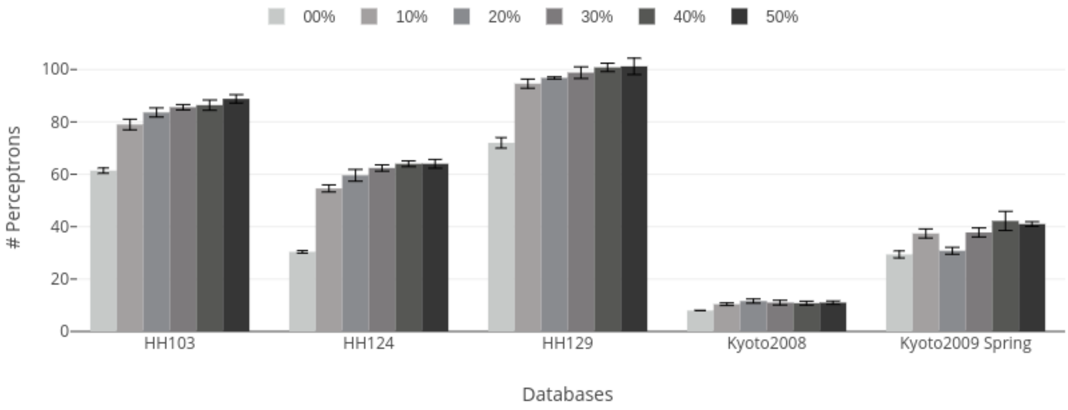


Figure 15 – Size of the pool of classifiers defined by SGH for each dataset.

SGH technique was selected to generate the classifier pool. For more details, in Appendix A there are the results for the remaining data sets.

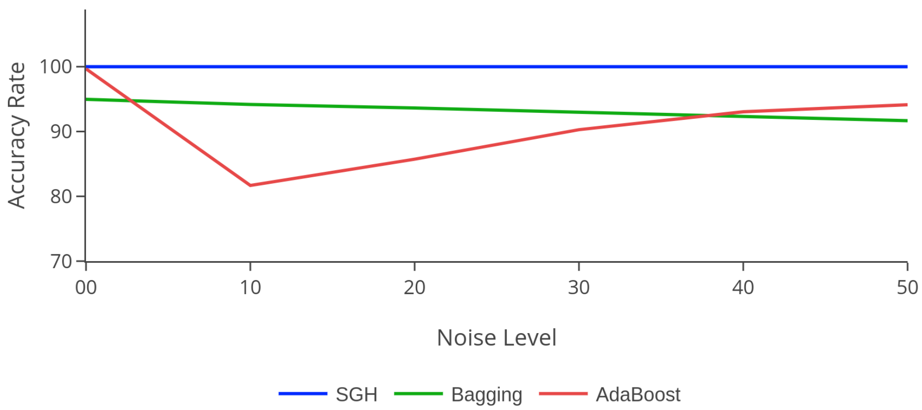


Figure 16 – Oracle evaluation in HH124 database considering three generations of pool algorithms: SGH, AdaBoost, and Bagging.

5.3.1.2 Dynamic Selection

Considering that the pool generation technique that had the best results in Oracle evaluation was chosen, the objective of this section is to evaluate the performance of DS methods as the noise increases. Thus, considering the CASAS data set, Table 4 describes the accuracy rate of the DS in comparison with Random Forest, the technique which achieved the most accurate results, according to (ALBERDI et al., 2018).

Regarding the performance of the proposed method, the DS techniques were compared to the Random Forest algorithm. In this work, four DCS algorithms were evaluated: OLA, LCA, Rank, and MCB. Besides, six DES techniques were also investigated: KNORAU, KNORAE, DESKNN, DESP, METADES and KNOP.

From Table 4 it can be observed that MCB and OLA, at high noise levels, most of the time, outperform the Random Forest results. According to Table 3, the data sets evaluated are highly imbalanced due to some activities which are more common or spent more time to be executed than others. However, in cases where the majority class corresponds to a large part of the data, the algorithms yield similar results. Thus, when the amount of samples correspondent to the majority class, the techniques become experts. This fallacy is named Overfitting. However, some techniques have difficulty separating classes when data is more balanced. In general, DS techniques present stable results due to the use of neighbors' information to select the most competent classifiers. Appendix B presents the visualization of the evaluation following depicted.

Despite not presenting the highest rate of imbalance, the performance of DS techniques in HH103 and HH109 is similar to the Random Forest. It can be observed that is not affected by the high noise levels, remaining steady even at the maximum noise level, 50%. In other data sets, the results for these two techniques present stability as the noise increases.

Noise Level	Random Forest	OLA	LCA	Rank	MCB	KNORAU	KNORAE	KNOP	DESKNN	DESP	METADES
HH103											
00	**91.53 (0.09)**	81.73 (0.25)	64.0 (0.4)	88.12 (0.23)	81.57 (0.26)	79.8 (0.31)	88.33 (0.22)	19.8 (1.5)	37.51 (7.27)	63.91 (0.47)	20.8 (1.81)
10	**83.11 (0.29)**	80.56 (0.26)	42.53 (0.59)	71.72 (0.19)	80.48 (0.32)	78.98 (0.65)	72.06 (0.2)	19.99 (0.94)	30.84 (4.04)	47.02 (1.29)	21.41 (1.22)
20	75.58 (0.25)	79.13 (0.2)	32.05 (0.57)	62.35 (0.1)	**79.17 (0.23)**	77.43 (0.27)	62.63 (0.11)	19.31 (1.88)	26.83 (2.55)	36.81 (0.44)	21.11 (2.18)
30	68.5 (0.22)	76.65 (0.2)	26.11 (0.5)	54.76 (0.16)	**77.12 (0.2)**	75.4 (0.7)	55.02 (0.16)	18.07 (0.85)	25.74 (1.72)	32.17 (0.86)	19.35 (0.83)
40	62.13 (0.34)	73.75 (0.2)	21.67 (0.52)	48.66 (0.15)	**74.13 (0.26)**	72.07 (0.74)	48.94 (0.16)	17.34 (1.29)	22.85 (2.17)	27.49 (0.65)	17.82 (1.35)
50	56.17 (0.38)	69.96 (0.28)	18.17 (0.41)	43.47 (0.16)	**70.54 (0.39)**	68.05 (1.22)	43.71 (0.16)	16.68 (0.87)	20.57 (1.75)	23.83 (2.01)	16.62 (0.97)
HH124											
00	**95.02 (1.76)**	88.79 (1.35)	80.29 (2.3)	92.31 (1.73)	88.99 (1.17)	87.11 (1.83)	92.47 (1.91)	16.36 (1.21)	31.43 (2.15)	77.46 (2.24)	14.87 (1.22)
10	81.46 (1.03)	84.34 (0.75)	36.77 (1.17)	51.56 (0.55)	**84.57 (0.89)**	84.43 (1.79)	51.93 (0.64)	5.01 (0.97)	23.95 (2.77)	36.71 (2.87)	5.06 (1.09)
20	68.62 (0.98)	82.35 (0.72)	23.6 (1.03)	39.54 (0.36)	82.48 (0.82)	**82.9 (1.48)**	39.96 (0.35)	4.46 (0.56)	19.35 (2.75)	28.44 (2.42)	4.21 (0.3)
30	58.68 (1.52)	77.82 (1.92)	15.86 (0.36)	32.48 (0.38)	**79.38 (2.07)**	77.58 (2.0)	32.88 (0.47)	4.04 (0.17)	15.74 (3.55)	22.41 (2.65)	4.04 (0.17)
40	50.09 (0.58)	71.38 (0.88)	11.92 (0.59)	27.72 (0.28)	**72.97 (1.36)**	70.06 (5.1)	27.9 (0.32)	4.03 (0.17)	12.1 (1.8)	17.46 (2.65)	4.04 (0.21)
50	42.7 (0.81)	62.68 (0.84)	9.63 (0.52)	23.85 (0.39)	**64.93 (0.75)**	64.92 (3.75)	24.13 (0.42)	4.26 (0.3)	13.47 (1.81)	16.34 (2.14)	4.21 (0.25)
HH129											
00	**93.87 (0.45)**	88.11 (0.53)	72.58 (0.58)	91.17 (0.3)	87.59 (0.42)	86.61 (0.57)	91.59 (0.37)	7.36 (1.07)	38.14 (2.32)	72.5 (1.07)	7.28 (1.03)
10	84.12 (0.55)	85.81 (0.43)	45.56 (0.43)	71.5 (0.46)	**86.63 (0.4)**	85.95 (0.15)	72.09 (0.43)	7.57 (0.48)	33.47 (2.34)	51.11 (0.85)	7.59 (0.52)
20	76.44 (0.46)	84.86 (0.47)	31.46 (0.42)	61.47 (0.33)	**85.4 (0.4)**	84.46 (0.65)	61.95 (0.3)	6.42 (0.34)	27.82 (2.08)	38.98 (0.8)	6.35 (0.38)
30	69.32 (0.33)	82.04 (0.18)	24.11 (0.39)	53.72 (0.16)	**82.67 (0.42)**	81.46 (1.44)	54.08 (0.16)	6.87 (0.45)	25.99 (2.47)	31.99 (2.07)	6.8 (0.54)
40	62.51 (0.36)	78.26 (0.33)	19.19 (0.35)	47.36 (0.24)	**79.44 (0.34)**	76.37 (1.11)	47.68 (0.24)	6.43 (0.6)	22.27 (3.17)	26.36 (2.11)	6.03 (0.63)
50	56.21 (0.2)	74.05 (0.42)	16.12 (0.4)	42.16 (0.23)	**75.47 (0.62)**	72.03 (1.72)	42.5 (0.19)	5.14 (0.43)	19.41 (2.41)	22.68 (2.11)	4.55 (0.44)

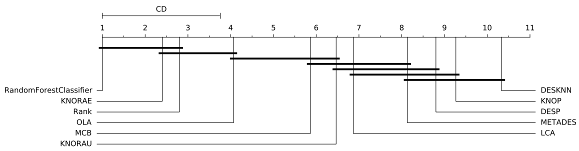
Noise Level	Random Forest	OLA	LCA	Rank	MCB	KNORAU	KNORAE	KNOP	DESKNN	DESP	METADES
Kyoto2008											
00	**99.56 (0.15)**	98.58 (0.52)	95.68 (0.44)	99.08 (0.13)	98.47 (0.5)	97.76 (0.61)	99.09 (0.13)	54.21 (1.59)	68.97 (1.58)	95.69 (0.43)	50.68 (4.94)
10	92.83 (0.5)	98.04 (0.37)	68.02 (1.5)	82.39 (0.7)	**98.32 (0.39)**	96.8 (0.47)	82.5 (0.74)	50.08 (3.2)	74.57 (8.83)	89.86 (1.48)	51.65 (5.58)
20	85.84 (0.83)	**97.79 (0.31)**	54.3 (3.02)	72.71 (0.79)	97.62 (0.49)	95.42 (1.15)	72.79 (0.69)	37.6 (5.04)	71.63 (5.21)	88.78 (1.23)	40.9 (5.96)
30	78.87 (0.89)	95.15 (0.83)	47.04 (2.6)	65.23 (0.87)	**95.54 (0.73)**	91.68 (2.73)	65.29 (0.93)	39.32 (2.43)	69.31 (9.42)	81.08 (3.44)	35.93 (5.64)
40	73.58 (0.69)	91.72 (1.11)	40.97 (4.4)	59.74 (1.09)	**92.51 (1.67)**	85.55 (2.71)	59.79 (1.11)	33.64 (2.31)	57.95 (3.47)	74.58 (2.78)	32.4 (4.83)
50	68.03 (1.03)	86.54 (0.79)	33.62 (3.6)	55.07 (1.07)	**88.02 (0.9)**	79.21 (2.41)	55.15 (1.02)	28.58 (4.23)	56.7 (6.55)	67.53 (2.6)	31.01 (5.69)
Kyoto2009Spring											
00	**99.47 (0.1)**	96.31 (0.23)	91.28 (0.22)	98.32 (0.15)	96.33 (0.25)	95.62 (0.32)	98.33 (0.17)	46.11 (2.32)	49.62 (5.17)	90.45 (0.41)	46.55 (1.64)
10	92.62 (0.4)	**95.95 (0.35)**	59.91 (0.33)	83.13 (0.13)	95.86 (0.34)	95.6 (0.2)	83.22 (0.11)	50.85 (1.62)	53.15 (1.85)	69.57 (2.1)	48.7 (1.66)
20	86.02 (0.38)	**95.53 (0.22)**	43.26 (0.49)	73.47 (0.22)	95.41 (0.23)	94.14 (0.42)	73.57 (0.21)	46.97 (0.81)	39.23 (3.91)	53.42 (1.7)	46.33 (0.94)
30	79.47 (0.6)	94.43 (0.15)	35.16 (0.76)	65.56 (0.08)	**94.73 (0.03)**	92.8 (0.8)	65.65 (0.09)	45.02 (1.51)	40.2 (2.42)	46.76 (1.12)	33.26 (4.41)
40	73.1 (0.47)	91.99 (0.18)	29.21 (1.04)	59.06 (0.17)	**92.45 (0.15)**	90.13 (2.48)	59.15 (0.19)	46.99 (3.03)	35.18 (6.79)	39.42 (5.87)	30.85 (2.34)
50	67.3 (0.28)	88.39 (0.27)	22.87 (0.79)	53.39 (0.31)	**89.05 (0.43)**	84.42 (3.05)	53.47 (0.31)	43.81 (2.2)	28.66 (2.91)	31.26 (3.49)	27.69 (2.61)

Table 4 – Mean and Standard deviation of the Average of Multi-label F-measure rate using the DCS techniques: OLA, LCA, MLA, Rank, and MCB. Also, it was evaluated the DES techniques: KNORA-Union, KNORA-Elimination, KNOP, DESKNN, DESP, and METADES. Since the SGH outperformed the results of Bagging and Adaboost, was used as the pool generation approach. The Random Forest was compared with the results achieved.

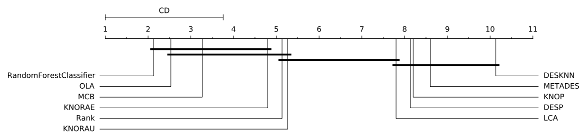
In Section 5.3.2, an in-depth evaluation of the performance of OLA and LCA it will be discussed due to opposite behaviors as the noise increases.

5.3.1.3 Statistical Analysis

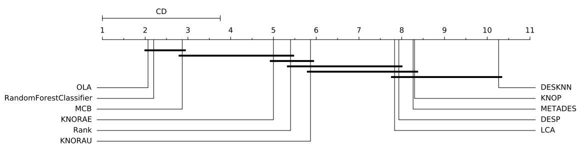
For the statistical comparison of the Dynamic Selection methods and Random Forest over the 5 classification data sets, the Friedman test was used (FRIEDMAN, 1937). Considering each data set, the average rank is calculated to evaluate the method that achieves the best performance. Then, the algorithm that achieves the best result is given rank 1, the second receive rank 2, and so forth. In the case of a tie, i.e., two methods presented the same classication accuracy for the data set, their average ranks were summed and divided by two.



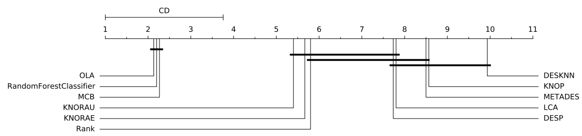
(a) 00%



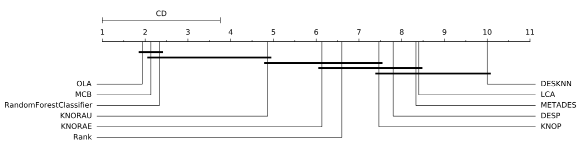
(b) 10%



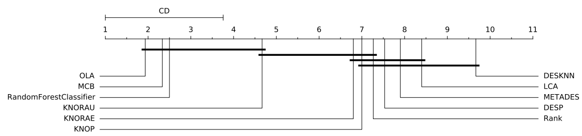
(c) 20%



(d) 30%



(e) 40%



(f) 50%

Figure 17 – Critical difference diagram representing the results of a post-hoc Bonferroni-Dunn test on the accuracy rates of the methods from Table 4. The calculated critical difference value was CD = 1.1. The values near the methods' labels indicate their average rank. Statistically similar methods are connected by an horizontal line, while statistically different ones are disconnected

After this, the critical difference (CD) value was calculated using the Bonferonni-Dunn post-hoc test recommended in (DEMSAR, 2006). In the Bonferonni-Dunn post-hoc test, two configurations are different if their average rank difference is greater than the critical difference CD. We use the critical difference diagram proposed in (DEMSAR, 2006) to have a visual illustration of the statistical test. The CD diagram with the results of the Bonferonni-Dunn post-hoc test is shown in Figure 17.

From the figures, it is possible to see that, as noise levels in datasets increase, the performance of the Random Forest algorithm decreases with the noise iterations. On the other hand, MCB and OLA have stable performance and, even at high noise levels, the accuracy outperforms the Random Forest results. The $CD = 2.27$ is common to all figures. The configurations with no significant difference are connected by a bar, while significantly different ones are not intersected in the diagram.

5.3.2 Discussion

This section presents a deep evaluation of the OLA and LCA performance for the previous experiments. Despite these techniques evaluating the performance of a classifier using accuracy as an approach, the methodology is different. It can be observed in Table 4 that the accuracy rate achieved by OLA surpasses the results of LCA. To understand these differences, the purpose of this evaluation is to analyze the behavior of the techniques as the noise increases over classes and the classifier competence when the neighbors have the same class.

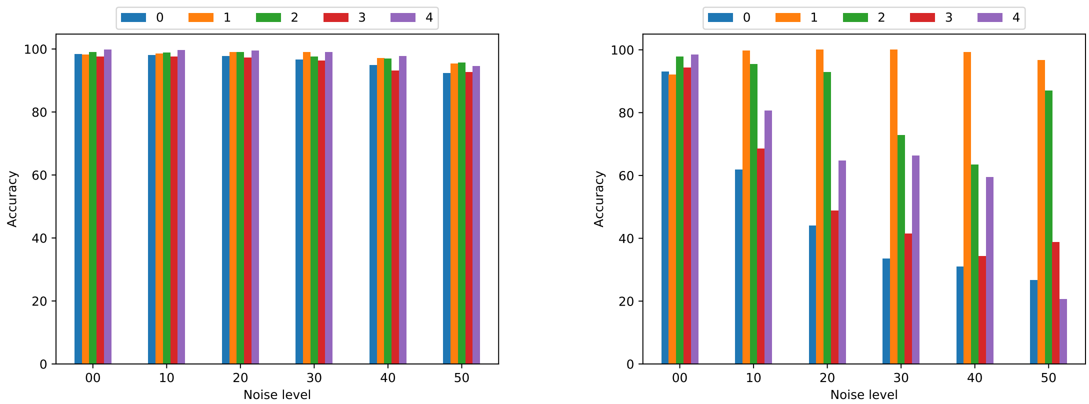
In order to analyze and evaluate the performance of OLA and LCA, two experiments were conducted on the Kyoto 2008 dataset:

1. Performance over the activity classes;
2. Performance over the neighbor.

The Kyoto2008 dataset contains 5 classes. Each class corresponds to an activity executed by the two residents living in this environment. The technical features available on this dataset can be found below:

- Number of Instances: 16,736
- Classes distribution:
  - 0: 3255; 1: 279; 2: 5360; 3: 1081; 4: 10948
- Imbalanced Ratio: 39.0
  - Minority Class: 1.33%
  - Majority Class: 52.17%

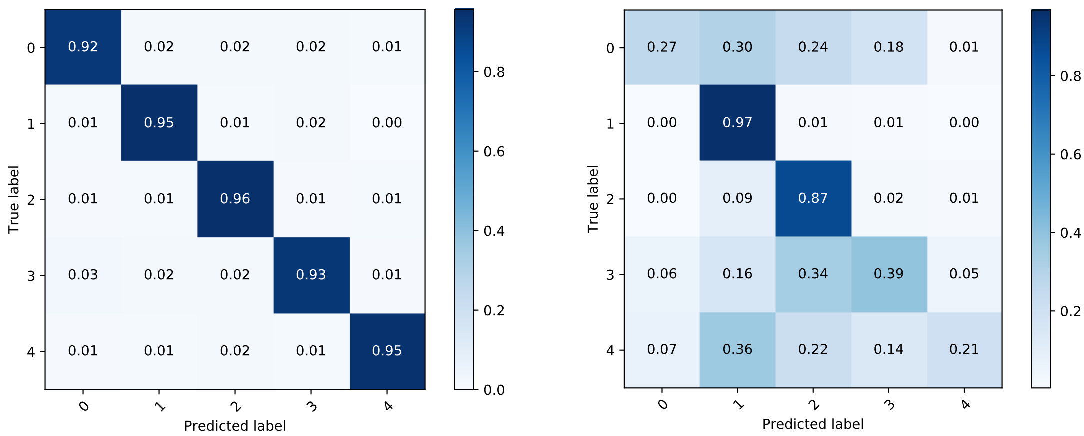
According to Figure 18, as the noise increases, the OLA algorithm achieved better results than the LCA. In this evaluation, the noise increase affects, significantly, the performance of the majority classes in LCA. However, OLA remains stable. The difference can be sharply seen in the first round of noise additions: The LCA performance is heavily affected, whereas the OLA performance is not.



(a) OLA (b) LCA

Figure 18 – Accuracy rate. OLA and LCA accuracy evaluation on the noise augmentation in RoC for instances that present the same classification, in which the labels are defined 0 to 4.

Regarding the 50% noise level, it can be noted that the LCA was drastically affected in the majority class (class 4), while the OLA remains with stable results. Figure 19 details the performance of the algorithms to comprehend how the techniques behave in a noisy scenario. In our evaluation, each row of the matrix corresponds to the actual class and each column to the classification task. It can be seen that LCA presents a misclassification in most categories, whereas the OLA remains consistent even in the minority class.

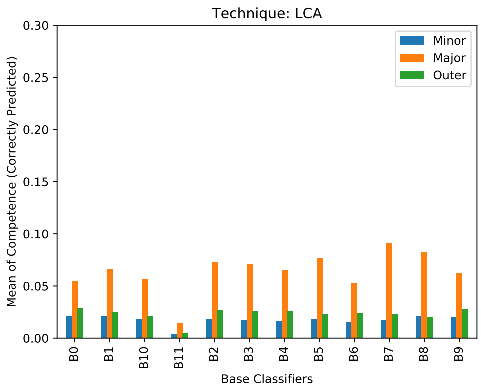


(a) OLA (b) LCA

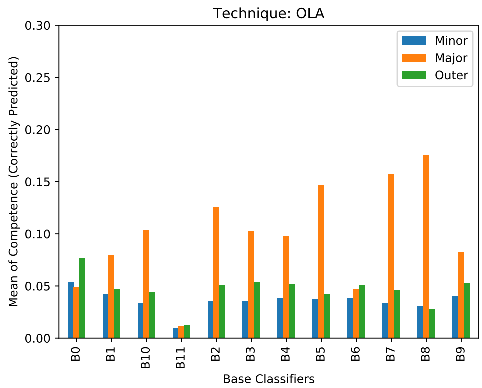
Figure 19 – Normalized confusion matrix for Kyoto2008 dataset considering 50% of noise insertion. Class 4 is the majority, while class 1 is the minority. The OLA technique presents consistent results even in a high noise rate, while the LCA is significantly affected.

Hence, an in-depth evaluation of the base classifiers' competences can explain why in the majority classes, the results are discrepant in OLA and LCA. The mean competence was estimated considering when Target and Predictions are the same. The following three cases were found when the predictions refer to Minority class, Majority class, and Other classes.

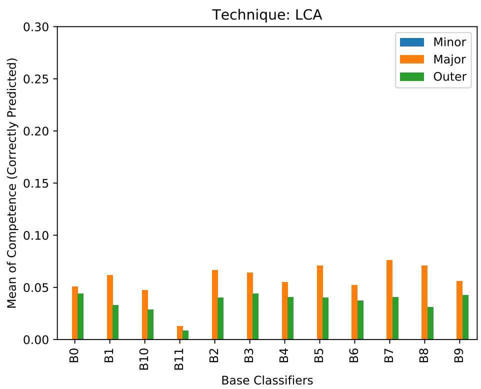
In Figure 20, each one of the 12 base classifiers, B_n, that compose the Pool C was evaluated to classify correctly: the majority class, minority class, and the other one into the RoC. It can be noted that even with six of the seven neighbors agreeing with the label, the mean competence of the base classifiers selected by LCA is lower than that of OLA. Moreover, the behavior of the base classifiers remains similar among the techniques.



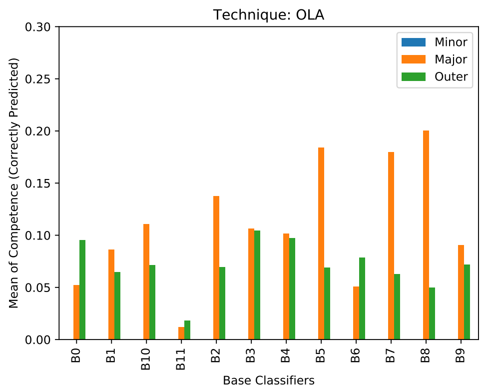
(a) Class: 4, Neighbors: 5



(b) Class: 4, Neighbors: 5



(c) Class: 4, Neighbors: 6



(d) Class: 4, Neighbors: 6

Figure 20 – Mean of competence to Base Classifiers. OLA and LCA evaluation to the highest noise level considering five and six neighbors with the same class.

It is possible to conclude that the classifiers selected by OLA do not consider the specialization in its evaluation. Although the noise insertion was the same, the OLA does not find the number of samples for each class on its assessment and, because of this, the performance is not significantly affected as the LCA. Therefore, in scenarios where there is an increase of noise in smart environments, and when there is a variation between the frequency of activities performed inside the home, algorithms similar to OLA present more stable and accurate results.

In this evaluation, we could add a similar performance obtained by the MCB technique. As we previously presented in 2, the MCB has a similar approach to OLA: Based on the number of hits in the RoC, the classifier most competent, or well-ranked, is selected from the pool of classifiers. However, MCB has a slight difference in their RoC: It prunes the instances which don't have related behavior. Considering that we have an environment even more polluted as the noise increases, this pre-step might be responsible for the best performance of the classifier selected among the others. So, we can see this fine difference in Table 4.

6 CONCLUSION

In this work, it was evaluated the MCS techniques in the context of Activity Recognition in Smart Homes, one of the applications of IoT technologies. The literature reports the usage of machine learning techniques to predict the activities performed in Smart Homes, and the challenge to improve their precision when it has noise insertion due to misleading data capture. The CASAS project, the main dataset used in our approach, was randomly changed to simulate a noise insertion to evaluate the DCS techniques performance in contrast to Random Forest's accuracy.

The stages of an MCS approach were presented: Generation, Selection, and Integration. The pool generation of classifiers was detailed, and evaluation of Oracle accuracy rate in known techniques, such as Bagging and Adaboost, was made in comparison to SGH performance. Further, the process of dynamically selecting classifiers was introduced, and the DCS techniques evaluated in our project were introduced. Finally, a discussion over the robustness of ensemble techniques in the noisy environment was raised, and the evaluation has shown that even when the half of labels are randomly changed, the DS techniques presents stable results in comparison to Random Forest.

Premised on the challenge to recognize and predict activities in noisy environments, the proposed method was presented. Firstly, the generation techniques must guarantee an accurate and diverse pool of classifiers. Thereby, the generation techniques were evaluated and the most suitable, which has the highest Oracle, to fully cover the feature space is selected. In our proposed method, the SGH had the highest Oracle accuracy ratio. After that, experiments were conducted over 30, five smart home data sets each one with 6 different insertion noise rates, including a noise-free scenario. The accuracy, precision and recall were performed in five machine learning techniques: Random Forest, OLA, LCA, Rank, and MCB.

The results have shown that, in most cases, DCS techniques are less sensitive to noise insertion than Random Forest. We observed that the configurations SGH+MCB and ,SGH+OLA present the most stable results even with a 50% noise level. However, even though the LCA considers the accuracy as the criteria to evaluate the performance of a classifier as OLA, its performance was highly impacted by the noise insertion. Thus, to comprehend the decision made by LCA, a deep evaluation of the competence of the classifiers selected was performed. As exhibited previously, the LCA was not able to select the classifiers most accurately to fit the problem as the OLA did. The behavior of the base classifiers remains similar among the techniques even though achievements are different.

As the SGH is a pool generation method that defines a centered hyperplane to separate the classes, boundaries instances may represent a challenge. Related to this scenario, the future works may lead to the representation of the hyperplane that considers all instances, instead of only centroids, to handle multi-class problems. Similarly to SVM, the proposal idea is to maximize the margin in order to cover the instances more distant of centroids. Besides, due to the generation process used in this proposed method, DES techniques were not evaluated. The DES techniques have shown promise achievements to handle imbalanced data problems, as discussed in (CRUZ et al., 2019). In future experiments, the proposal is to evaluate DES methods with different generation techniques in comparison to DCS techniques.

REFERENCES

AL-FUQAHA, A.; GUIZANI, M.; MOHAMMADI, M.; ALEDHARI, M.; AYYASH, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, IEEE, v. 17, n. 4, p. 2347–2376, 2015.

ALBERDI, A.; WEAKLEY, A.; SCHMITTER-EDGECOMBE, M.; COOK, D. J.; AZTIRIA, A.; BASARAB, A.; BARRENECHEA, M. Smart home-based prediction of multi-domain symptoms related to alzheimer's disease. *IEEE Journal of Biomedical and Helath Informatics*, IEEE, p. 1–1, 2018.

ALKOOT, F.; KITTLER, J. Experimental evaluation of expert fusion strategies. *Pattern Recognition Letters*, Elsevier, v. 20, p. 1361–1369, 1999.

ALMEIDA, P. R. L. de; JúNIOR, E. J. da S.; CELINSKI, T. M.; BRITTO, A. d. S.; OLIVEIRA, L. E. S. de. Music genre classification using dynamic selection of ensemble of classifiers. *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, p. 2700–2705, 2012.

AMINIKHANGHAHI, S.; COOK, D. Enhancing activity recognition using cpd-based activity segmentation. *Pervasive and Mobile Computing*, v. 53, 02 2019.

BREIMAN, L. Bagging predictors. *Machine Learning*, Springer, v. 24, n. 2, p. 123–140, 1996.

CHEN, C.; DAS, B.; COOK, D. J. A data mining framework for activity recognition in smart environments. *2010 Sixth International Conference on Intelligent Environments*, IEEE, p. 80–83, 2010.

CHEN, L.; NUGENT, C. D.; WANG, H. A knowledge-driven approach to activity recognition in smart homes. *IEEE Transaction on Knowledge and Data Engineering*, v. 24, 2012.

COOK, D. J.; CRANDALL, A.; THOMAS, B.; KRISHNAN, N. C. Casas: A smart home in a box. *IEEE Computer*, v. 46, p. 62–69, 2013.

COOK, D. J.; KRISHNAN, N. C. *Activity Learning: discovering, recognizing, and predicting human behavior from sensor data*. [S.l.]: John Wiley & Sons, 2015.

CRUZ, R. M.; CAVALCANTI, G. D.; REN, T. I. A method for dynamic ensemble selection based on a filter and an adaptive distance to improve the quality of the regions of competence. In: IEEE. *International Joint Conference on Neural Networks*. [S.l.], 2011. p. 1126–1133.

CRUZ, R. M.; SABOURIN, R.; CAVALCANTI, G. D. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, v. 41, p. 195 – 216, 2018. ISSN 1566-2535.

CRUZ, R. M.; SABOURIN, R.; CAVALCANTI, G. D.; REN, T. I. META-DES: A dynamic ensemble selection framework using meta-learning. *Pattern Recognition*, Elsevier, v. 48, n. 5, p. 1925–1935, 2015.

CRUZ, R. M.; SOUZA, M. A.; SABOURIN, R.; CAVALCANTI, G. D. Dynamic ensemble selection and data preprocessing for multi-class imbalance learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 2019.

CRUZ, R. M. O.; SABOURIN, R.; CAVALCANTI, G. D. C. Prototype selection for dynamic classifier and ensemble selection. *Neural Computing and Applications*, v. 29, p. 447—-457, 2016.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 7, p. 1–30, 2006.

ESSA, I. Ubiquitous sensing for smart and aware environments. *Personal Communications, IEEE*, v. 7, p. 47 – 49, 11 2000.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, v. 55, p. 119–139, 1997.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, v. 32, p. 675–701, 1937.

GIACINTO, G.; DIDACI, L. Dynamic classifier selection by adaptive k-nearest-neighbourhood rule. *Multiple Classifier Systems*, v. 3077, p. 174–183, 2004.

GIACINTO, G.; ROLI, F. Methods for dynamic classifier selection. *Proceedings 10th International Conference on Image Analysis and Processing*, p. 659–664, 1999.

GIACINTO, G.; ROLI, F. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, v. 34, p. 1879–1881, 2001.

GROVE, A. J.; SCHUURMANS, D. Boosting in the limit: Maximizing the margin of learned ensembles. *American Association for Artificial Intelligence*, p. 692–699, 1998.

HONG, X.; NUGENT, C.; MULVENNA, M.; MCCLEAN, S.; SCOTNEY, B.; DEVLIN, S. Evidential fusion of sensor data for activity recognition in smart homes. *Pervasive and Mobile Computing*, v. 5, p. 236–252, 2009.

HUANG, Y. S.; SUEN, C. Y. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 90–94, 1995.

INTILLE, S.; LARSON, K.; BEAUDIN, J.; TAPIA, E.; KAUSHIK, P.; NAWYN, J.; MCLEISH, T. The placelab: a live-in laboratory for pervasive computing research (video). In: . [S.l.: s.n.], 2010.

IRVINE, N.; NUGENT, C.; ZHANG, S.; WANG, H.; NG, W. W. Y. Neural network ensembles for sensor-based human activity recognition within smart environments. *Sensors*, v. 20, n. 1, p. 216, 2019.

JR, A. S. B.; SABOURIN, R.; OLIVEIRA, L. E. Dynamic selection of classifiers - a comprehensive review. *Pattern Recognition*, Elsevier, v. 47, p. 3665–3680, 2014.

KITTLER, J.; HATEF, M.; DUIN, R. P. W.; MATAS, J. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, p. 226–239, 1998.

KO, A. H.; SABOURIN, R.; BRITTO, A. S. From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, v. 41, p. 1718–1731, 2008.

KRISHNAN, N. C.; COOK, D. J. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, Elsevier, v. 10, p. 138–154, 2014.

KUNCHEVA, L. I. A theoretical study on six classifier fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, p. 281–286, 2002.

LIMA, T. P. F. de; SERGIO, A. T.; LUDERMIR, T. B. Improving classifiers and regions of competence in dynamic ensemble selection. *2014 Brazilian Conference on Intelligent Systems*, p. 13—-18, 2014.

LU, L.; QING-LING, C.; YI-JU, Z. Activity recognition in smart homes. *Multimedia Tools and Applications*, v. 76, p. 24203–24220, 2017.

MATTERN, F.; FLOERKEMEIER, C. From the internet of computers to the internet of things. *From active data management to event-based systems and more*, Springer, p. 242–259, 2010.

MINOR, B.; DOPPA, J. R.; COOK, D. J. Learning activity predictors from sensor data: Algorithms, evaluation, and applications. *IEEE Transactions On Knowledge and Data Engineering*, IEEE, v. 29, p. 2744–2757, 2017.

MOZER, M. C. The neural network house: An environment hat adapts to its inhabitants. *AAAI Spring Symposium Intelligent Environments*, p. 110 – 114, 1998.

Myagmar, B.; Li, J.; Kimura, S. Heterogeneous daily living activity learning through domain invariant feature subspace. *IEEE Transactions on Big Data*, p. 1–1, 2020.

NETTLETON, D. F.; ORRIOLS-PUIG, A.; FORNELLS, A. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, Springer, v. 33, p. 275–306, 2010.

ORRIOLS-PUIG, A.; BERNADO-MANSILLA, E. Evolutionary rule-based systems for imbalanced data sets. *Soft Computing*, Elsevier, v. 13, p. 213–225, 2009.

QUINLAN, J. R. Boosting first-order learning. *International Workshop on Algorithmic Learning Theory*, p. 143–155, 1996.

RATSCH, G.; ONODA, T.; MüLLER, K.-R. Soft margins for adaboost. *Machine Learning*, v. 42, p. 287–320, 03 2001.

ROKACH, L. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *2006 International Joint Conference on Neural Networks*, Elsevier, v. 53, p. 4046—4072, 2009.

SABOURIN, M.; MITICHE, A.; THOMAS, D.; NAGY, G. Classifier combinat ion for hand-printed digit recognition. *Proceedings of the Second International Conference on Document Analysis and Recognition*, p. 163–166, 1993.

SANABRIA, A. R.; YE, J. Unsupervised domain adaptation for activity recognition across heterogeneous datasets. *Pervasive and Mobile Computing*, p. 101–147, 2020.

SCHAPIRE, R. E. The strength of weak learnability. *Machine Learning*, v. 5, p. 197–227, 06 1990.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM Computing Survey*, p. 1–47, 2002.

SOARES, R. G. F.; SANTANA, A.; CANUTO, A. M. P.; SOUTO, M. C. P. de. Using accuracy and diversity to select classifiers to build ensembles. *2006 International Joint Conference on Neural Networks*, IEEE, 2006.

SOUZA, M. A.; CAVALCANTI, G. D.; CRUZ, R. M. M.; SABOURIN, R. On evaluating the online local pool generation method for imbalance learning. *International Joint Conference on Neural Networks*, p. 59–70, 2019.

SOUZA, M. A.; CAVALCANTI, G. D. C. D.; CRUZ, R. M.; SABOURIN, R. Online local pool generation for dynamic classifier selection. *Pattern Recognition*, v. 85, p. 132–148, 2019.

WEISER, M. The computer for the 21 st century. *Scientific American*, JSTOR, v. 265, n. 3, p. 94–105, 1991.

WEISER, M. Hot topics: Ubiquitous computing. *IEEE Computer*, v. 26, n. 10, p. 71–72, 1993.

WOLPERT, D.; MACREADY, W. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, v. 1, p. 67–82, 1997.

WOODS, K.; KEGELMEYER, W. P.; BOWYER, K. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 19, 1997.

WOŹNIAK, M.; GRAÑA, M.; CORCHADO, E. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, Elsevier, v. 16, p. 3–17, 2014.

Wu, X. Mining with noise knowledge: Error aware data mining. In: . [S.l.: s.n.], 2007.

ZHU, X.; WU, X. Artificial intelligence review. *Kluwer Academic Publishers*, p. 117–210, 2004.

APPENDIX A – ORACLE F1 PERFORMANCE IN NOISY ENVIRONMENTS

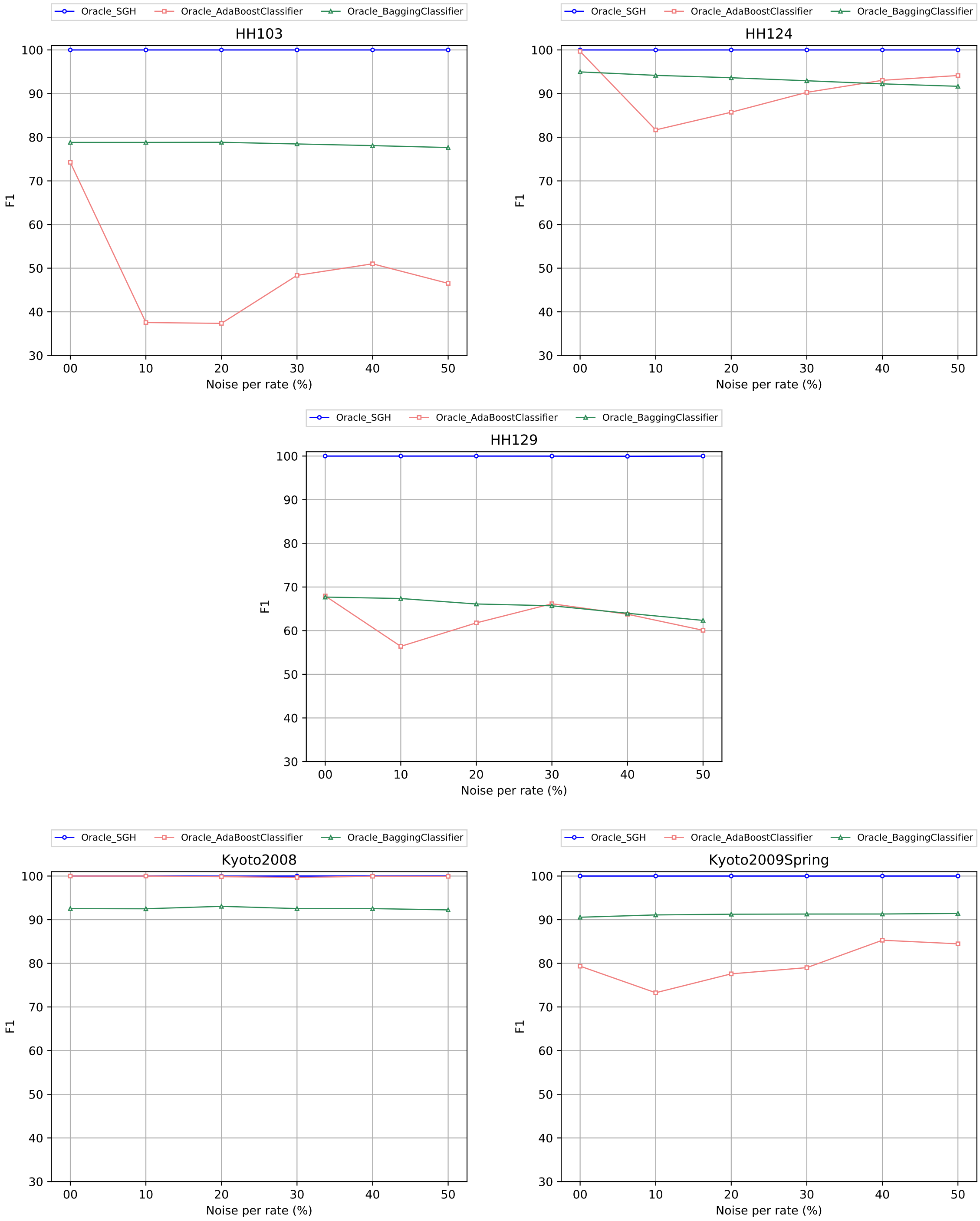


Figure 21 – F1 Score rate. The Selection approaches SGH, Adaboost, and Bagging were evaluated in order to define the most suitable technique to our problem. According to the results achieved, the SGH presented the better rate even in high noise levels.

APPENDIX B – MULTI-LABEL F-MEASURE IN NOISY ENVIRONMENTS ACROSS THE EVALUATED GENERATION METHODS

B.0.1 HH103

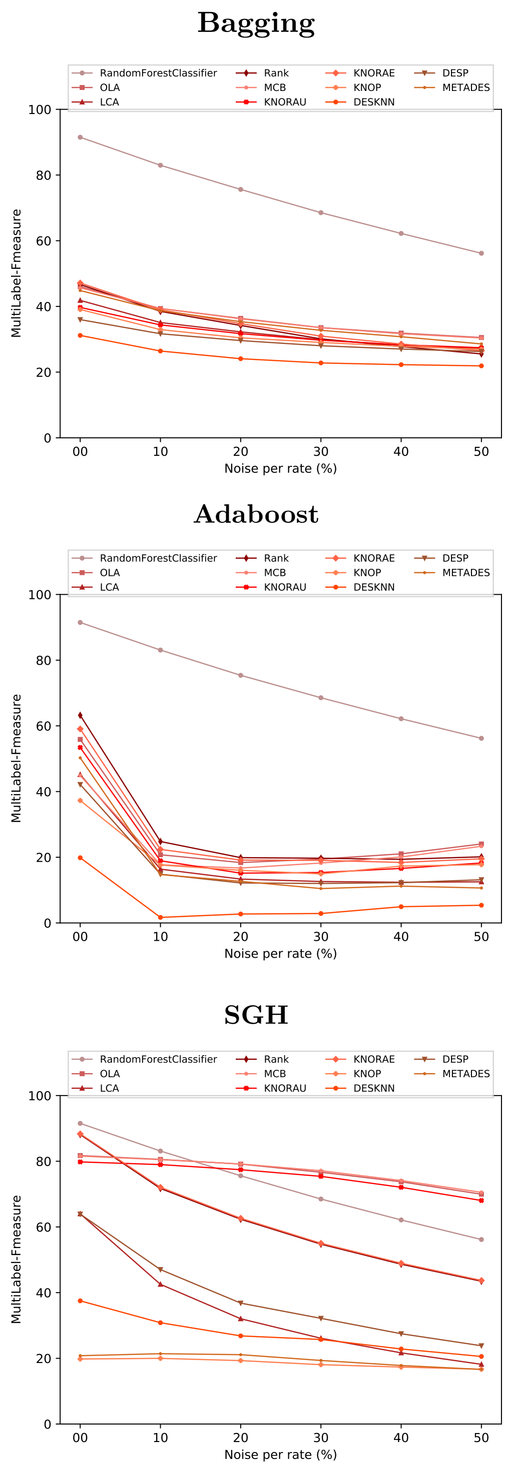


Figure 22 – Multi-Label Fmeasure in a noisy environment to Dynamic Selection techniques in comparison with Random Forest achievements using three Generation methods: Bagging, Adaboost and SGH.

B.0.2 HH124

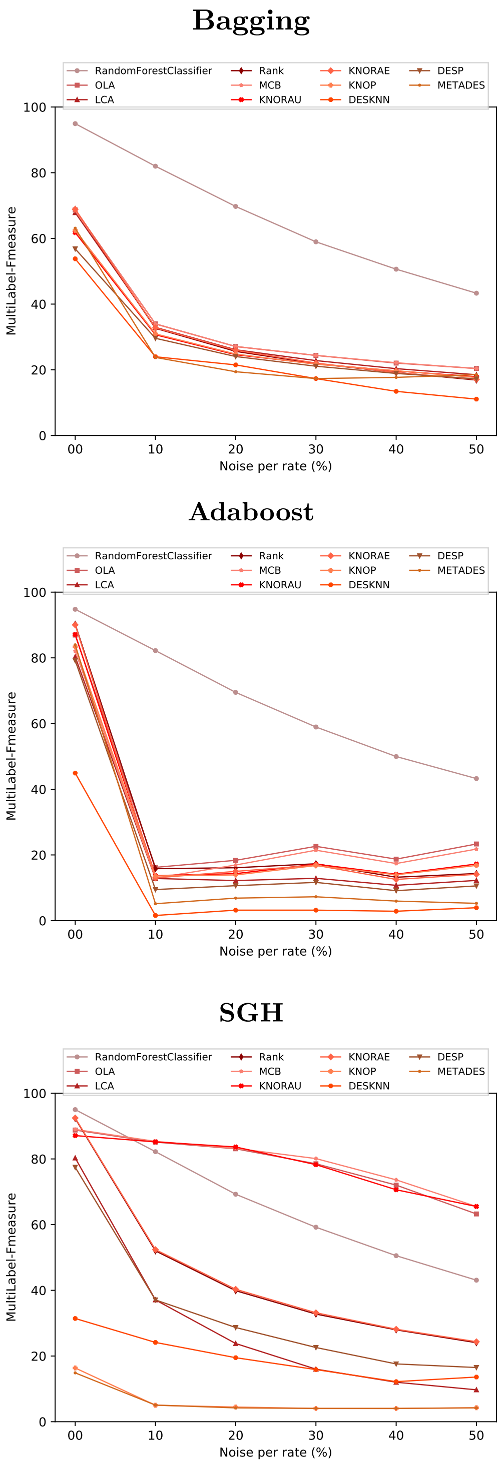


Figure 23 – Multi-Label Fmeasure in a noisy environment to Dynamic Selection techniques in comparison with Random Forest achievements using three Generation methods: Bagging, Adaboost and SGH.

B.0.3 HH129

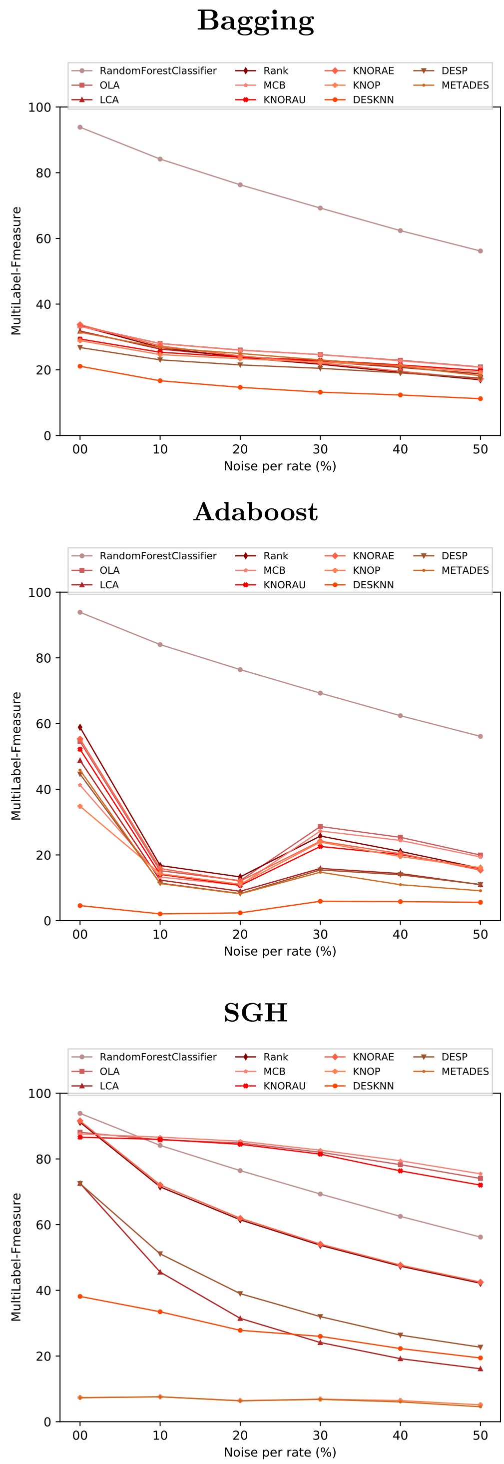


Figure 24 – Multi-Label Fmeasure in a noisy environment to Dynamic Selection techniques in comparison with Random Forest achievements using three Generation methods: Bagging, Adaboost and SGH.

B.0.4 Kyoto 2008

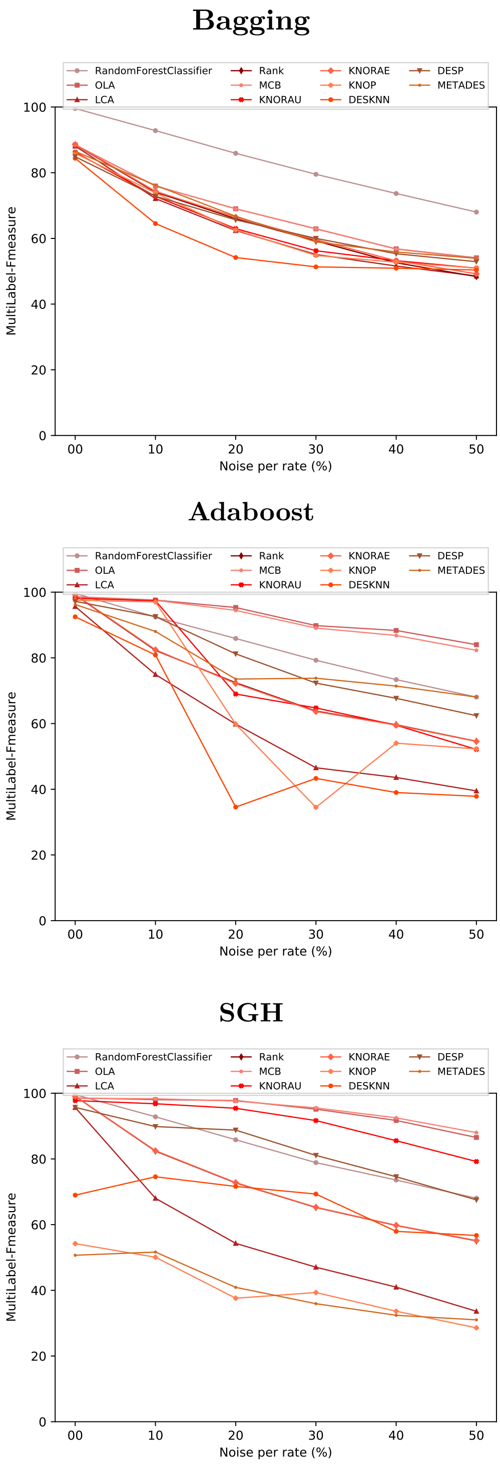


Figure 25 – Multi-Label Fmeasure in a noisy environment to Dynamic Selection techniques in comparison with Random Forest achievements using three Generation methods: Bagging, Adaboost and SGH.

B.0.5 Kyoto 2009 Spring

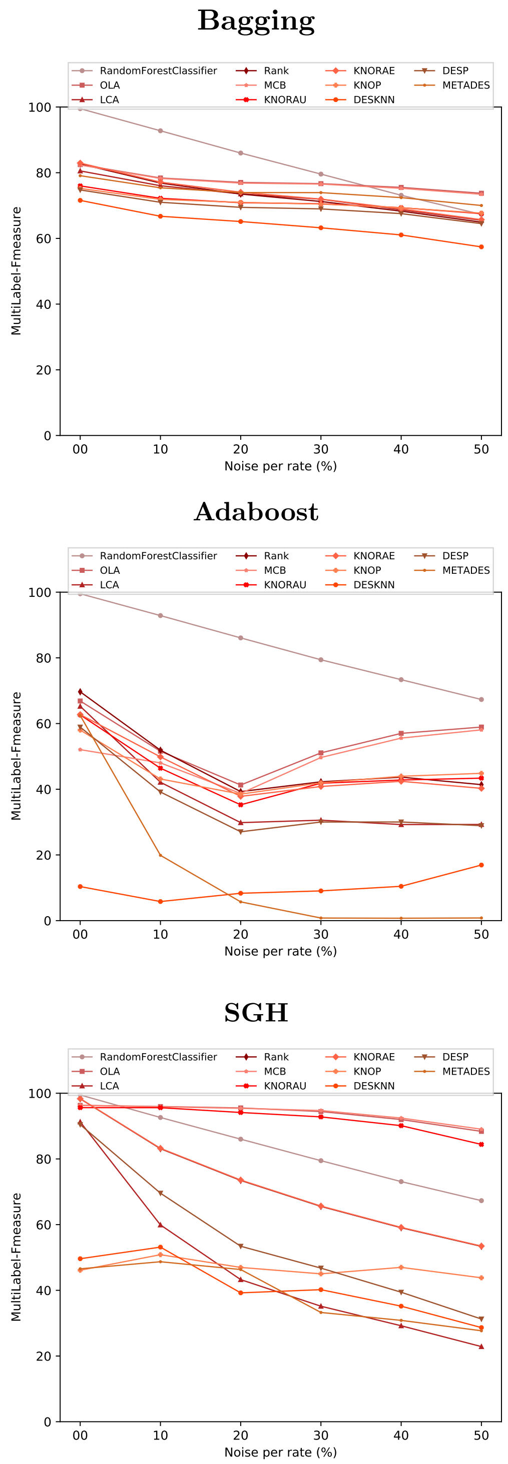


Figure 26 – Multi-Label Fmeasure in a noisy environment to Dynamic Selection techniques in comparison with Random Forest achievements using three Generation methods: Bagging, Adaboost and SGH.

APPENDIX C – F1-SCORE + SGH PERFORMANCE IN NOISY ENVIRONMENTS

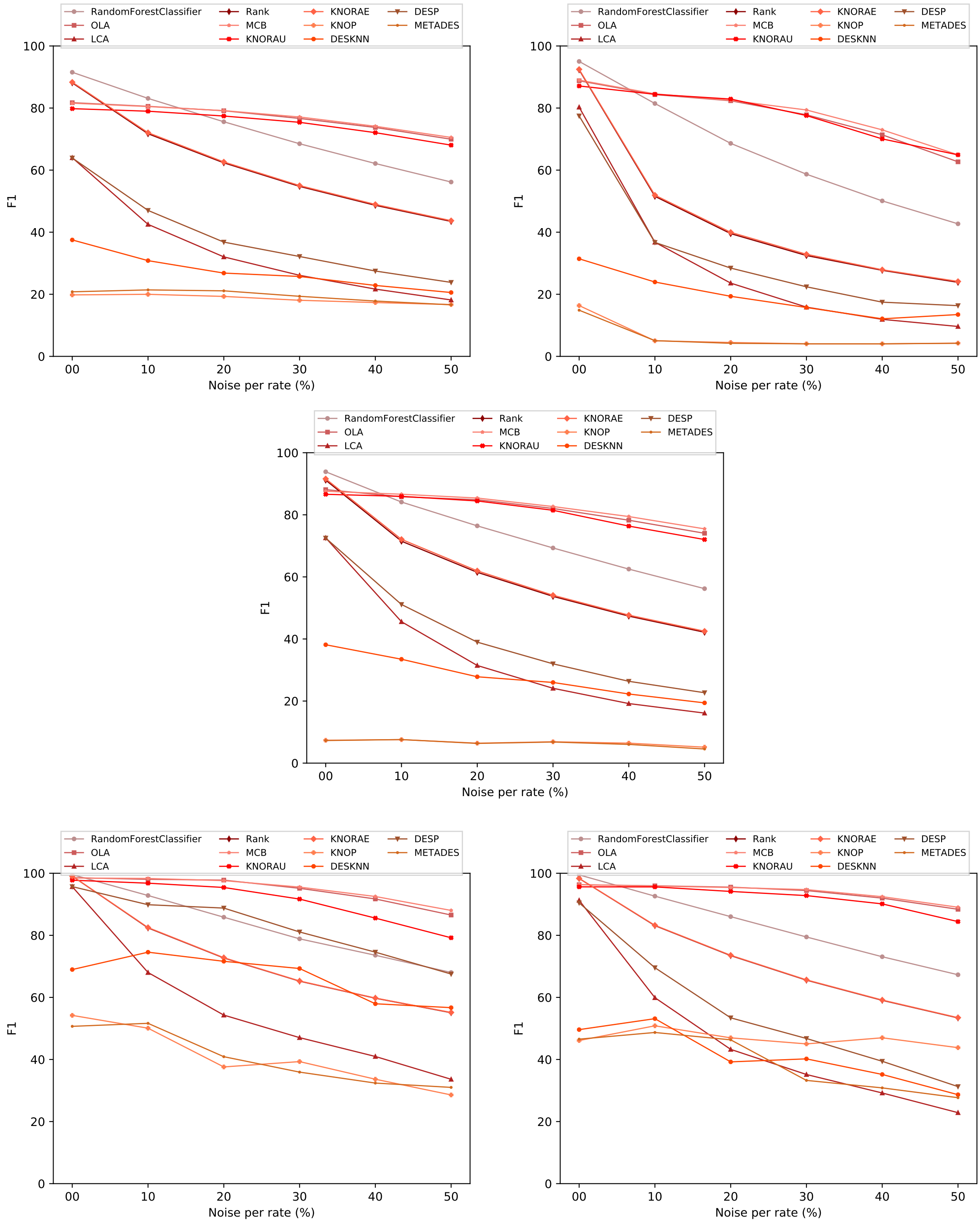


Figure 27 – F1-Score rate. The Perceptrons were chosen by DCS and DES techniques and compared with Random Forest results through noise augmentation. The KNORA-U, OLA, and MCB presented better results even in noises level up.

APPENDIX D – CONFUSION MATRICES TO DYNAMIC SELECTION APPROACHES

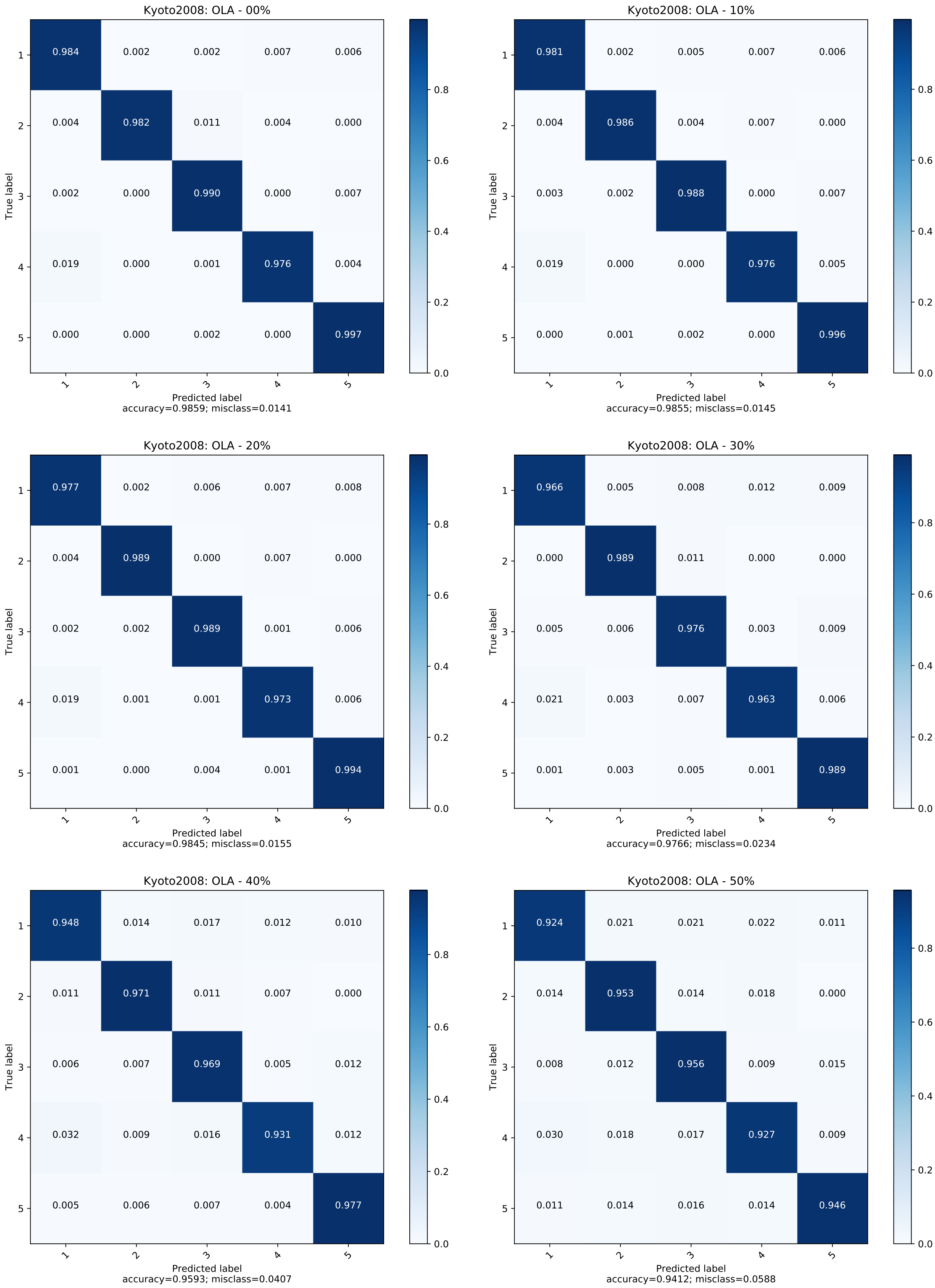


Figure 28 – The confusion matrix evaluated on Kyoto 2008 considering OLA as the dynamic selection technique on labels randomly modified in up to 50% of cases.

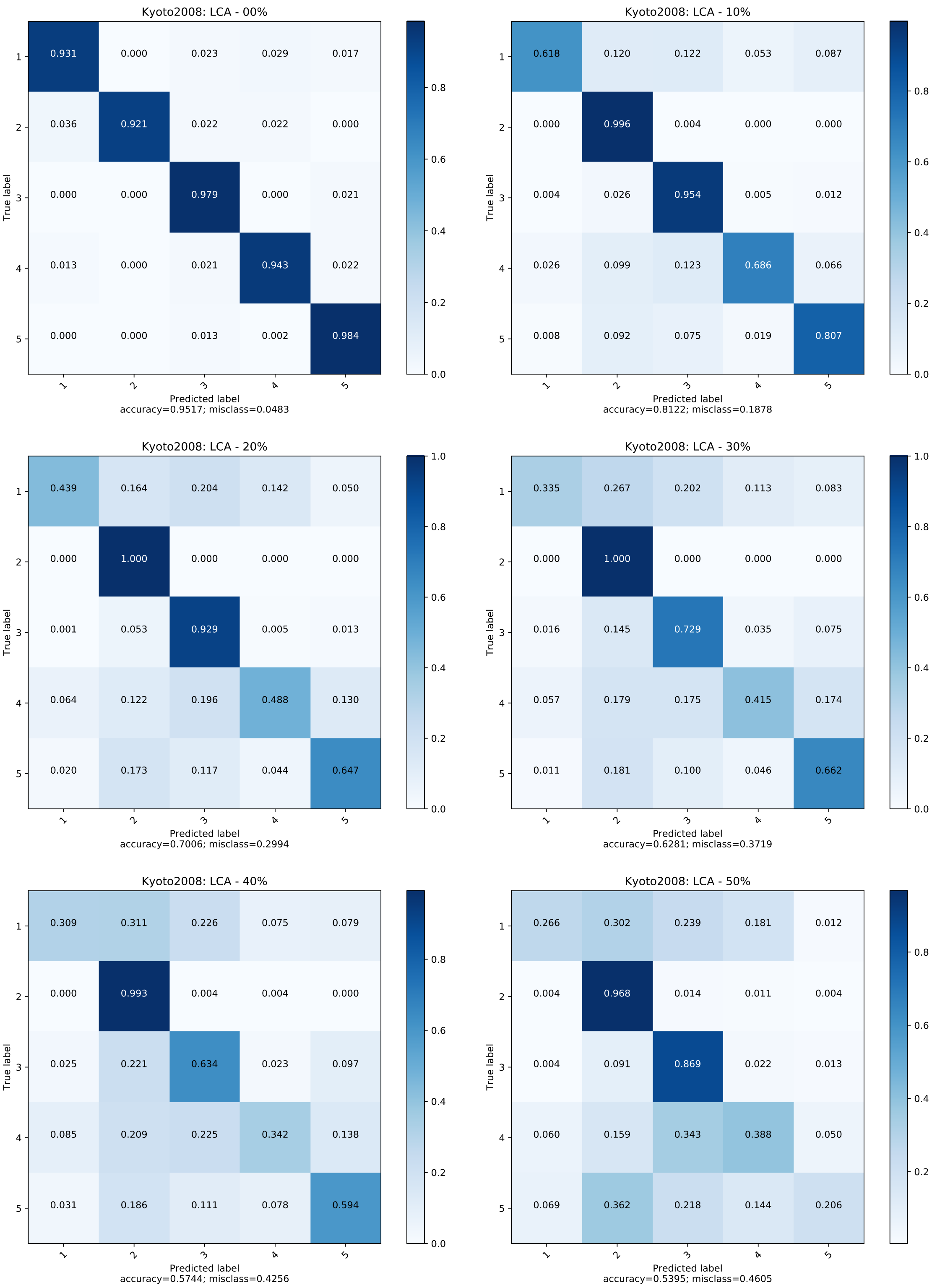


Figure 29 – The confusion matrix evaluated on Kyoto 2008 considering LCA as the dynamic selection technique on labels randomly modified in up to 50% of cases.

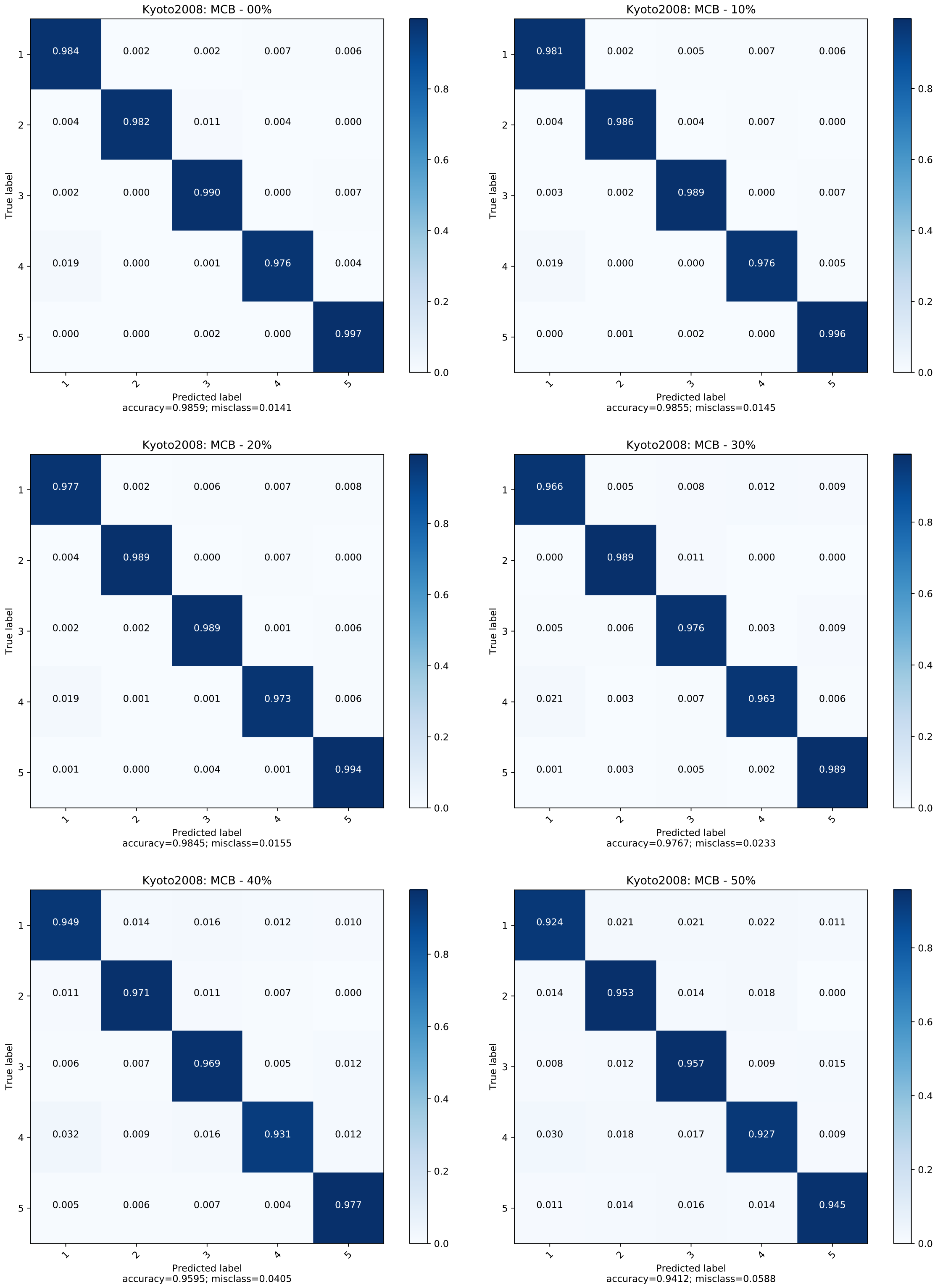


Figure 30 – The confusion matrix evaluated on Kyoto 2008 considering MCB as the dynamic selection technique on labels randomly modified in up to 50% of cases.

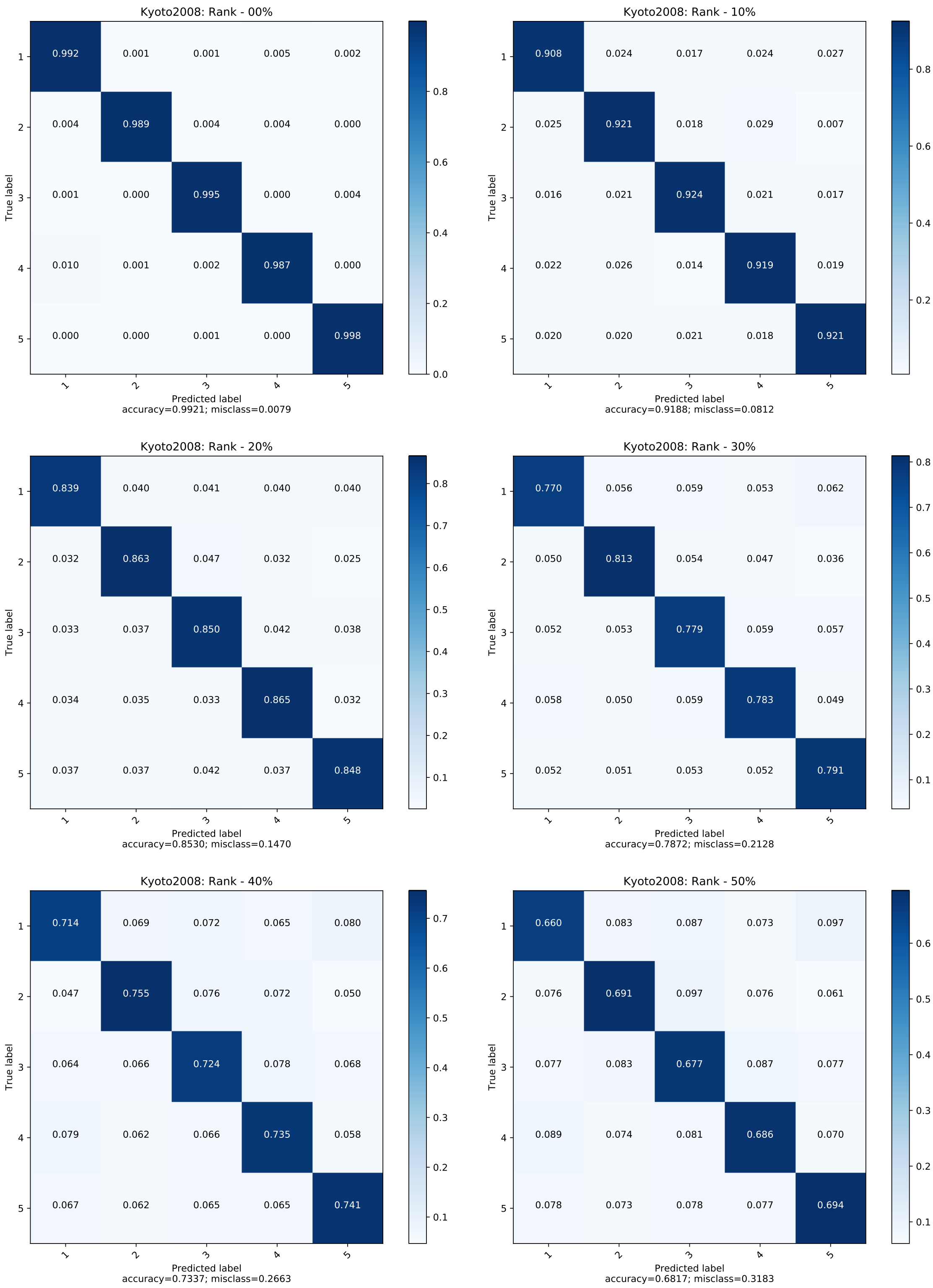


Figure 31 – The confusion matrix evaluated on Kyoto 2008 considering Rank as the dynamic selection technique on labels randomly modified in up to 50% of cases.

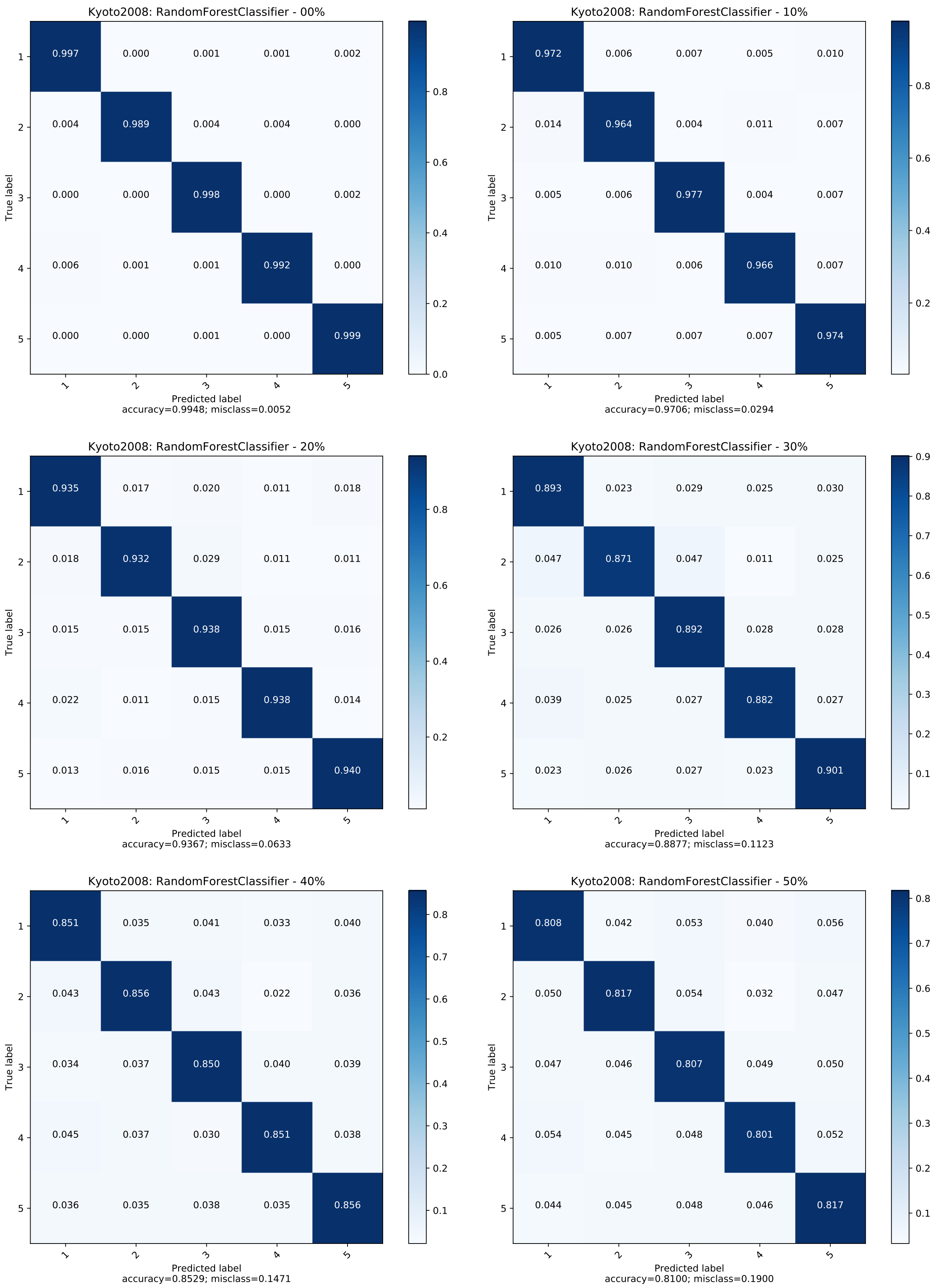


Figure 32 – The confusion matrix evaluated on Kyoto 2008 considering Random Forest as technique on labels randomly modified in up to 50% of cases.

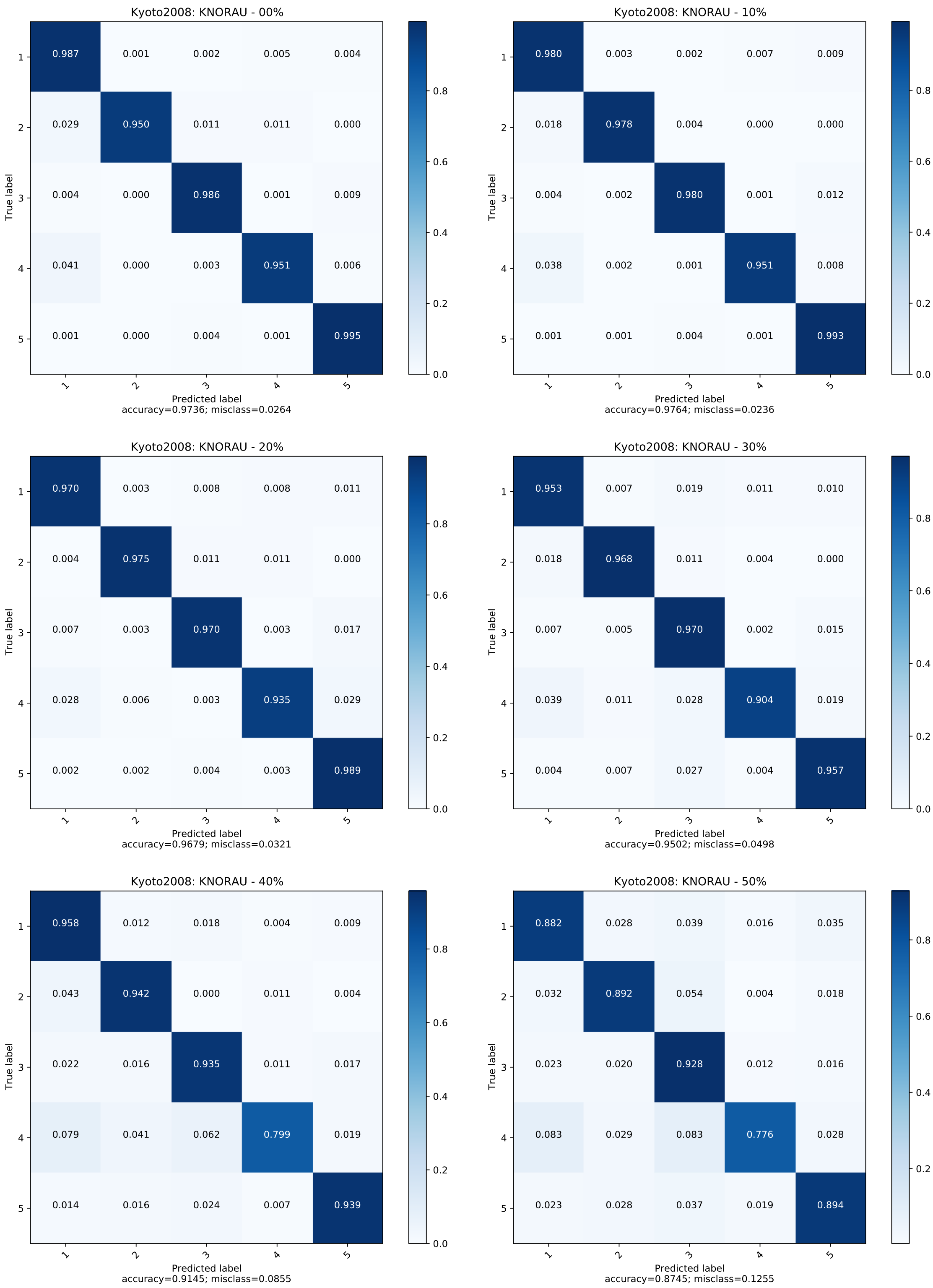


Figure 33 – The confusion matrix evaluated on Kyoto 2008 considering Knora-Union as the dynamic selection technique on labels randomly modified in up to 50% of cases.

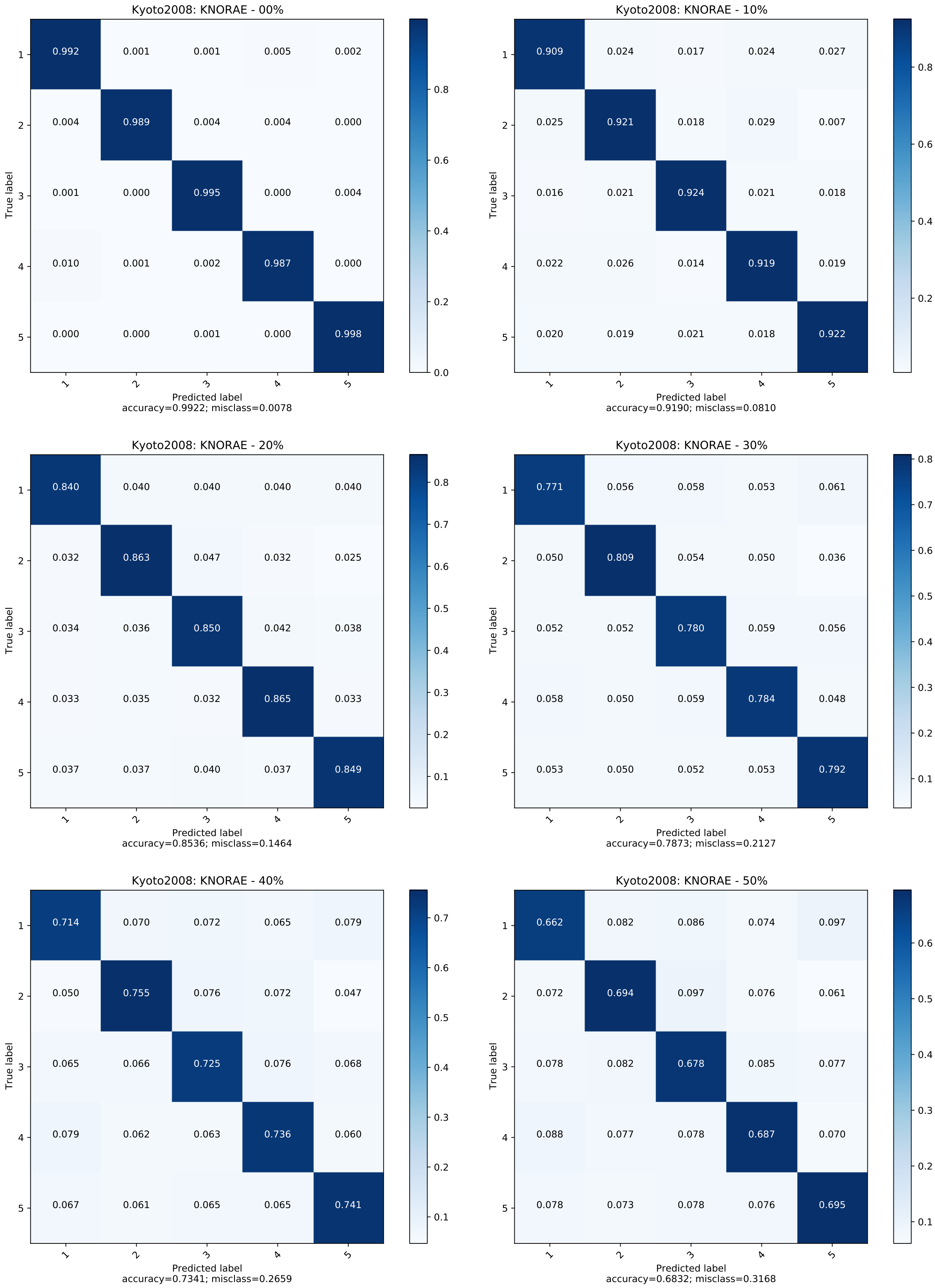


Figure 34 – The confusion matrix evaluated on Kyoto 2008 considering Knora-Elimination as the dynamic selection technique on labels randomly modified in up to 50% of cases.

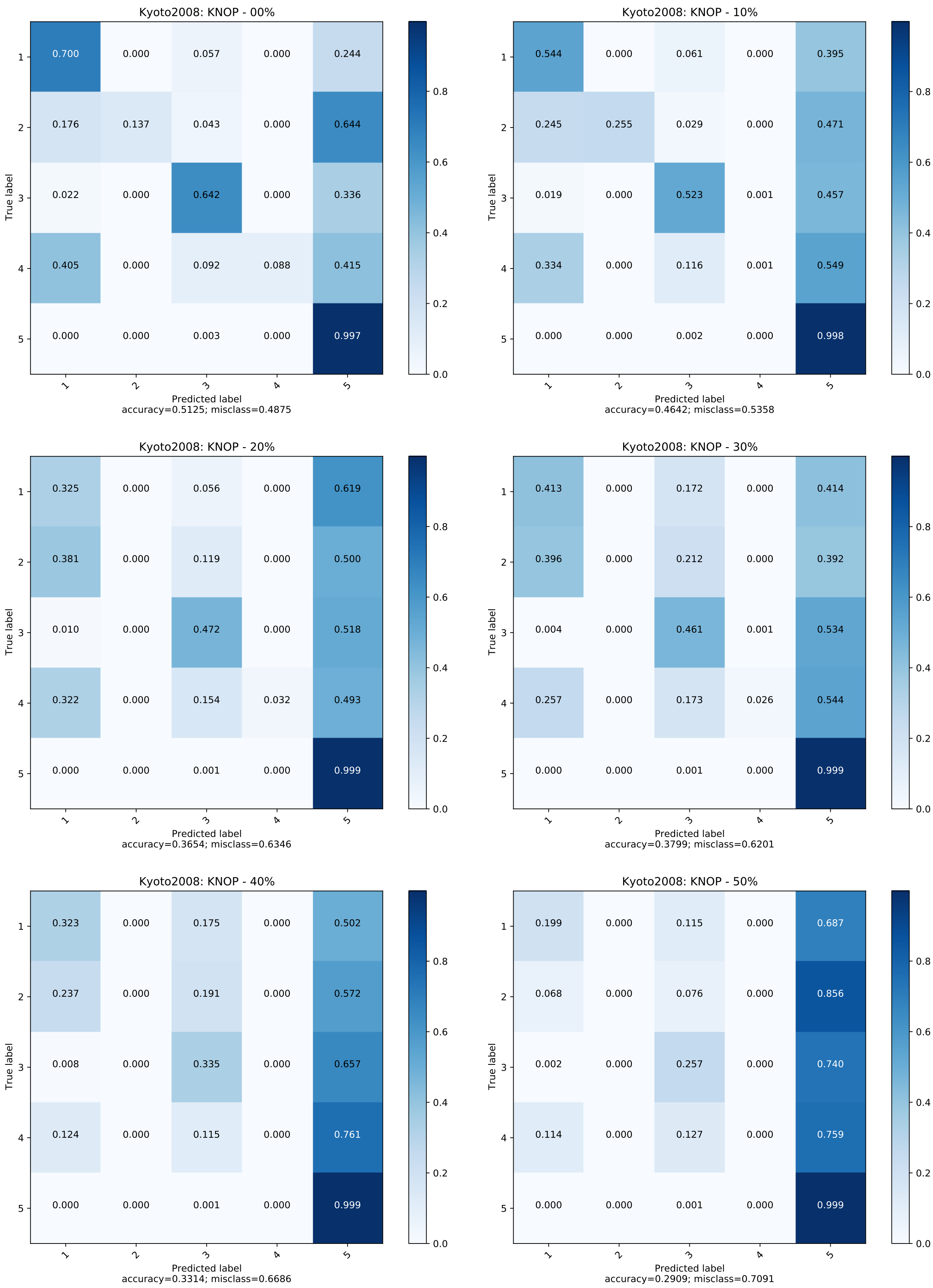


Figure 35 – The confusion matrix evaluated on Kyoto 2008 considering KNOP as the dynamic selection technique on labels randomly modified in up to 50% of cases.

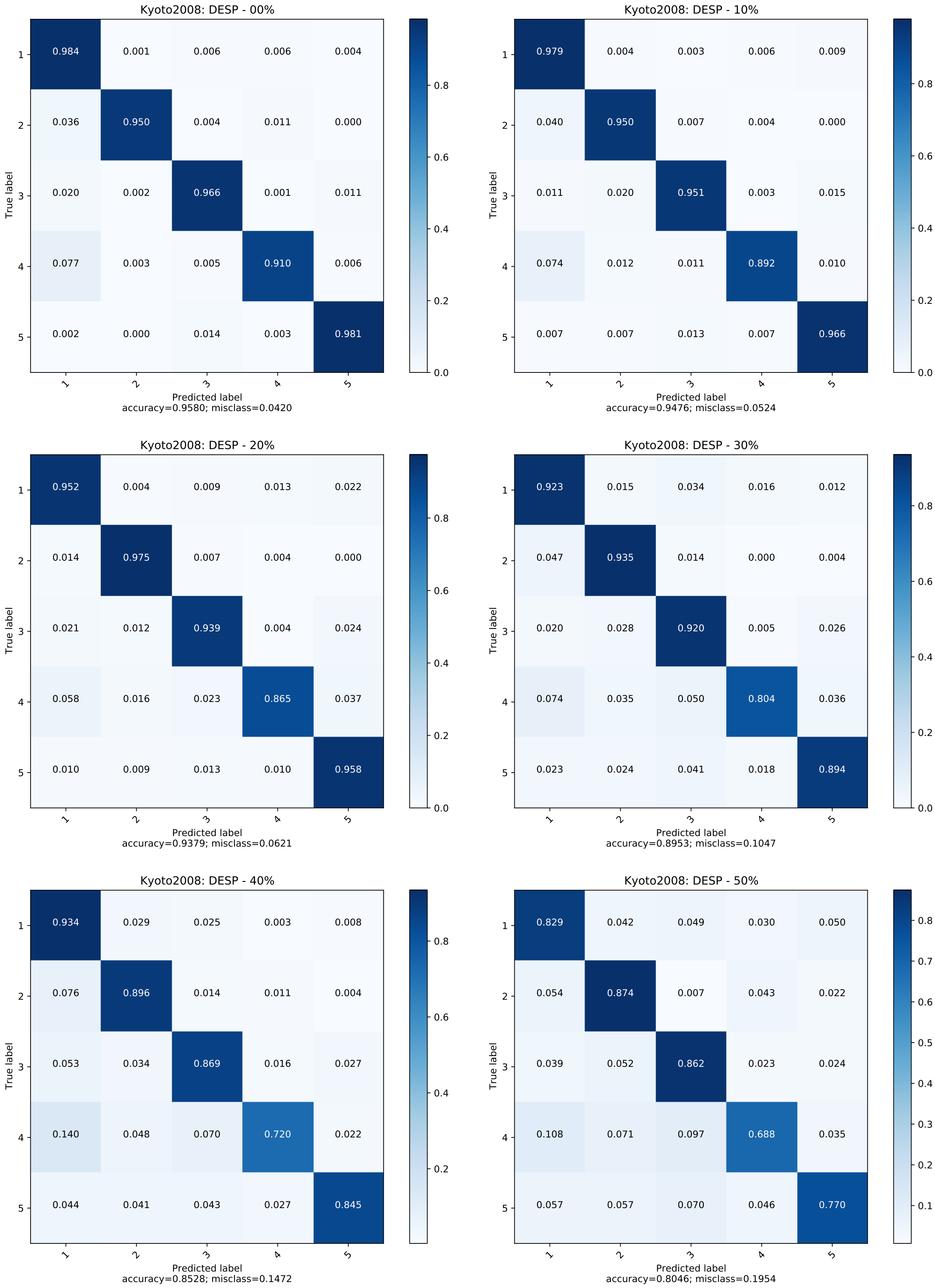


Figure 36 – The confusion matrix evaluated on Kyoto 2008 considering DESP as the dynamic selection technique on labels randomly modified in up to 50% of cases.

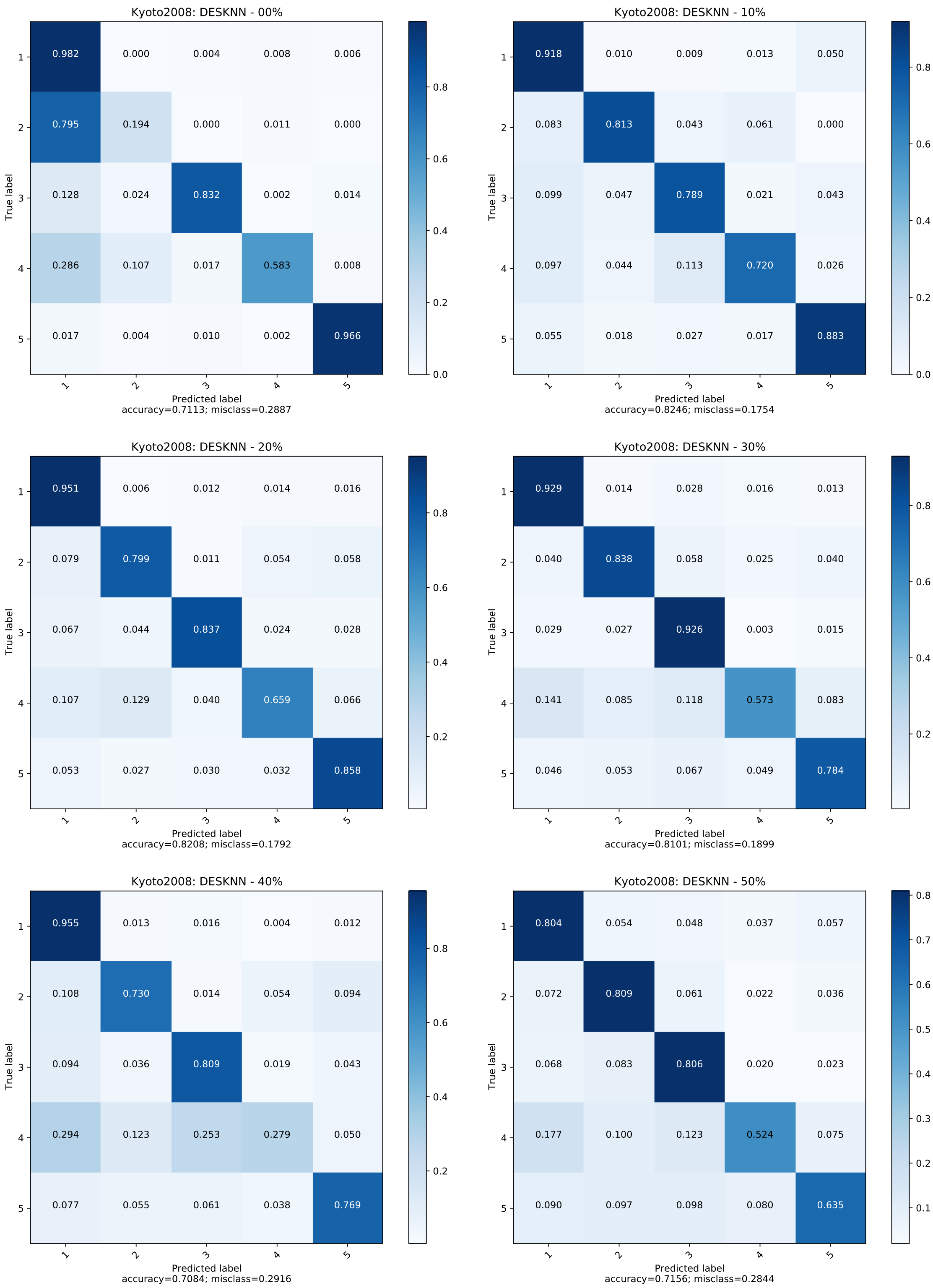


Figure 37 – The confusion matrix evaluated on Kyoto2008 considering DESKNN as the dynamic selection technique on labels randomly modified in up to 50% of cases.