I. A Blackboard based Hybrid Multi-Agent Approach for Machine Learning using Reinforcement Learning Techniques

Vasileios Manousakis Kokorakis

Thesis

Submitted to the University of Brighton for

PhD Thesis

Supervised by: Miltos Petridis ,Stylianos Kapetanakis

Department of Computing, Engineering and Mathematics

March 2018

University of Brighton

II. Contents

III. Acknowledgments

Firstly, I would like to express my sincere gratitude to my supervisors Miltos Petridis and Stelios Kapetanakis for their continuous guidance and support. I am also very thankful to Nikolay Burlutsky for our fruitful conversations and for providing me with his data. I was delighted to share my office and my flat with PhD student Eleftherios Bandis and I want to thank him for his daily support. I appreciate his help during the proofreading of the thesis and his providing some valuable input on the ideas challenged in this thesis. In addition, I would like to give an extra mention to Nikolaos Christakis for his continuous enthusiasm and sincere interest in my PhD work. I could not even imagine a better support during these long, hard, nevertheless exciting days. Finally, I would like to thank my parents and my brother for their endless support and their unconditional belief in me. I also wish to acknowledge and thank my close friends for their constant support and friendly advice. I hope I will be able to compensate for the loss of our time together over all these years.

This work is dedicated to all my failures.

IV. Declarations

Declaration

I declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the original work of the author. The thesis has not been previously submitted to this or any other university for a degree, and does not incorporate any material already submitted for a degree.

Vasileios Manousakis Kokorakis

2018

V. Abstract

This thesis argues that diversity in error rate and heterogeneity in classifiers can successfully improve prediction performance through voting and then reward them for their success or failure. To that end, a central manager called the blackboard, it is capable of learning the area of expertise for each agent and exploit their capabilities to boost the predictive performance. We apply and combine several traditional ML methods and algorithms in a novel way in order to construct a blackboard based multi-agent system. In experimental work we focus on two case studies. The first case study includes datasets where accuracy is the performance metric that needs to be improved. On the other hand, the case study focuses on datasets for which recall, specificity and precision are the metrics boosted.

Firstly, a novel blackboard based approach structure are introduced. An intelligent central manager exploits the knowledge diversity. The implementation and combination of the individual components described below, offer the ability to capture classifiers properties and quantify them using artificial intelligent techniques. Secondly, reinforcement learning techniques were utilized to reward its agent members with a trust value that reflects their competence. Thirdly, a number of computational voting functions are implemented and used in a novel way in order to let the agents express their opinion. Trust reward has an influence in each vote. Fourthly, two optimization methods are applied in order to optimize individual parameters included in the proposed trust function. Finally, three versions of the proposed approach are described and evaluated: naive, which is unaware of any information about the examined problem, then a version that is aware of certain patterns that are extracted during cluster analysis, and finally, a dynamic version that is capable of adapting dynamically.

To sum up, this work has mainly contributed to knowledge by developing an intelligent multi-agent framework to exploit collective experience and improve the global prediction performance. To achieve that, three different version of the proposed approach, and a reinforcement learning based manager were introduced. At the same time, five voting schemes were implemented influenced from ta trust metric. The proposed approach is evaluated on a number of datasets from different domains e.g. bio-medical, Stack Exchange, SMS Spam and Financial.

VI. Abbreviations

AI – Artificial Intelligence

CBR – Case-Based Reasoning

DT – Decision Trees

EM - Ensemble Methods

KS – Knowledge Sources

k-NN - k Nearest Neighbours

LR – Logistic Regression

MAS – Multi-Agent Systems

ML – Machine Learning

PCA - Principal Component Analysis

RF - Random Forest

SGD - Stochastic Gradient Descent

SVM - Support Vector Machines

XGB - Extreme Gradient Boosting

VII. List of Figures

Chapter 1 Introduction

"*I propose to consider a question, can machines think?*"

-Alan Turing(TURING, 1950)

The aim of this doctoral dissertation is to explore the challenge of improving prediction performance of under-performing agents. As a result, a novel approach with autonomic properties aimed at improving prediction performance is proposed. The methodology is tested and evaluated in a number of experiments on real world problems, such as datasets from Stack Exchange, financial, bio-medical and spam domains. In this Chapter, the problem of boosting performance metrics through classifier diversity is introduced. Firstly, the overview of the research is explored in Section 1.1. Secondly, the motivation for this work is presented in Section 1.2. Thirdly, the research questions, the research methodology and contribution to knowledge are discussed in Sections 1.3, 1.4, and 1.5. Fourthly, in Sections 1.6, 1.7 and 1.8., the structure of the thesis and the papers published as a result of this research are presented, as well as the contribution to knowledge.

1.1 Summary

This Chapter introduces the problem of improving prediction performance using an intelligent hybrid multi-agent approach. To begin with, the problem of exploiting heterogeneity from a pool of classifiers in order to improve the overall prediction performance is discussed. Following, the underlying motivation for attempting this research was stated. Proceeding, the research methodology and questions addressed in this thesis along with the main contribution to knowledge are presented. Finally, the organization of this thesis and a list of the peer-reviewed papers published by the author as a result of this research are declared.

1.2 Problem Overview

Recently, researchers, businesses & enterprises have paid huge attention to data science. Every process, either human or mechanic, produces data. If we scale this up to a large business we can securely infer that excessive amount of data is daily produced and stored. As the speed of information growth increases in geometrical progress, problem complexity increases at the same rate (Eric Savitz, 2013). From one point of view, excessive data and high complexity seem to cause great troubles to human experts, because traditional data processing and analysis approaches become obsolete and inefficient. Some of the difficulties and challenges include data capture and storage, data analysis and visualization (Eric Savitz, 2012).

Often collected datasets are so large in volume and complex in terms of high dimensional, unstructured, noisy, erroneous and heterogeneous, that traditional statistical and predictive data processing applications are inadequate to deal with them. Big enterprises utilize statistical tools and machine learning (ML) models to expand their productivity by optimizing or predicting a potential situation about a problem. As a result, the need for robust ML models is more urgent than ever.

ML techniques are capable of performing that, but struggle with big, complex and unstructured data. As the volume of data scales up, ML algorithms need to scale up too in order to cope with high demands. Thus, learning time increases because of the increased complexity of ML algorithms. Thus, the learning process for this kind of data is severely consuming as far as time and resources are concerned. Therefore, two are the main challenges: one is that the conventional ML algorithms perform very poorly in terms of prediction performance, and the other is that the training time increases exponentially and memory resources for handling this kind of tasks are limited. The current state of the art approach in prediction analysis towards increasing prediction performance was the manual selection or combination of techniques that presented better results. Experts need to go through rigorous processes, perform experiments and conclude as to which method suits best to the domain problem they are currently facing.

EM came to scene to deal with the first issue, prediction performance. Ensembles are learning algorithms that are constituted by a set of classifiers and then classify new data points by taking a weighted vote of their individual predictions (Dietterich, 2000). Briefly some of them are: Adaboost, Bagging, Boosting, Random Forests, etc (Polikar, 2009). The main advantage of these methods is that they can increase prediction performance (accuracy, recall, precision) by either reducing variance or bias. However, in many cases fail to produce much better results since the data are unstructured or noisy. Moreover, in case of new incoming data, one must repeat training process which could be a time consuming and tedious process.

Nevertheless, researchers have tried to utilize Multi-Agent System (MAS) in recent years to enhance prediction performance, too (Twardowski & Ryzko, 2014). Multi-agent techniques and architectures are more mature, using more sophisticated communication, multi-threading and agent control and management protocols (Braubach, Pokahr, & Lamersdorf, 2005; Jennings, Sycara, & Wooldridge, 1998; Lars Braubach, 2005, 2012; Shehory, 2001). Utilizing multi-agent advantages and the existing high-performance computing technologies offers the possibility to extend and enhance the abilities of MAS systems in addressing issues of prediction performance, domain complexity, and unstructured data. In references (Teodorescu & Petridis, 2013) and (Pourpanah, Tan, Lim, & Mohamad-Saleh, 2017), the authors examined the issue of prediction performance using multi-agent approaches. Similarly, the problem of prediction performance was approached by applying intelligent Case Based Reasoning (CBR) agents in heterogeneous distributed case bases from the same domain inside a multi-agent environment (Teodorescu & Petridis, 2013). Also, Q-learning and Bayesian formalism were used to tackle with classification tasks utilizing Neural Network Ensembles (NNE) in a multi-agent environment (Pourpanah et al., 2017).

The problem addressed in this thesis is the scenario where robust prediction performance is crucial. We take the standpoint of using multi-agent architectures that provides us with the capability of utilizing hybrid techniques along with methods for calculating trust reward, confidence and competence of an individual agent. More specifically, in order to illustrate this scenario, a variety of examples where improving prediction performance is crucial are presented below. For example, improving classification accuracy and precision of SMS spam model corresponds to

the problem scenario. At the same time, improving sensitivity or specificity for a bio-medical dataset is another part of the examined problem scenario. The purpose of this work is not to examine closely each problem individually, but rather to collaboratively improve prediction performance. In order to impact any of the metrics boosted, a reinforcement learning function is implemented. That reward measurement produces a value that reflects the competence of each agent classifier. Consequently, that value influences the vote value of each individual prediction. Finally, the blackboard, which is responsible for supervising the entire process, results to a final prediction.

1.3 Motivation

Traditional ensembles are the state-of-the-art approach for reducing variance or bias in under-performing models. Ensembles like Random Forests, Bagging, Boosting, etc. already make use of computational voting techniques to aggregate the produced knowledge (Bauer & Kohavi, 1999). Improving predictive force using majority voting enhanced by reinforcement learning has certain advantages and a number of possible applications:

- *Reduce uncertainty in multi-agent environments.* Multi-agent environments present high uncertainty as far as the hosted agents are concerned (Huynh, Jennings, & Shadbolt, 2006). Monitoring the performance of the agents in a multi-agent environment dealing with classification tasks is a challenging task. Live monitoring and transparency are essential (Ramchurn, Huynh, & Jennings, 2004).
- *Variance vs Bias.* The goal of any supervised ML algorithm is to prevent overfitting or underfitting. Low variance and bias is the ideal state in which the model achieves good prediction performance where the algorithm does not miss relevant relations or is either sensitive to small fluctuations (Bart Jacob, 2004).
- *Enhance Predictive force.* Common practice amongst ML experts, when they attempt to solve a problem using supervised algorithms, is to try many models using different train-test splits and choose the one that performs the best. That process is time consuming and challenging since there are lots of

little things to consider. Taking off the equation a fraction of these things and enhance predictive performance using a number of heterogeneous classifiers is another potential application.

1.4 Problem Specifications

1.4.1 Trust Measurement

A function based on reinforcement learning is a great tool for capturing and exploiting the capabilities offered by the experts. According to literature, reinforcement learning is a suitable approach for agents to explore and exploit unknown and dynamic environments (Leslie Pack Kaelbling, 1996). Researchers such as (Pourpanah et al., 2017; Teodorescu & Petridis, 2013) examined trust in MAS to deal with classification tasks in ML. The latter, although capable of identifying the most competent entity inside the environment do not improve the overall prediction performance. In this research work, blackboard supervisor has the ability to develop trust during the process and evaluate the quality of the individual predictions. Also, since reinforcement learning techniques are applied during training and runtime, the entire system is capable to adapt dynamically when the structure of the problem changes.

1.4.2 Voting

Knowledge aggregation from a pool of diverse classifiers is a major issue in this thesis. Combination of the produced knowledge from heterogeneous experts can be challenging. Computational voting can facilitate the aggregation of different opinion and provide the freedom to all the agents to equally express their opinion. Indeed, voting can help the blackboard to conclude to a final answer by applying two kind of voting methods, SCF and SWF functions (Procaccia, 2008). The first declares one winner which the black will take into consideration. The second aggregates the opinions of every single agent. Even though there has been long lasting research into computational voting in multi-agent environments (Xia, 2011), there is not yet work in the literature yet that has combined all these together to construct an ensemble approach based on multi-agent principles.

1.5 Efficient Set-up for Improving Prediction Performance

1.5.1 Naïve

The first experiment outline describes the performance of the proposed system without considering the potential knowledge areas (clusters) extracted from the dataset. The goal of this setup is to identify if CHIMACS intelligence can capture the underlying trends in the data and recognize the individual capabilities of the experts, without stating to the system explicitly the number of the distinct knowledge areas. The performance of the proposed system is compared to its agent classifiers performance.

1.5.2 Knowledge Area Aware

In the second experiment, blackboard is relying on the underlying patterns hiding inside the data. It's intuition regarding the expert competence is not restricted sorely on the expert's classifiers competence. This time blackboard has the knowledge of the distinct knowledge patterns existing in the data and is now capable of capturing the individual performance in a distributed manner. K-means unsupervised clustering algorithm is used to capture patterns in the data. Silhouette performance metric determines the optimal number of clusters. However, in many cases when the data is considered very complex three clusters instead of two are used. That way, the blackboard is given the space to better capture the individual performances of each learning agent. The iterations are set to 400 and the kernel is configured as k-means++ for better centroid assignment.

1.5.3 Dynamic Knowledge Areas Extraction

The modified kernel of the third experiment determines dynamically the essential knowledge patterns without employing any unsupervised learning. The previous approach conceals a few drawbacks in regard to code scalability and flexibility of the learning process. Building a new unsupervised model in case of new incoming data causes performance issues. To that end, a more intelligent approach was followed. A

k-NN based approach is now responsible for distinguishing patterns inside data by getting back the n nearest neighbours. The number of nearest neighbours and the similarity algorithm is defined based on the examined problem.

1.6 Research Questions

The aim of this doctoral dissertation is to answer the main question stated as follows: "How c*an a hybrid multi-agent system enhance prediction performance efficiently and promote collaborative reasoning under the supervision of a centralized intelligent supervisor?* The answer to the stated main research question is an attempt to expand further the research of Elena, Irena Teodorescu related to Multiple, Heterogeneous, Case-Based, Reasoning Multi-Agent Systems (Teodorescu & Petridis, 2013).

The first step to address the main research question implies answering five individual sub-questions. Addressing the problem of enhancing predictive force of poorly performed agents leads to the first sub-question:

1. How can a centralized and intelligent multi-agent system capture, learn and exploit the diversity in capabilities of the accommodated agent estimators in order to increase prediction performance using model-free techniques?

The second step addresses the problem of suitability of the selected multi-agent architecture and its capability to accommodate heterogeneous agent classifiers under a centralized supervisor:

2. Does the blackboard architecture address efficiently the requirement of the accommodation of heterogeneous agents under the supervision of a central manager that enables flexibility and scalability?

The third step refers to the related methods developed so as to take advantage of the agent expertise contributing to the overall prediction force improvement. In more detail, it explores the ability of the proposed approach to enhance the predictive power of models with high variance or bias. This step leads to the second and the

third sub-questions addressing the problem of implementing the suitable components to identify diversity and combine the produced knowledge:

3. How can a function capture, combine and evaluate the quality of the produced knowledge?
    3.1. How can reinforcement trust measurement and voting functions can enable collaborative reasoning and contribute in reducing variance and bias in trained models?

The fourth step includes the ability of the system to behave autonomically. This step involves choosing and implementing a function that can improve the effectiveness of the proposed approach and alleviate the expert from the burden of constant supervision. As a result, the following sub-question is addressed:

4. Can autonomic principles and behaviours be embedded in the hybrid system and help improve its effectiveness?

The fifth step is to choose a set-up of the proposed approach. This step involves choosing an efficient set-up to improve predictive force. Three blackboard kernel versions are proposed and implemented. As a result, the following question and sub-question are addressed:

5. Can an intelligent blackboard control the reasoning combination by learning about its environment and can it adapt during the execution process?
    5.1. What kind of information about the examined problem is essential to be available to the central blackboard supervisor?

Finally, the sixth step explores methods for optimizing trust calculation process. Choosing appropriate values for the reinforcement signal estimation and gamma factor is challenging, since the search space is very big. Thus, the last sub-question addresses the problem of how to optimize the process of choosing near optimal values for the above-mentioned parameters during trust calculation process:

6. How can one optimize the process of searching near optimal values for the reinforcement signal and gamma parameters?

1.7 Research Methodology

The research methodology adopted in this research work is divided in six steps. Each step is presented and discussed below.

The first step is finding a suitable architecture flexible and scalable enough to accommodate the agents and the implemented components. Since the proposed approach is designed using a fusion of modules, the resulted framework designed to improve prediction performance needs to be based on an architecture that enables an expert to adapt and scale-up or scale-down the framework according to the current requirements. At the same time, a major requirement is supervision by a central manager. That narrows down the options regarding the selected multi-agent architecture. Hence, the chosen architecture is the blackboard. For the purpose of demonstrating the applicability of the proposed approach seven datasets are used: five from UCI ML Repository (waveform, wine quality, bio-medical, financial), one from Stack Exchange (Q&A) and finally one from Kaggle (SMS spam) repository. As a result, it is proved that it is feasible to improve predictive force using the proposed approach.

The second step includes the implementation of a function, based on reinforcement learning techniques and voting methods which enable knowledge fusion. The function rewards or penalizes each agent classifier based on the evaluation of the provided answer. Then, the voting process takes place which allows all the agents to express their individual opinion about the prediction. In general, five majority voting function have been implemented but only two are affected by trust value. Trust measurement influences the power of the individual votes. Five state-of-the-art ML algorithms and ensembles are used as agents, namely Logistic Regression (LR), Adaboost, Bagging k-Nearest Neighbours (k-NN), Decision Trees (DT), Random Forests (RF), Support Vector Machines (SVM), Stochastic Gradient Descent (SGD) and X-Gradient Boost (XGB). The agents change every time another dataset is examined. As an evaluation metric for hyper-parameter optimization, accuracy, F-measure, and time performance are recorded.

The third step is to construct the kernel the blackboard will be based. The kernel is structured in three conceptually different ways. The first is based on the simple

context which the blackboard does not know anything about the problem and tries to figure out in the process how to improve classification performance. The second knows a priori the number of clusters structured inside the data. A cluster analysis using k-means is performed on the data. The last version, exploits the nearest neighbour's algorithm as a kernel and instead of being static like the clustered version, it uses k-NN algorithm to reveal the underlying patterns from the data. We compare the performance of the proposed approach between the three set-ups and the individual trained agents.

Finally, the problem of optimizing the trust reward calculation is addressed. For that purpose, two methods for optimizing the reinforcement signal and gamma factor parameters are implemented. A Bayesian randomized probability, namely Thomson Sampling algorithm, is used as a function for updating the reinforcement signal r. Agents can be considered as a Multi-Armed Bandit Problem (MAB) thus the blackboard used that algorithm to optimize it. Then, a process called Simulated Annealing is employed to optimize the gamma factor values. Annealing is used in metallurgy, where tempering a metal, glass, or crystal by heating above its melting point and then cooling it very slowly until it solidifies into a perfect crystalline structure, and produces high-quality materials.

1.8 Contribution to Knowledge

The main contribution of this thesis to knowledge is through proposing a novel approach to improve predictive force and enhance under-performing models by exploiting error rate diversity in classifiers. A summary of the main contributions are presented as follows:

1. Proposed a novel approach to improve predictive force using a hybrid, intelligent blackboard based framework.
2. Introduced a novel approach of employing voting functions that works as a bargaining system between the blackboard and the ML agents. The agents act as an electorate and use five proposed majority voting functions. Majority SCF nominate one agent's answer as a winner and SWF one alternative over another.
3. Introduced a novel way of using reinforcement trust reward. The value calculated as a function of each agent performance influences the individual vote power. Trust measurement reflects the competence of the agent classifiers in accordance with the certain patterns revealed from the examined data.
4. Improved predictive force in terms of accuracy, recall, precision and specificity. Showed that by changing the evaluation of individual predictions provided by the agents, the blackboard is capable of improving recall, specificity and precision or accuracy only.
5. Proposed a novel application of Thomson Sampling algorithm for optimizing reinforcement signal reward of the trust function. At the same time, simulated annealing was employed to find near optimal weights for gamma factor parameter. Showed that the blackboard empirically converged faster.
6. Implemented an intelligent kernel for the blackboard. Proposed a novel approach of revealing and exploiting data patterns using the k-NN algorithm as the backbone of the blackboard supervisor. It provides blackboard the means to adapt by updating the knowledge in every epoch.

1.9 Structure of the Thesis

This thesis is structured as follows:

Chapter 1 introduces the problem overview addressed in this thesis, the underlying motivation, research questions, research methodology, contribution to knowledge, structure of this thesis, and research papers published.

Chapter 2 presents Literature Review and Related Work.

Chapter 3 introduces and describes the components of a hybrid and intelligent blackboard-based multi-agent approach to improve prediction force. Two methods for optimizing trust measurement are described.

Chapter 4 explores the problem of choosing the optimal set-up for boosting prediction performance. It also describes the design and functionality of three conceptually different kernels configurations.

Chapter 5 introduces the experimental design used to evaluate the effectiveness of the proposed approach. Results, visualizations and an analysis are provided for every examined dataset.

Chapter 6 summarises the main findings and contributions of this research work, revisits the research questions and, finally, proposes a few future steps for further research.

1.10 Publications

The following papers resulted from this research work:

1. **Vasileios Manousakis Kokorakis, Stelios Kapetanakis, Miltos Petridis**, ***A Multi-Agent architecture for collaborative reasoning in workflows*****, 19th UK CBR Workshop on Case-Based Reasoning, Cambridge, 2014.** The content of this paper describes the motivation for this research work and the results from the experimentation, proving the gap in knowledge.
2. **Vasileios Manousakis Kokorakis, Miltos Petridis and Stelios Kapetanakis,** ***A Blackboard Based Hybrid Multi-Agent System for Improving Classification Accuracy Using Reinforcement Learning Techniques***, **SGAI 2017: Artificial Intelligence XXXIV pp 47-57, LNCS, volume 10630.** The content of the specific paper describes the proposed CHIMACS and C-CHIMACS framework introduced in Chapter 3 and the respected results from chapter 6.

In the next Chapter, Literature Review and Related work are discussed.

Chapter 2 Literature Review

This Chapter leads the reader through a review in the most recent research findings in the multi-agent systems, reinforcement learning and computational voting theories. Firstly, an overview of machine learning and complex decision theory is introduced in Sections 2.1 and 2.2. Secondly, multi-agent systems and architectures are listed in Section 2.3. Thirdly, reinforcement learning, and trust concepts are introduced and described in 2.4. In section 2.5, a review on autonomic attributes is presented. Then, social choice theory and computational voting techniques are discussed in Section 2.6. Finally, the Chapter is summarised in Section 2.7.

2.1 Decision-Making in Complex Real-World Domains

A common case in many real-world problems is the presence of high dimensionality, uncertainty and approximate knowledge where due to these, traditional software systems often cannot cope with the intricacy presented.

Various fields are considered critical domains where incorrect management decisions may have disastrous consequences in many aspects. The complexity of the problems requires the development and application of distributed and intelligent tools capable of providing helpful insights during the decision-making processes. The complexity of real-world systems is characterized by (H. Chen, Chiang, & Storey, 2012):

- The inherent complexity of the systems. Depending on the application domain, processes usually involve a huge amount of knowledge containing complex interactions and dependencies amongst them e.g. social and economic processes. Furthermore, they are stochastic, spatial or temporal dependent processes.

- Uncertainty or approximate knowledge. Refers to the unstructured data during the knowledge extraction processes. Socio-ecological systems or self-organization processes are examples which present chaotic behavior. Sometimes uncertainty can be tamed with additional data or thorough analysis. However, most of the times uncertainty is insurmountable considering the limitations of human capabilities.
- A Huge quantity of data/information (Big Data). Production of great volume of data and information is a common characteristic of these domains.
- Mathematical theory or a deterministic model cannot be extracted from many of the facts and principles underlying the domain.
- Heterogeneity, provenance, and scale. Refers to data heterogeneity due to provenance. Media and the processes that take place while gathering the data are not homogeneous and cannot easily be characterized by measurable parameters and that results in data with different structures e.g. scale multiplicity, which includes distinct spatial scales (i.e., local, national, global). Moreover, the different inherent scale is also an important part of the complexity.

According to (Funtowicz & Ravetz, 1995), not all mechanical, physical or software systems present the same level of complexity. Both the degree of uncertainty and the risk associated with decisions depends on factors like provenance and application domain. Also, (Funtowicz & Ravetz, 1995) distinguished three levels of complexity based on uncertainty and magnitude or importance of the decision.

- Low uncertainty systems. Simple systems correspond to the first level of complexity, where simple models describe a single perspective and would suffice to provide a satisfactory description of the system.
- High uncertainty systems. A Higher degree of uncertainty in such systems requires more than a simple model in order to provide satisfactory descriptions. Gained experience during the decision process and involving individual experts in problem-solving becomes essential.
- Truly complex systems. In this level of complexity, uncertainty is associated with a high number of elements, the inherent relationships amongst them and the conflicting goals. Many systems like environmental and medical systems

belonging to the second and third level cannot only be tackled with the traditional tools and the approaches of mathematical modeling (Sànchez-Marrè et al., 2008). To confront this complexity, a new paradigm is needed and it requires new intellectual challenges.

When presenting the complexity levels and their respective issues it becomes obvious that a flexible and dynamic approach to solving problems more efficiently is essential. In our case, the examined problem domains fall into the first level of complexity.

2.2 Machine Learning

ML or automated learning is a field of research where computers learn from available input (data) without explicitly programming specific instructions. ML approaches, namely ensemble learning, have shown very good results in enhancing predictive performance (Drucker, Cortes, Jackel, LeCun, & Vapnik, 1994). Ensemble learning is a ML sub-field in which trained classifiers are combined in order to provide a collaborative solution to a given problem (Zhou & Tang, 2003; Zhou, Wu, & Tang, 2002). An ensemble is a fusion of trained heterogeneous or homogeneous base classifiers, whose decision boundaries differentiate and as a result they produce different error rates (Dietterich, 2000). The main goal of this concept is the integration of a set of diverse estimators so as to produce higher and more reliable estimates than what can be achieved through a single model (Kuncheva & Whitaker, 2003; Wang, 2010). Figure 1, depicts a visualization of high-level concept of ensemble workflow.

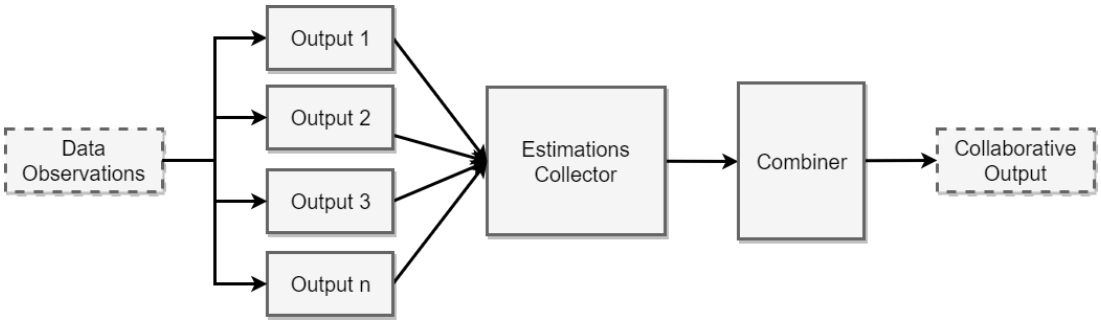


Figure 1 - General Ensemble Workflow Concept

In this thesis, all problems are formulated as a binary classification task. Hence, the aim is to predict the outcome Y for unseen 'test' observations of any examined problem. For this purpose, many predictive models were built employing a function with adjustable hyper-parameters. The training examples X are used during the training process to select the optimal combination of hyper-parameters.

The first step in the process of building an estimator is to divide the dataset into two sub-sets, training and test dataset. The training dataset is used for training the classifier whereas the test dataset is used to evaluate the actual performance of the classifier. The initial dataset can be split in different proportions, for example, 70% of the data becomes the training dataset and the other 30% is used as test bed. Then supervised and unsupervised learning algorithms were employed.

2.2.1 Machine Learning Algorithms for Supervised Learning

In a standard supervised learning, observations of the form $\{(x_1, y_1), \ldots, (x_n, y_m)\}$ Y are used in the training process to approximate an unknown function $y = f(x)$ (Polikar, 2009). A binary classification task assumes that the depended feature has only two classes Y = {0, 1}.

According to "*No Free Lunch Theorem*" (Gómez & Rojas, 2015), *"there is no algorithm that is always the most accurate in all situations"*. Ensemble learning is the process by which multiple classifiers models are generated and combined using different methods to solve a particular computational ML problem. EM are primarily used to improve the performance of a model (classification, prediction, etc.), or reduce the likelihood of an unfortunate selection of a poor one. Unlike ensemble in

statistical analysis, which is usually infinite, ensemble in ML refers only to a finite set of alternative models but typically allows for the much more flexible structure to exist among those alternatives (Dietterich, 2000).

Given $c_1, \dots, c_M$ as the individual classifiers, an ensemble is considered as a set whose individual decisions are combined in some to classify new examples. In (Dietterich, 2000) the main discovery is that more reliable and accurate results are often produced from ensembles than individual estimators. Two are the most important conditions for an ensemble structure to be more reliable and accurate compared to its individual estimators. Firstly, the individual members need to yield error rates less that 0.5, which is at least better than random guessing. Secondly, it is the diversity in error rates on new data observations. We consider n number of classifiers $\{h_1, .., h_n\}$, in the case of correlated errors (not diverse) all the estimators will be wrong, where on the other hand diversity in errors is more probable to offer a correct answer through majority voting. The most common reason to employ EM is the variance-bias trade-off problem. In case, the trained model is too simple and has very few parameters then during training, it may have high bias and low variance. That means it underperforms both for training and cross-validation datasets. On the other hand, in case of a large number of parameters that does not reflect the complexity of the problem, it is may observed high variance and low bias. In that case, the performance is very high in regards to the training set but very low in the test set. Thus, it is important to find the near-optimal balance without overfitting or underfitting the data. This trade-off in complexity reflects the trade-off between bias and variance (Alpaydin, 2010).

According to (Dietterich, 2000) another reason for using ensemble systems is in the case of too much data. A large training dataset can be partitioned into smaller subsets and we can then use each subset to train a separate classifier. The derived models can be combined using an appropriate combination rule. On the other hand, bootstrap samples are created to train different classifiers, where each bootstrap sample is a random sample of the data drawn with replacement (Beran, 1992).

High data complexity is related to data that is too difficult for certain type of classifiers. Decision boundaries are too complex or lie outside the space of functions that can be implemented by a chosen classifier, Figure 2.

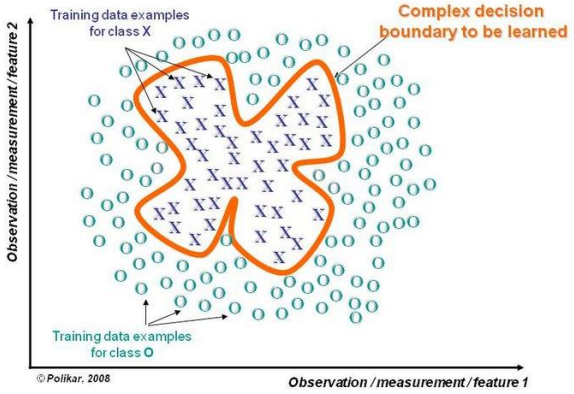


Figure 2 – A non-linear complex decision boundary (Polikar, 2009).

Information fusion relates to data with heterogeneous features that is obtained from diverse data sources, but for the same domain. By providing complementary information, this data can lead to improved accuracy of the classification decision compared to a decision based on any of the individual data sources alone. All these features are not likely to be used together to train a single classifier. In such cases, using the ensemble approach can be extremely helpful, where a separate classifier gets trained using independent features sets but combine their results into more confident decisions (Kuncheva & Whitaker, 2003).

2.2.1.1 Ensemble Methods

In our daily lives, we typically ask the opinions of several experts before agreeing to proceed with a procedure, e.g. take multiple doctors' opinion, we read user reviews before purchasing an item, a scientific paper is reviewed by several reviewers before it is accepted, etc. (Wibral, Lizier, & Priesemann, 2015). In general, routinely in our daily lives before reaching to a decision, we use such an approach. Thus, a final decision is made by combining the individual decisions of several experts and categorizing them by their respective quality value. The primary goal of this procedure is to minimize the selection of actions that are less valuable to us during a decision process (Dasarathy & Sheela, 1979; Dietterich, 2000).

Model selection is the primary reason why EM are used in practice. Model selection is divided into the questions stated below: i) What type of classifier should be

chosen among many competing models? ii) Given a specific ML algorithm, which realization of this algorithm should be chosen? - for example (Rogova, 1994), different initializations in parameters of MLPs can give different decision boundaries.

In order for this process to be effective, within the classification context, the individual algorithms must exhibit some level of diversity among themselves. This is usually achieved by using different training parameters for each classifier that allows individual classifiers to generate different models and thus different decision boundaries. In that case, a different error is made by each classifier and the combination of these can then reduce the total error. This concept is graphically illustrated in Figure 3, where each classifier received different training subsets of the available training data. The models produced different errors (instances with dark borders), but the combination of the three classifiers provides the best decision boundary.

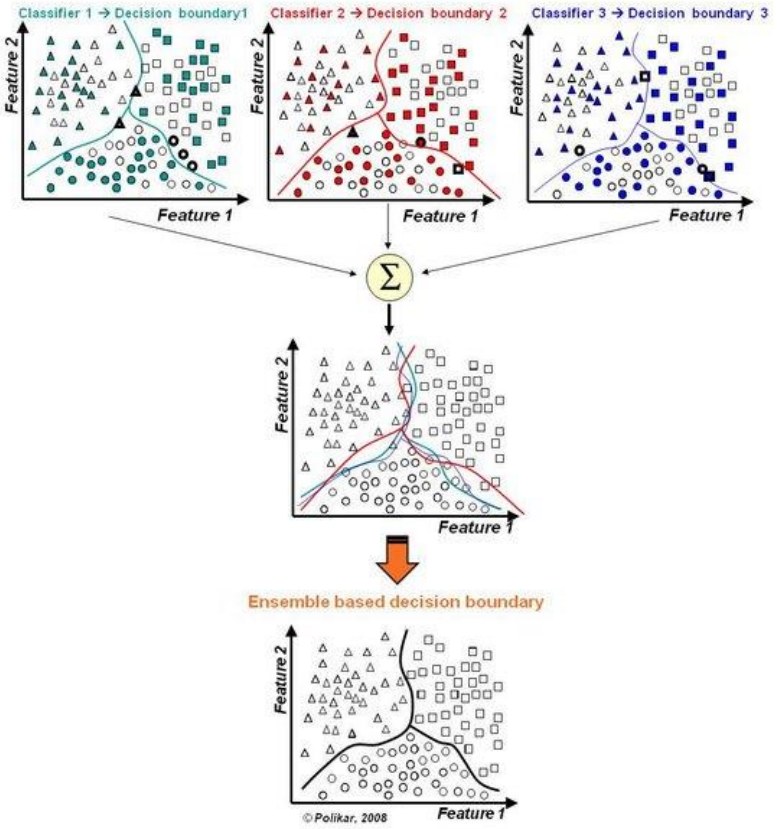


Figure 3 - Combining Classifiers for Reducing Bias (Polikar, 2009)

However, overfitting or even sometimes under-performing compared to an individual classifier are some of the drawbacks that EM hide. Thus, it is important to stress out that the combination of multiple classifiers does not always guarantee better performance. Nor an improvement on the ensemble's average performance of different realizations belonging to the same algorithm can be guaranteed except for some special cases (Fumera & Roli, 2005). Hence, combining classifiers may not necessarily beat the performance of the best classifier in the ensemble, but it certainly reduces the overall risk of making a particularly poor selection. For example, experiments performed in this thesis using EM showed the same or slightly better results compared to each classifier independently.

2.2.1.1.1 Ensemble Algorithms

In the literature, a plethora of terms such as fusion, combination, aggregation, the committee instead of ensembles has been used to indicate sets of learning ML algorithms that collaborate together to solve a ML problem (Jain, Duin, & Mao,

2000; Kittler, Hatef, Duin, & Matas, 1998; Kuncheva, 2004). Nowadays EM represent one of the main current research interests in ML field(Dietterich, 2000; Drucker et al., 1994). A general taxonomy is presented in this section in order to describe the differences between the EM stated in the literature review.

2.2.1.1.2 Voting and Averaging Based Ensemble Methods

Voting and averaging are two of the easiest methods to understand and implement. On the one hand, voting is used for classification problems and on the other hand, averaging is used for regression. In both methods, the first step is to create multiple base classification/regression models using the same algorithm with different splits of the same training dataset, or using the same dataset with different ML algorithms. Methodologies that include algorithms using voting and averaging methods are Boosting and Bagging.

Bagging, which stands for bootstrap aggregating, is one of the earliest and perhaps simplest ensemble based algorithms, which according to (Breiman, 1996) performs very good. Replicas of the training data ensures the diversity of classifiers in bagging is obtained by using bootstrapped. A different version of the same classifier receives a training data subset that is randomly drawn with replacement from the training dataset. Individual classifiers are then combined by taking into account the results of a simple majority vote procedure of their decisions. For any given instance, the answer chosen by the most classifiers is the ensemble final decision (Hall et al., 2009). The pseudo code of Bagging is provided in Algorithm (1).

Input: Data set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
Base learning algorithm $\mathcal{L}$;
Number of learning rounds T.

Process:
for $t = 1, \cdots, T$:
$\mathcal{D}_t = Bootstrap(\mathcal{D})$; % Generate a bootstrap sample from $\mathcal{D}$
$h_t = \mathcal{L}(\mathcal{D}_t)$ % Train a base learner h_t from the bootstrap sample
end.

Output: $H(\boldsymbol{x}) = \text{argmax}_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(y = h_t(\boldsymbol{x}))$ % the value of $\mathbb{I}(a)$ is 1 if a is $true$ and 0 otherwise

Algorithm 1 - Bagging Pseudo code (Hall et al., 2009)

Boosting also creates an ensemble of classifiers by resampling the data, which are then combined by majority voting (for classification) and averaging (for numeric

prediction). However, in boosting, resampling process provides the most informative training data for each consecutive classifier. In bagging individual models are built separately whereas in Boosting is an iterative process. AdaBoost (Freund & Schapire, 1997), is considered as an example and is one of the most famous algorithms and the pseudo code is provided in algorithm (2). Another limitation of boosting is that it applies only to binary classification problems (Hall et al., 2009).

Input: Data set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
Base learning algorithm $\mathcal{L}$;
Number of learning rounds T.

Process:
$D_1(i) = 1/m.$ % Initialize the weight distribution
for $t = 1, \cdots, T$:
$h_t = \mathcal{L}(\mathcal{D}, D_t);$ % Train a base learner h_t from $\mathcal{D}$ using distribution D_t
$\epsilon_t = \Pr_{i \sim D_i}[h_t(\boldsymbol{x}_i \neq y_i)];$ % Measure the error of h_t
$\alpha_t = \frac{1}{2}\ln\ \frac{1-\epsilon_t}{\epsilon_t}$; % Determine the weight of h_t

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \exp(-\alpha_t) & \text{if } h_t(\boldsymbol{x}_i) = y_i \\ \exp(\alpha_t) & \text{if } h_t(\boldsymbol{x}_i) \neq y_i \end{cases}$$

$$= \frac{D_t(i)\exp(-\alpha_t y_i h_t(\boldsymbol{x}_i))}{Z_t}$$

% Update the distribution, where Z_t is a normalization
% factor which enables D_{t+1} to be a distribution

end.

Output: $H(\boldsymbol{x}) = \text{sign}\,(f\,(\boldsymbol{x})) = \text{sign}\ \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})$

Algorithm 2 - Pseudo code for Boosting(Hall et al., 2009)

2.2.1.2 Traditional Machine Learning Algorithms

Since we do not focus on a specific domain, the choice of the ML algorithms is not influenced by any prior research from literature review. The current evidence presented from testing was considered in the selection process. Certain ML algorithms like LR, prove to be very effective and useful in solving predictive tasks due to scalability and transparency of results interpretation. At the same time, other more computationally demanding algorithms such as SVM, ANN and RF can show higher accuracy in prediction but the computational requirements are much more expensive. Also, interpretability of the prediction results is a major issue in these approaches. Nine diverse ML algorithms, namely LR, NB, RF, XGB, AdaBoost, Bagging, SVM, DT and k-NN algorithms are described in the below table, in more

detail. All the above-mentioned algorithms were not used for experimentation simultaneously, as performance and current proof of concept are the selection requirements.

Table 1 - ML Algorithms

Abbr.	**Full name**	**Description**	**Reference**
LR	Linear Regression	A probabilistic binary statistical classification model; can use linear or non-linear kernel.	(Wilson & Lorenz, 2015)
k-NN	kNearest Neighbours	An instance based learning algorithm choosing k nearest instances.	(Sun & Huang, 2010)
DT	Decision Tree	A probabilistic graph-based model.	(J. Chen, Luo, & Mu, 2009)
NB	Naïve Bayes	A Bayes' Theorem based probabilistic classifier model.	(Frank, Hall, & Pfahringer, 2003)
AdaBoost	Adaptive Boosting	An ensemble method for boosting biased ML algorithms.	(Bauer & Kohavi, 1999)
Bagging	Bootstrap Aggregating	An ensemble method for reducing variance of underperforming ML algorithms.	(Bauer & Kohavi, 1999)
RF	Random Forest	A probabilistic graph-based estimator; uses a predefined number of DT to avoid overfitting through correction.	(Zhou & Tang, 2003)
SVM	Support Vector	A non-probabilistic binary estimator; can use linear or non-linear kernel.	(Tabrizi & Cavus,

	Machines		2016)
XGB	Extreme Gradient Boosting	An ensemble technique for regression and classification prob lems, which uses weak models, typically DT, to produce an estimator.	(T. Chen & Guestrin, 2016)

2.2.2 Machine Learning Algorithms for Unsupervised Learning

Unsupervised learning receives unlabeled data as training input without knowing the result a priori. It is the ML task of building a model by deducing and describing hidden structures present in data, using mathematical functions to reduce redundancy, extracting rules or organizing data by similarity. What distinguishes supervised and reinforcement learning from unsupervised learning is that there is no error or reward signal to evaluate a potential solution since the examples given to the learner are unlabeled. This kind of learning is closely related to the problem of density estimation in statistics (Erhan et al., 2010; Hofmann, 2001). Example algorithms are the a priori algorithm, association rule learning, k-Means etc.

Cluster analysis is one of the unsupervised techniques used to analyse and capture the underlying relationships among the data observations. In more detail, the K-Means algorithm clusters data by trying to separate samples in *N* number of groups of equal variances, minimizing a criterion known as the inertia or within-cluster sum-of-squares. This algorithm requires a fixed number of clusters to be specified. It scales well to large datasets and has been used widely across a large range of application areas.

That task seeks to identify homogeneous group observations to partition the data into distinct areas which share similar properties. Knowledge areas or patterns are the terms assigned to these groups since in later section they are going to be used as the area of expertise for each knowledge source individually. Clustering algorithms are the mechanisms employed for this task by the blackboard. K-means is one of the algorithms responsible for the identification of the different knowledge patterns in CHIMACS.

The K-means algorithm divides a set of *N* Samples *X* into *K* disjoint clusters *C* of equal variance by minimizing a criterion known as the inertia or within-cluster sum-of-squares, where each cluster is described by the mean m_j of the samples in the cluster. The means are also called the *"centroids"* of the cluster. The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum of squared criterion in equation (3) ("Cluster Analysis," 2007).

$$argmin_{\{r_{nk},\mu_k\}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

Equation 1 - K-Means Objective Function

where x_n is the value of variable *n, r* is the weight and μ is the mean of the variable over the examined cluster (Mooi & Sarstedt, 2011). This implicit objective function above measures the sum of distances of observations between their knowledge areas (cluster) centroid and is called Within-Cluster-Sum-of Squares (WCSS). WCSS is a measure of variation within a knowledge area and thus minimizing the WCSS is the methodology that provides valuable information in regards to the optimal number of different knowledge areas.

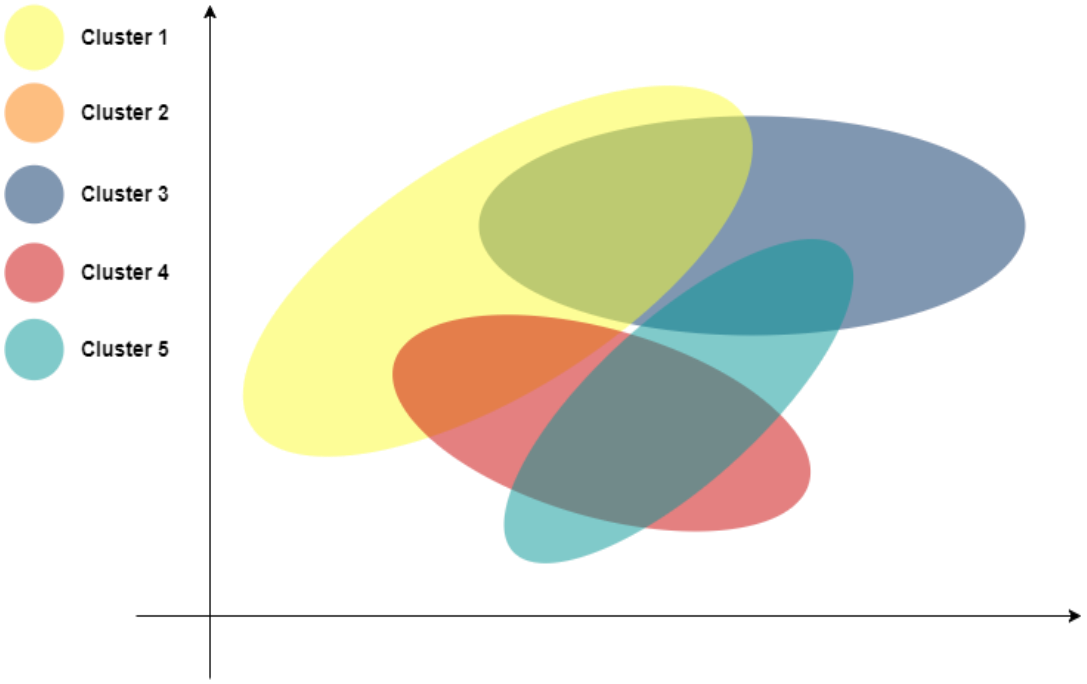


Figure 4 – Cluster Analysis

In Figure 7 there is an example of the discrete clusters of a dataset. The knowledge areas are determined through an iterative process that tries to minimize the distances between the candidates that have common characteristics as described above. Silhouette analysis is performed in order to determine the optimal number of clusters ("Cluster Analysis," 2007). Blackboard is now aware of the patterns that emerge from the data and is capable of performing dedicated voting and trust update.

2.2.3 Prediction Evaluation

To evaluate the performance of a model a number of performance measures are employed. Typically, the distribution of the predicted class $Y \in [0,1]$ defines the most appropriate performance metric. In the case of binary classification, relatively balanced distribution of the class, the accuracy measure is widely used for the evaluation of the model's performance. On the other hand, where the distribution is skewed other metrics like recall (sensitivity) and precision are used for evaluating the model's performance.

Performance Metrics Elements

- ***True positives (TP)***: the number of correctly predicted observations as belonging to the positive class.
- ***True negatives (TN)***: the number of correctly predicted observations as pertaining to the negative class.
- ***False positives (FP)***: the number of observations predicted as positive, which belong to the negative class.
- ***False negatives (FN)***: the number of examples predicted as negative, where the class they belong to is positive.

In Chapter 5, a detailed description of the performance metrics is given in the experiment outline.

Cross-Validation

N-fold cross-validation is employed to provide an overview of how accurately a predictive model will perform in practice. For example, in n-fold cross-validation, where $n = 1,2,3,..,n$, first, the data is divided in n consequent parts and estimators are built on $n - 1$ parts of the data. Finally, each model is tested on the n_{th}, part of the data and the average and standard deviation of the performance is calculated.

2.3 Multi-Agent Systems & Architectures

Architectural styles are consisted of specific features which act as restrictions from activities scope point of view. Considering that, the appropriate selection of an architectural style is essential during the implementation phase (Fard & Rafe, 2014). Multi-agents have been used in different contexts, in order to deploy distributed MAS both for traditional and ML applications. Examples can be seen in areas like process control (Linkens, Abbod, & Browne, 1999), business workflows (Kapetanakis, Petridis, Knight, Ma, & Bacon, 2010), robotics (Brunet, de Lafontaine, & Lachiver, 2003) and intelligent system control (Heath, 2009; Linkens et al., 1999). Many of these frameworks have been used in domain specific context MAS applications like air and road traffic control, information gathering, robotics, etc. being most popular amongst them the DMASON, JADE, JACK, JACAMO, JASON and DARBS (Heath, 2009).

2.3.1 MAS Architectural Structure

This section is focused on the presentation of the chosen architecture, the blackboard. It aims to discuss the arguments that helped to choose the respected architecture. This presentation should allow for comparison between and assessment of different MAS infrastructures. The next section provides a brief presentation of the most used architectures. Sub-sections 2.3.2.1 and 2.3.2.1.1 justifies the choice behind the implemented architecture.

2.3.2 Architectures

Designing a MAS requires the architecture that is going to be the basis of the application. All of the above mentioned multi-agent frameworks are used for general purpose applications and as a result the developer is in charge of the architecture that he is going to apply. DARBS, on the other hand, implements blackboard architecture and the developer is obliged to follow this paradigm (Nolle, Wong, & Hopgood, 2002).

Due to the wide variation of the application, different architectures exist with each one having specific architectural attributes e.g. coordination, communication, organization protocols. Blackboard, Archon, Osaca, Dide, Adept (Shehory, 2001; Sorici, 2012) are some of the architectures implemented in a multi-agent applications. Each architecture has its predefined design properties and of course due to this fact, every infrastructure has its pros and cons. The main difference that distinguishes the blackboard architecture from the rest architectures is that fact that is not domain specific. Where the rest of the architectures are tied to the domain which they were implemented for. That implies that the domain experts can take advantage of the pre-defined features and alleviates the issues of efficiency and efficacy.

Another characteristic of the blackboard is the central management. It is a component that supervises the multi-agent environment and it is not offered as a feature in none of the rest of the architectures. At the same time, blackboard offers features that are not tied to a specific domain. That gives the room to choose and implement techniques and protocols that suit in our needs. The main features the blackboard architecture has to offer are flexibility, central management and

interpretability. However, the main disadvantage of is that flexibility burdens the domain expert with the proper choice of the tools for implementation. Poor implementation of the respected processes and tools could cause efficiency issues. The next sub-sections provide a description of the chosen architecture and it's features.

2.3.2.1 Adept & Archon

Archon is a general multi-agent infrastructure with characteristics such as flat organization and static openness. It supports distributed control, information exchange and ad-hoc sub-systems without re-compilation. Adept (Advanced Decision Environment for Processes Tasks) and Archon are multi-agent architectures dedicated to the management of business processes using autonomous agents. More specifically, Adept architecture was implemented by British Telecom and is based on Archon agent-model. It involves 9 agents and 93 tasks developed with a modified version of C Language Integrated Production System. The components for knowledge sharing, negotiation and communication functions are depicted in figure below (Wittig, 1992; Wooldridge & Jennings, 1995).

2.3.2.2 Osaca & Dide

OSACA (Open System for Asynchronous Cognitive Agents) is a general-purpose multi-agent framework, while DIDE (Distributed Intelligent Design Environment) handles more specific type of problems. In DIDE, the agents can be imported or removed dynamically without disrupting any processes or halting the system, where in OSACA that is not possible. Finally, broadcasting a message to the entire agent network in real-time is an advantage that exists in both architectures (E. S. a. J. P. Barthes, 1993; W. S. a. J.-P. Barthes, 1995).

2.3.2.3 Blackboard Architecture

Blackboard is an architecture that is not pended on any particular domain and it offers the capability to integrate heterogeneous modules, referred to as KS, that offer the tools for solving a specific part of the main problem. ("Book review: Blackboard

Architectures and Applications Edited by V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum (Academic Press)," 1990).

As we described previously in section 1.4, most of the architectures are applied in a specific context that depends on the requirements of the MAS implementation. Blackboard based systems are not limited to a single expert Knowledge Sources (KS) and attempt to employ more than two reasoning KS. They can be homogeneous or heterogeneous. For example, a system might integrate a module based on CBR system and potentially another solver based on neural networks ("Book review: Blackboard Architectures and Applications Edited by V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum (Academic Press)," 1990).

Each individual knowledge source corresponds only to a specific part of the problem. The blackboard central manager captures the current state of the problem solution, while the KS contribute to the solution. Hearsay-II speech recognition project is considered one of the first blackboard systems (Erman, Hayes-Roth, Lesser, & Reddy, 1980).

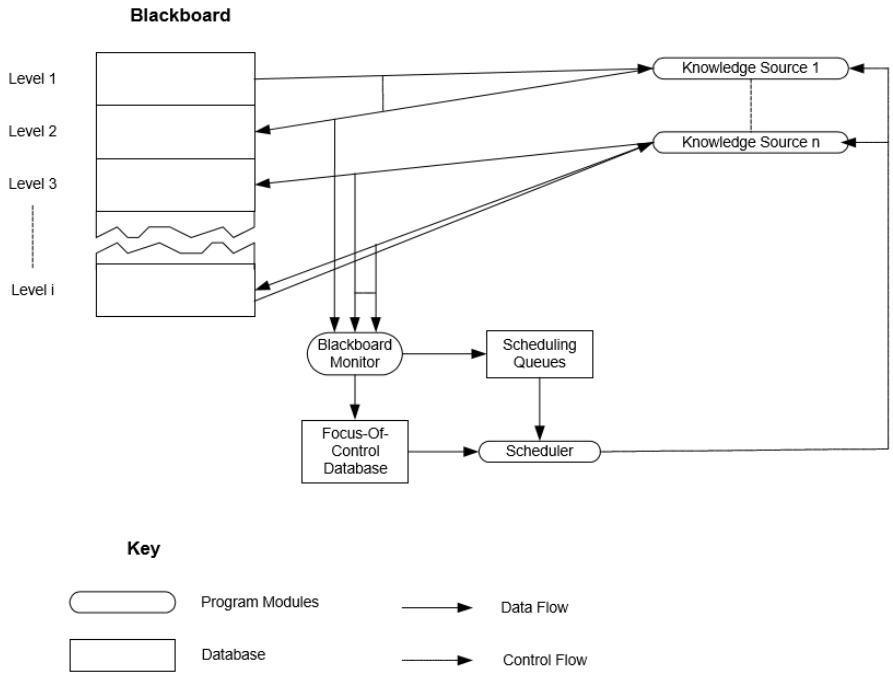


Figure 5 - Hearsay III Architecture

The Hearsay-III architecture (Erman, London, & Fickas, 1981) is an extension of Hearsay-II system that contains no specialised knowledge.

2.3.2.3.1 Centralized Control

In centralized control, there is usually a scheduling agent responsible for the coordination process. Its job is to follow plan instructions, analyse the current goals, objectives and possible conflict points and orchestrate the rest of the agents. Despite of being the simplest way of ensuring coordination by predefining each agent's responsibilities, capabilities, connectivity and the control flow of the whole system, this approach presents some drawbacks. Excessive control over the agent components mitigates against all the benefits of DAI, e.g. speed due to parallelism, reliability, robustness, concurrency, autonomy etc. (Schumacher, 2001) .

The agent that coordinates is necessary for splitting the plan into different parts which can be executed concurrently and uses several task allocation techniques to distribute sub-plans among executing agents. If executing agents can both follow the plan strictly based on the correct knowledge of the world and predictions of their actions, their parallel execution of sub-plans can reach a state of the world in consistent with the initial goals. However, this schema can hardly work efficiently, unless efficient decomposition and distribution of task is implemented (Weerdt & Clement, 2009). In a nutshell, most of the time might be spent on the efficient decomposition of the problem into tasks and the communication between the agents, if that's necessary.

2.4 Reinforcement Learning in Hybrid Multi-Agent Classification Systems

Taking into account the knowledge from the previous sections and especially EM from section 2.3.2 it's obvious that there is not a single best ML approach. Research in classification is focused on combination of multiple classifier systems, which can be built exploiting either the same or different ML models and/or datasets (Van Dyke Parunak, Nielsen, Brueckner, & Alonso, 2007). These systems combine classification decisions at different levels collaboratively overcoming limitations of

traditional approaches based on single classifiers. This section describes multiple classifier system (MCS) from the point of view of Hybrid Intelligent Systems.

Hybrid Systems offer many ways for handling complex problems, involving uncertainty and high-dimensionality of data (Luo, Chen, & Zhang, 2013). Hybridization appears in many domains of human activity and it seeks to exploit the strengths of the individual experts, resulting in enhanced performance by combining them. Fig. 7 is a representation of the domains covered by the H I S. Some of them use probabilistic or fuzzy representations and feature extraction to deal with the uncertainty and ambiguity in the data (Khalid, Khalil, & Nasreen, 2014; Murphy, 2012) and others use stochastic process to deal with optimization related problems (Cotter, 2003). Finally, classifiers that implement intelligent decision processes are also subject to hybridization using various forms of combination.

In this section, focus is given in this specific domain and referring to Wolpert's theorem (Gómez & Rojas, 2015): "*Since each has its own domain of competence there is not a single classifier modelling approach which is optimal for all tasks*".

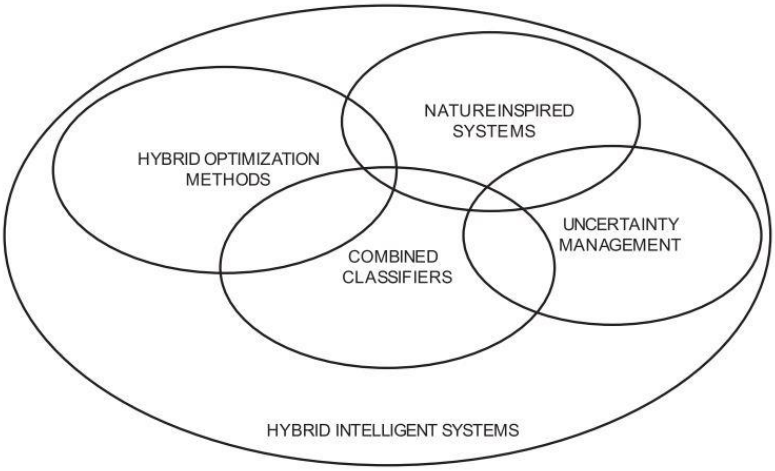


Figure 6 - Domains of Hybrid Intelligent Systems

Summarizing (Woźniak, Graña, & Corchado, 2014) :

- HIS are combinations of computational intelligence techniques to deal with a given problem, ideally covering all analysis phases from data processing up to final decision making. Specifically, due to their expertise they mix

individual heterogeneous expert views, blending them into one effective working system.

- Information Fusion refers to the ways of collecting and combining information sources in order to provide new properties that could possibly allow solving the examined problem more efficiently.
- Multi-Classifier Systems (MCS) or EM focus on the combination of classifiers form heterogeneous or homogeneous experts so as to give the final decision.

2.4.1 Hybrid System Structure and Topology

The general structure of HIS is depicted in Figure 10 following a classical pattern recognition (Giacinto, Roli, & Fumera, 2000) application structure. The main issues of hybrid intelligent system design are:

- System topology: How to interconnect individual classifiers.
- Ensemble design: How to generate and select the most valuable classifiers and their decisions.
- Fuser design: Figure out how to build a decision combination methodology, the fuser which can exploit the strengths of the desirable classifiers and optimally combine them.

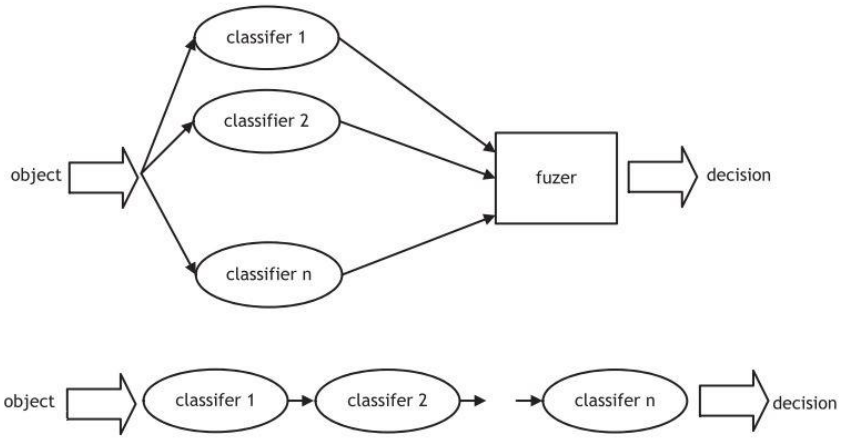


Figure 7 - Structure of Multiple Classifier System

Figure 10, illustrates the two topologies employed in multiple classification systems. The majority of the literature mentions structures in a parallel topology (Kuncheva,

2004). In this structure, each classifier receives same input data, so that the final decision from combination output is made on the basis of the outputs of the individual classifiers independently. Alternatively, a second representation of Figure 10 presents a serial topology, where individual classifiers are applied in sequence, implying a form of ranking or ordering based on their individual capabilities. When the primary classifier cannot underperform a given object, e.g., because of the low confidence in its result, then the data is fed to the next classifier in order (Lam, 2000; Rahman & Fairhurst, 1999) and so on. This topology is adequate when the computational cost of a classifier is so high, so that the primary classifier is the cheapest one and secondary classifiers are computationally most costly (Fumera, Pillai, & Roli, 2004).

Hybrid systems provide the opportunity to combine different experts that handle the given problem differently. This enhances the decision process and provides the ability to reach a more efficient solution. As the above approaches have been implemented in EM, it is logical to adopt parts of this research and adapt them for our benefit. A reasonable question would be "Why wouldn't we use EM"?. As it is stated in the previous section, ensembles present certain drawbacks as far as overfitting is concerned. Thus, in our case we adopted parts of the ensembles approach adapting it into a more robust process.

2.4.2 Reinforcement Learning Trust in Multi-Agent Systems

Trust is one of the main concerns when talking about multi-agent large scale distributed systems. It allows healthy interactivity between agent components that rely on one another in a dynamic environment. A common scenario in MAS is that agents might provide solutions with different qualities. According to (Granatyr et al., 2015; Ramchurn et al., 2004), trust in MAS is conceptualized in the following ways:

- Individual-level trust, whereby an agent judges the eligibility of the interaction with its partner.
- System-level trust, whereby the agents are forced to be trustworthy by protocols and mechanisms that regulate the system process and interactions.

In order for the trust to be calculated within the multi-agent system, it is required that the performance of an agent is described in quantitative manner and rating should be

given (Ramchurn et al., 2004). Recording agent's performance over time can provide essential information on how good and consistent was and thus, apply calculation metrics to extract the trust level.

Reinforcement learning (RL) is learning by interacting with an environment. An RL agent "learner" receives feedback and learns from the consequences of its actions, rather than from being explicitly taught what to do. An agent selects and exploits the actions that maximize the future reward based on past experiences while at the same time explores new actions. Essentially, it is considered trial and error learning.

The specific type of learning is concerned with how an ML algorithm ought to take appropriate actions in an environment so as to maximize a cumulative reward value for the action taken (Abbeel & Ng, 2004). The problem environment is formulated as a Markov decision process (MDP) as many reinforcement learning algorithms for this context utilize dynamic programming techniques (Brázdil et al., 2014; St, Ross, Pineau, Chaib-draa, & Kreitmann, 2011). The main difference between the classical techniques and reinforcement learning algorithms is that the latter do not need knowledge about the MDP. They also differ significantly in situations in which the learner's behavior is in some way uncertain or inappropriate. Correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected. Reinforcement learning only says that the behavior was inappropriate or not and usually how inappropriate (e.g. +1, -1) it was. In other types of learning, the "learner" knows exactly what it should have been done. Example algorithms include deterministic Q-Learning, Monte-Carlo Methods, Temporal Difference Methods, Deep Learning etc. (Leslie Pack Kaelbling, 1996). Reinforcement learning cycle is depicted in Figure 11 below:

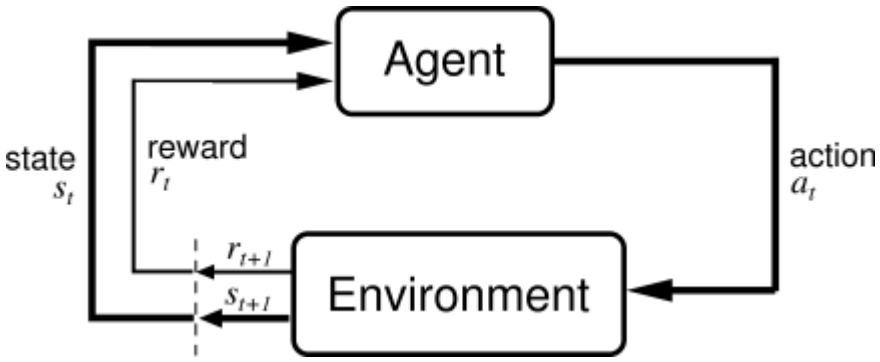


Figure 8 - Reinforcement Learning Model

Elena Irena Teodorescu (Teodorescu & Petridis, 2013) investigates the application of a blackboard-based multi-agent approach employing Heterogeneous Case Based Reasoning agents (MHCBR). CBR agents operate on different structures/views of the same problem domain. The blackboard supervisor coordinates the process in a transparent and autonomous way enabling it to retrieve solutions for a new problem from more than one case-base. This framework includes sub-processes for subscribing case-base providers through agents, creating dynamic domain structures and retrieving optimal solutions through a mechanism measuring the competence of a provider case base by using agents and employing a blackboard communication architecture.

A multi-agent classifier system for tackling data classification problems based on reinforcement learning has been proposed (Pourpanah et al., 2017). The agent environment, which is comprised by hybrid neural networks and a trust measurement using Q-learning and Bayesian formalism, is formulated to identify the agent's competence. The reinforcement signal that the team manager agent receives is a numerical reward, which encodes the success of a prediction outcome and the manager seeks to select the team of NNEs that maximize the accumulated reward over time, hence produce the best classification accuracy. Neural Network Ensembles (NNE) presented very good results compared to other methods found in literature, as will be discussed later and the system was capable of identifying the best team of agents.

2.5 Computational Social Theory and Voting in Multi-Agent Systems

Social choice theory deals the methods for aggregating the preferences of multiple entities inside a multi-agent environment. An example, includes voting procedures, that aggregate the voters preferences in a voting procedure. The goal is to determine an alternative form a set of alternatives and use that dominant alternative to conclude to a final decision. Multi-agent systems can be viewed as a "society" of agents, with each agent having different goals and objectives, different capabilities and understands the problem domain differently. Then procedures for aggregating their assumptions so as to be able to make decisions based on a collective process. Computational voting is performed under the following setting, firstly, a set of outcomes or alternatives and a number of agents that have preferences over them. The goal of a social choice function is the mapping of profiles of preferences to an outcome, where an entity e.g. an agent can manipulate the voting scheme, by how their preferences are presented.

2.5.1 Social Choice Scope

Projecting social choice theory foundation in MAS applications, we consider two similar areas, namely game and decision theory. Decision theory tools are utilized when it is essential to capture, analyse and model an agent's actions, choices and preferences ("Decisions in Human Centric Multiagent Systems: Dealing with Softness and Bipolarity in Judgments, Intentions and Evaluations," 2016; Schweitzer, 2007). On the other hand, game theory focuses on analysing and modelling the interactions of a number of agents inside an environment and final goal is to take an optimal decision based on the individual decisions and preferences. The distinction between these two theories relies on the way a final decision needs to be taken. Game theory is concerned with the interactions between one agent with the rest of the agents, while decision theory is concerned with one individual agent at a time. Finally, social choice theory extends that and is focused on the interactions between a group of agents as a whole. Three major questions are asked and related with the above mentioned theories, in social choice theory is asked *"what benefits this group?"*, while in decision theory is asked *"what benefits this agent?"* and in

game theory *"how can this agent perform well, given that others try too?"* (Endriss, 2014).

2.5.2 Computational Social Choice in Multi-Agent Systems

MAS are systems composed of several autonomous entities (agents) which interact with each other, including operations and processes related to cooperation, coordination and competition. There are similarities between human societies and multi-agent systems, since people behave and interact with other members of the society and functions collectively in most of the cases. In macro-economics, the type of entities generally studied under the prism of social theory, share common societal attributes with agents in MAS e.g. decentralization, self-interest, heterogeneity, etc. (Endriss, 2014).

Computational Social Choice gained a lot of attention from AI/ML researchers in the last two decades. The attributes of MAS stated below, have been shown to be facilitated from Social Choice theory approaches (Procaccia, 2008) :

Computational Social Choice is concerned with the following areas (Chevaleyre, Endriss, Lang, & Maudet, 2007):

1. *Judgment Aggregation*. It is concerned with the aggregation of agents' decision into collective judgements.

2. *Computational Voting Theory*. In many computational and especially, MAS, it is desirable to have a voting mechanism which enables the accommodated entities e.g. agents within the system to make a collective decision on a given issue.

These three, are the areas examined under this thesis research work.

When we consider an autonomous and therefore an intelligent system, we conceptualize different autonomic elements (components/agents/services) heterogeneous or not that in their entirety make a system present a partially autonomic behaviour. Relationships among these autonomic elements are also a very important part of autonomy since different components are required for the accomplishment of a task. We consider these elements as different agents inside a

multi-agent system that its main task is to present autonomic and collaborative behaviour. This motivates the use of AI techniques incorporated inside a MAS, in order to deal with that complexity and thus support the decision making process.

2.5.3 Voting

Given a set of available alternatives, a group of voters and their preferences over these alternatives, voting is a framework for choosing a best alternative. The example given in section 2.9.1, is a classic example of political elections, where every political candidate standing for election is an alternative and voters express their preferences on the ballot sheet. But as literature review showed, MAS share common social attributes, thus voting rules can also be applied to MAS too. For example, the alternatives may be different recommendations that a group of agents needs to execute together and each individual agent could have its own preferences over the alternatives, that are determined by their structure and reasoning capabilities (O'Leary, 1994).

Literature review on social choice theory, provides a simple mathematical (Milgrom∗, 2004) framework for modelling the process of voting. The formal framework is represented as showed below:

- *A finite set of voters (electorate) N = {1, . . . , n},*
- *Usually a finite set of alternatives X = {x1, . . . , x},*
- *One preference order i for each agent i ∈ N. Every such preference order i is taken to be a strict linear order on X,*
- *L(X) represents the set of all such linear orders, with preferences and ballots belonging to it.*

For example, a voting rule is a function $F: L(X)\ n \rightarrow 2\ X \setminus \{\emptyset\}$, that provides the means to map any preference profile to a non-empty set of alternatives. Due to the possibility of ties a single winner may not be an option.

Voting theory provides many different voting procedures and rules (Endriss, 2014; Taylor, 2005). The choice of the appropriate voting procedure and consequently the rules that apply, depend entirely on the requirements (axioms) of the voting process. To exemplify the range, only voting procedures and rules meeting the criteria of this

thesis research are stated and analysed. A detailed analysis on the axioms related to this research and on the procedures followed is presented in Chapter 3. Figure 15 depicts a general voting process:

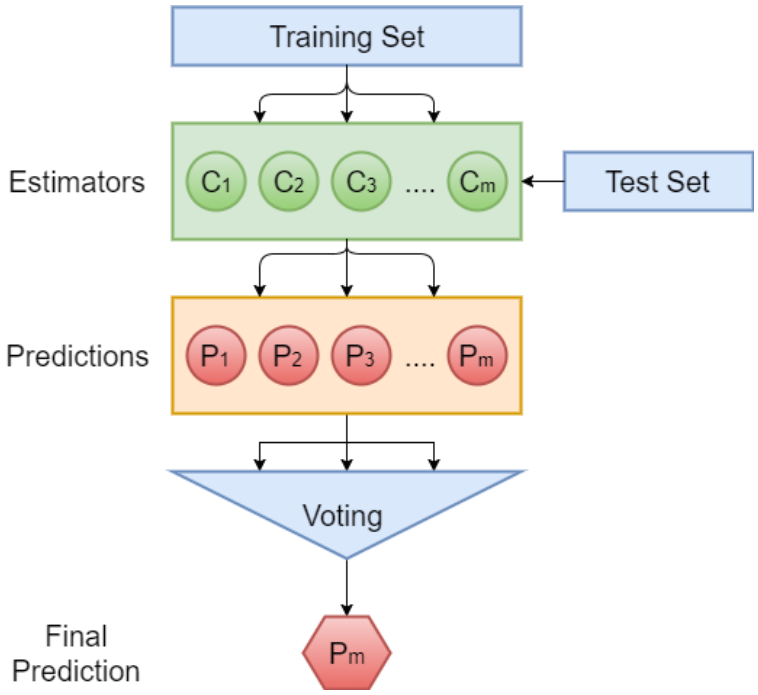


Figure 9 - General Voting Workflow

2.5.3.1 Scoring Functions

Given only two alternatives, then all the voting procedures according to (Endriss, 2013)coincide and intuitively produce valid results. However, for the purposes of this research only the rules applied are described.

2.5.3.1.1 Majority Voting (Plurality)

In political elections, the most predominant social choice function (SCF) is the Plurality function also known as simple majority function. Under Plurality, each agent in a multi-agent system awards one point to the most preferred alternative it ranks first. The alternative that summed the most votes, wins the election. Other SCF examples similar to the Plurality function is the Veto and the Borda rule, devised in 1770 by Jean-Charles de Borda (Endriss, 2014): *"A scoring function can be expressed by a vector of parameters α = (α1, . . ., αm), where each αi is a real*

number and α1 ≥ · · · ≥ αm. Each agent awards α1 points to its first-most-preferred alternative, α2 to its second-most-preferred alternative, etc. The alternative with the most points dominates. "

2.6 Summary

This Chapter summarised related research work and also provided the necessary background for the proposed methodology in order to increase predictive force by collaborative reasoning in a multi-agent system. The next Chapter introduces the underlying theoretical background of this methodology, which includes three distinct set-ups of the proposed approach.

Chapter 3 Coordinated Heterogeneous Intelligent Multi-Agent Classification System

An intelligent blackboard based multi-agent framework accommodating heterogeneous ML classifiers is proposed in this Chapter. Firstly, a blackboard controller component is proposed. This component includes a naive and two intelligent versions of the controller. Secondly, five variations of SCF and SWF voting procedures are employed. The resulting Coordinated Heterogeneous Intelligent Multi-Agent Classification System (CHIMACS) combines the capabilities where individual experts possess over a specific knowledge area (cluster). The purpose of such an attempt is to take advantage of the diversity in error rates and increase the overall classification performance.

In the beginning, an introduction on blackboard based classification systems is introduced in Section 3.1. The proposed structure of the framework is introduced in Section 3.2. The structure of the intelligent agents accommodated by the proposed framework is shown in Section 3.3. Sections 3.4 and 3.5, are related to the trust calculation and the update process as well as the techniques applied on the gamma and learning rate parameters. The formulation and description of the voting functions is described in section 3.6. Finally, Section 3.7 and Section 3.8 provide a conclusion and a summary of the Chapter.

3.1 Introduction

The Blackboard paradigm was introduced in the 80s and used on a number of applications of expert systems (Morgan, 1988). In the early 90s, Petridis et al proposed a blackboard based architecture for the integration and combination of diverse hybrid ML, AI and mathematical models (Petridis & Knight, 1996; Petridis,

Knight, & Edwards, 1991). A number of other blackboard based architectures and systems were proposed and shown to work effectively over a range of application areas (Nolle et al., 2002). Teodorescu (Nolle et al., 2002; Teodorescu & Petridis, 2013) proposed a blackboard based architecture for a hybrid MAS to combine heterogeneous Case Based Reasoning classifiers. In this, she proposed a reinforcement learning algorithm based on the concept of agent confidence and agent trust and a weighted voting process. However, in this work, the hybrid element of the problem was based on similar agents operating on different datasets. This research proposes a Coordinated Heterogeneous Intelligent Multi-Agent Classification System (CHIMACS), whose approach applies on hybrid systems consisting of heterogeneous classifiers operating on the same datasets. This presents the additional issue of different data representation and reasoning paradigm among agents that introduces the challenge of reconciliation and normalization of the various views of the dataset and the classification process among the agents.

This chapter presents an approach towards a hybrid multi-agent architecture utilising heterogeneous agents that operate on different problem angles and individually constitute a reasoning mechanism towards a global near optimal solution. A multi-agent framework is proposed based on the Blackboard architecture coordinating the process, orchestrating intelligent agents and aggregate extracted knowledge towards the identification and construction of a more suitable and accurate solution.

Irena Teodorescu (Teodorescu & Petridis, 2013) presented an investigation into applying an adaptive agent-based CBR to Multiple Heterogeneous Case Bases. The proposed system is based on the blackboard architecture so as to support the efficient collaboration between the homogeneous agent components and achieve an efficient global solution. To optimize the collaboration among the agents, appropriate similarity metrics were applied. HMCBR formed trust in heterogeneous databases by measuring the quality of the retrieved solution in each classification cycle. This research investigates a multi-agent approach based on blackboard architecture for efficient agent collaboration and knowledge aggregation, extracted from multiple heterogeneous ML agents for increasing the overall classification accuracy.

3.2 CHIMACS Framework Structure

The structure of CHIMACS is shown in Figure 16. It contains three layers. The top layer produces the final decision. The results from the voting procedures are sent to the blackboard manager which concludes to a final decision. The middle layer comprises individual agents, while the data are placed in the bottom layer. Its modularized nature allows to plug in agents on demand. The entire process cycle is supervised by the blackboard manager. A new trust measurement is proposed (as detailed in section 3.4). It is an algorithm based on trust and voting procedures is employed to enhance the predictive performance through decision combination. Then, a reinforcement learning approach is applied to update the trust score corresponding to the validity of the prediction that corresponds to each test data sample.

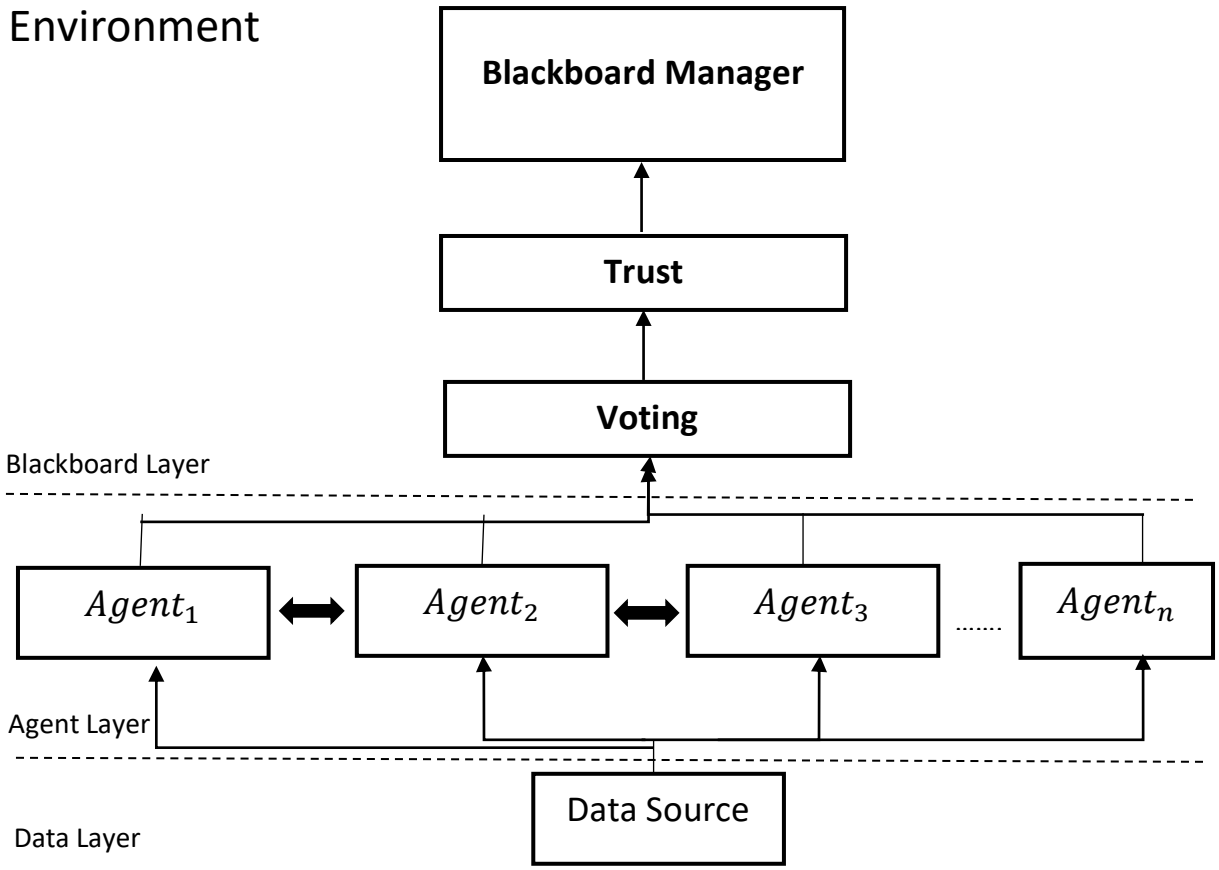


Figure 10 - CHIMACS Layers

In figure 17, a detailed graphical representation of the structure of the proposed Coordinated Heterogeneous Intelligent Multi-Agent Classifier System (CHIMACS) is depicted. It is based on the blackboard architecture and consists of three main modules. The blackboard manager module is responsible for orchestrating the entire process which include agent coordination, voting, trust update and final decision making. The knowledge area module represents the expertise of the KS and the agent repository module accommodates the agents. An approach using a trust measure based on reinforcement learning techniques, using Bayesian discounted future reward and a number of five voting procedures is proposed. The initialization of the trust score for each agent is random and in every execution cycle is updated depending on the validity and the quality of the predictions pertaining to each known data and class observation. Figure 17, presents the workflow process of the proposed framework.

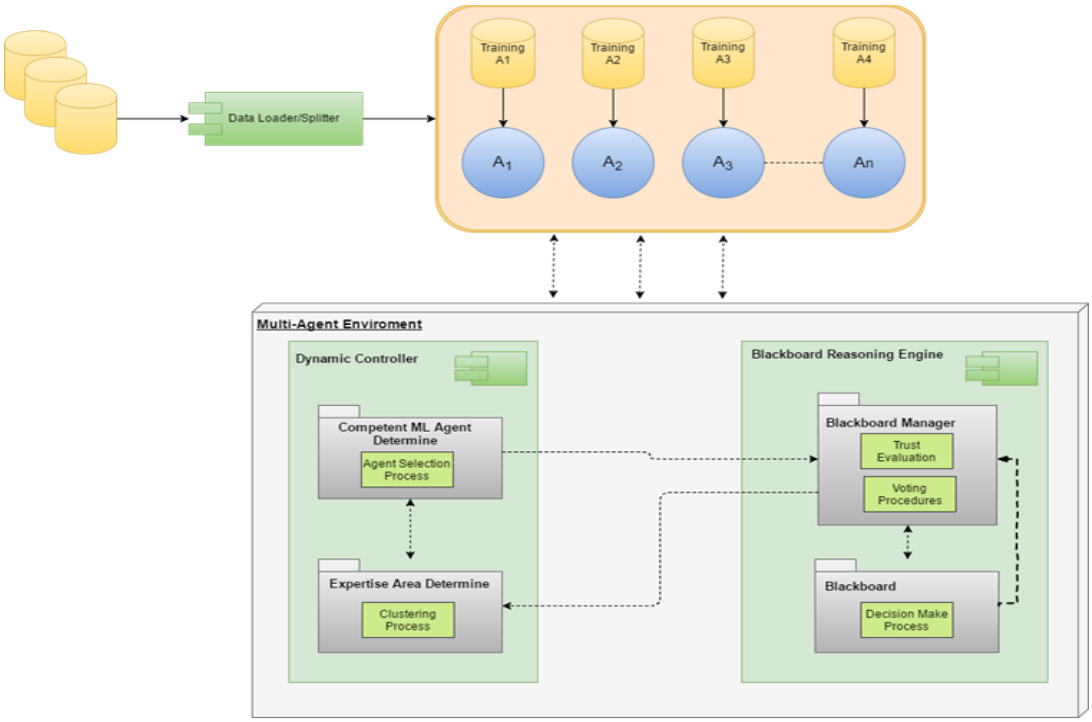


Figure 11 - CHIMACS Structure

A workflow diagram of the key processes in CHIMACS is shown in Figure 17. As stated earlier, measuring trust and knowledge combination through voting are important steps. A method for combining reinforcement learning techniques with Bayesian estimation of the reinforcement signal and voting is proposed to enhance the overall predictive performance. A novel use of reinforcement learning technique is employed for the trust measurement in this research. The proposed method consists of two steps. In the first step, the initial trust value of each agent is randomly assigned usually with very small initial values. In the second step, a reinforcement learning technique is employed to update the trust value of each agent when it provides a prediction pertaining to each sample test. Voting is essential part of the process since it guides the individual or public opinion regarding an alternative. A variety of SCF and SWF functions are employed as a tool of opinion expression. The details of the steps involved are explained in the next sections.

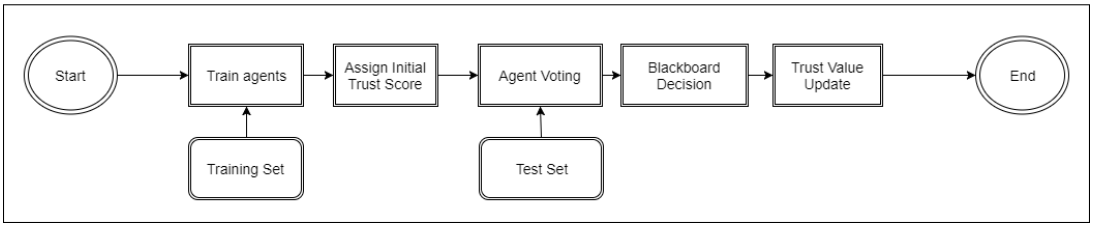


Figure 12 - CHIMAS Workflow

3.2.1 Voting

Voting state is related to the process followed regarding the final decision. Blackboard controller receives a number of suggestions from its agent's members and different versions of the voting process are used in order to conclude to the final decision. In the following chapters, all the implemented voting procedures are described thoroughly, however, in the results section only those presenting significantly different results are described. Five voting schemes based on two main concepts are implemented for the purposes of this study. The first voting is based on *"majority"* voting rule and the second is based on the *"aggregative"* voting rule. The winner for the first voting process is nominated through the highest bid submitted to the blackboard from the agent members. Each individual expert agent is evaluated from its bid score. Majority voting, on the other hand, allows the expert agents to combine their knowledge into one final proposal. The blackboard is then responsible for taking the final decision and updating trust measures for all agents.

3.2.2 Trust

According to (Ramchurn et al., 2004; Tweedale & Cutler, 2006) computational trust has been defined as the level of confidence an authority has over an individual that is going to fulfil its promises as expected. Trust and reputation in open MAS has been widely employed in (Mui, Mohtashemi, & Halberstadt, 2002; Teacy, Patel, Jennings, & Luck, 2006) as measurable metrics to evaluate the level of trustworthiness and eligibility of agent members. Trust based on reinforcement learning techniques is one of the main contributions of this study. Blackboard controller is responsible for calculating that metric. In this way, the system is capable of evaluating the performance of each agent and combining their knowledge. Trust measurement has an impact on the voting process thus voting score influences the quality of an

individual prediction. The purpose of trust is to influence the voting score in such way that the overall predictive performance increases.

For a given classification task it is proved that the suggested multi-agent classification system is able to identify and exploit the strengths of the individual classifier agents at its disposal to produce the high-quality compound classification system overcoming the performance of individual classifiers and thus increasing the prediction accuracy. To this end, in the proposed approach, the agent entity operates on multiple structures/views of the same problem or as we will call them from now on, knowledge areas. The different element procedures of the blackboard orchestrate the entire process in a transparent and semi-autonomous way. Individual solutions are proposed by each agent element, but at the same time the intelligent blackboard controller is capable of managing and exploiting different capabilities that derive from agent heterogeneity and aggregating the produced knowledge of its variants, towards the formulation of a better global solution compared to what each single agent could suggest.

3.2.3 Blackboard Controller

The controller is considered as the manager that mediates between the data and agent layer and takes the final decision after the voting process. Layer activity diagram in Figure 18 describes the blackboard pattern CHIMACS is based on.

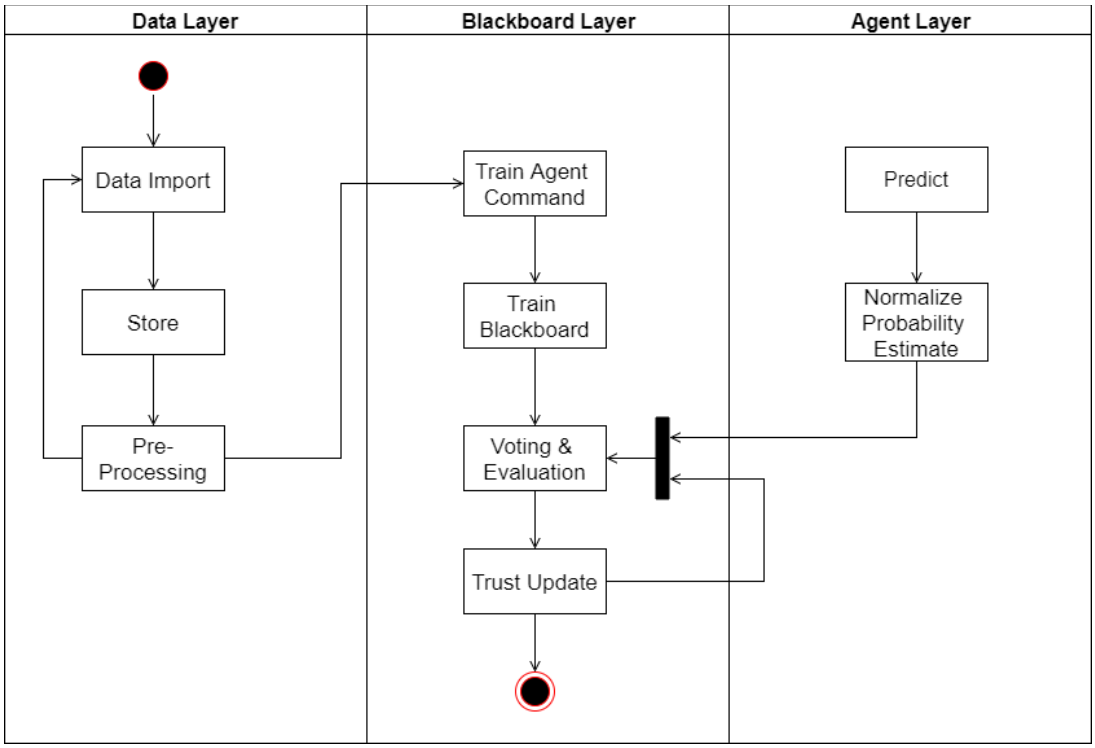


Figure 13 - CHIMAS Layer Activity Diagram

Many advantages of the blackboard architectural pattern have been presented in section 2.3.2.1, with the most important resembling these research requirements being the adaptability through generality. The classification process is coordinated in a centralized manner though a blackboard manager, however, freedom is given to its KS to behave in an autonomous decentralized way. Blackboard layer mediates between the data and agent layer. It holds the components for managing the data instances for training and classification, loads the models for the ML agents, performs the voting and trust re-evaluation procedure. Finally, visualizes the results during the execution process. Algorithm (3) illustrates the steps followed in pseudo code.

1. Initialize $test\ data$.
2. Initialize $training\ data$.
3. Deliver $test\ data$ to agents.
4. **Repeat:**
    a. **Repeat:**
        i. Predict $observation$
        ii. Vote
        iii. Validate $result$.
        iv. Validate agent prediction
            if $\hat{y}\ is\ true$, increase trust;
            else, decrease trust.

    until $agents\ End$.
Until $test\ data\ End$.

Algorithm 3 - Blackboard Procedure Pseudo Code

3.3 Intelligent Agent Architecture

ML agents seem a solution for managing, sharing and utilizing knowledge as gained from their training process. Differentiation in structure impacts their learning too. By using different tools they can learn from data and improve their reasoning mechanisms in different areas of the dataset while contributing to / improving the knowledge perception standards of their utilized (control) mechanism. This leads to autonomous systems with elements of self-adaptation capabilities.

3.3.1 Agent Structure

In this research work, the multi-agent system consists of agent components with a three-layer structure following the Belief, Desire, Intense model (Bonura et al., 2009). Agent implementation for this research has followed the BDI model structure.

Agent beliefs are related to itself in the sense that the agent is aware of its capabilities meaning that the agent knows how confident he can be based on the received training. In some cases, the agent tries to manipulate the system by

presenting extremely confident about its capabilities. At the same time overconfidence or under confidence, thus its beliefs, affect the agent's suggestion negatively or positively. Based on the current state of the agents, their beliefs don't change since they do not re-train themselves.

The agent knows what it wants to do and that is represented in the form of desires. Once a specific desire is fulfilled the agents rest and the system is halted. In this specific context their desire is to classify as many data points correctly as they can. Intense, describes the way the agent intents to fulfil its desires and is formed in plans. Specific plans are designed and followed until the agents fulfil their goals. The intention of every agent in the environment is to maximize the accumulated reward (trust value). The goals are: retrieving training data, agent training, calculating confidence based on the trained data and data prediction. Alternative plans can be designed in case the primary ones do not work. For this thesis only one plan has been designed and implemented: Agent Training

The training process each agent classifier follows depends on the examined problem. In case of a balanced dataset the agent model is optimized based on the performance metric boosted. On the other hand, in a highly imbalanced dataset model optimization process weighs more on precision or sensitivity. In any case, training and hyper-parameter optimization process depends on the characteristics of the dataset examined.

3.3.2 Probability Estimate Normalization

The probability estimates that the agent is providing answers, is one of the first metrics considered to evaluate the validity of the answer given and accept it or reject it. In short terms, the probability estimate is the confidence the agent provides a classification output. Trust measurement is based on the probability estimate during the update process. Normalizing the individual prediction output with the average probability estimation over the training data is essential, since the system updates its trust metric with shorter steps enabling it to learn in more detail. The system normalizes the probability estimate of the outcome by weighting it with the average training probability estimate. Thus, the final estimate that the blackboard must

process is more realistic and has less impact to the update process. Normalization score is calculated using the equation (4):

$$\overline{TrainConfidence_{Ai}} = \frac{\sum_i^n TrainScore_{Ai}}{n}$$

Equation 2 - Average Confidence over Training Data Function

where, n equals with the total number of the hosted agents and $\sum_i^n TrainScore_{Ai}$ is the summation of the estimated output probabilities over the training dataset. During the probability estimate normalization each agent is acting on the dataset which was used to get trained. More specifically, the agents are called to classify each data point from the train set. Then the average probability estimate is calculated and used to reduce the estimate of the predictions from the test set.

3.4 Reinforcement Learning Trust Metric

In chapter 4, we will be discuss how agents act like voters of an electorate by taking part in voting procedures that reflect their opinion either individually or collectively. Blackboard acts as a centralized decision maker that receives the prediction outcomes in order to reach a final conclusion without being able to draw any insights regarding each individual expertise. In this chapter a model free reinforcement learning technique function is utilized that rewards or penalizes each agent regarding the validity of their suggestion and the strength which the experts support their opinion. That way blackboard has the ability now to identify and capture the competence of its KS regarding certain areas of knowledge. Thus, the value extracted from that metric has an immediate impact on the respective vote of the expert.

3.4.1 Trust Measurement for Agents

According to literature (Petridis et al., 1991; Teodorescu & Petridis, 2013) there are many definitions of computational trust in regards to MAS. Commonly, it is defined as the degree of the trustworthiness an entity has towards another in regards to the reliability, truth or the ability to perform a task or how well it can perform a task in a

multi-agent environment (Witkowski & Pitt, 2000). Thus, the major factor for evaluating each expert agent's competence is through the trust score function.

In the beginning the blackboard is uncertain regarding the capabilities of each individual expert and the related performance. The purpose of utilizing the trust metric for the blackboard is to gradually develop objective trust towards its agents by measuring and evaluating the quality of the predicted outcomes without the blackboard getting influenced by biased results.

To that end, we propose a model free method based on Q-learning, as a metric for personalized trust calculation and assertion. In each classification state the algorithm learns an action-utility representation. The value of choosing a class for each observation i is the summation of the trust score in the previous state and the score of the state the agent is currently in. The trust score of each agent is calculated using equation (5):

$$Trust\ (i,a) = \sum \mu \left(r_a^i + \gamma \max_{Pr} \left(Pr(Y = y|X)_a^i \right) \right)$$

Equation 3 - Trust Function

where $Trust\ (i,a)$ represents the trust value with regards to the i_{th} test observation classified by the agent a, where $a = 1, \dots, m.$ total number of agents respectively. $\mu \in [0,1]$ is the learning rate and $\gamma \in [0,1]$ is the discount factor, and $Pr(Y = y|X)_{a,i}^c$ is the probability estimate of the predicted outcome.

3.4.1.1 Trust Measurement Update

After voting procedures finish trust gets updated based on the validity of each individual prediction outcome. The update function described in equation (6) is used during the entire classification process to update the trust value. Trust value is updated according to the performance metric defined. For balanced binary datasets trust update is based on accuracy metric in order to improve. Depending only on the validity of the examined test data observation, the blackboard manager increases or decreases the trust score accordingly by evaluating the reward for the current state and the discounted factor of the probability estimate enhanced by the learning rate μ.

Consequently, trust value based on sensitivity is updated according to the true positive observations, whereas on the other hand, trust based on precision is updated based on the true negative observations. The update function is depicted as follows:

$$Trust_a(i) = Trust_a(i-1) \pm \mu\left(r_a^i + \gamma \max_{Pr}\left(Pr(Y = y|X)_a^i\right)\right)$$

Equation 4 - Trust Update Function

where $Trust_a(i-1)$ is the trust value of the previous state regarding the agent a classifying the observation i. The initialization of the Trust function and its parameters is as follows, $Trust_a(i)t_0 = 0$, $\mu_{\alpha,\ i-1}t_0 = 1$, $r_{a,\ t-1}t_0 = 0$ and $\gamma_{\alpha,\ i-1}t_0 = 0.0001$. The blackboard uses the current state i, reward r and the estimated probability $Pr(Y = y|X)$ to evaluate the performance of each expert. The score values are updated for every knowledge source until the system reaches an equilibrium. i.e. there is no change in the Trust value for ten consecutive epochs. Once the Trust algorithm converges we can use these values to increase the overall accuracy performance.

The update algorithm, which is given by algorithm (4) below, describes the calculation and adaptation process of the trust reward based on the formula described above.

1. Define $\hat{y}, y$
2. Initialize γ factor
3. Initialize r reward signal estimation
4. Initialize Trust reward.
5. **Repeat:**
    a. **Repeat:**
        i. Evaluate $\hat{y}$
            Compare $\hat{y} = Ground\ Truth$
                Estimate r, γ
                Increase Trust
            else, decrease Trust
        Until test observations end
    Until agents end

Algorithm 4 - Trust Reward Adaptation

Trust reward updates are based only on the cluster provenance and the validity of a specific observation related to that cluster. The pseudo code in algorithm (5) describes another update approach employed where the validity of the prediction is also related to the class of the answer. Hence, the blackboard does not only take into account if the prediction is correct or not but also considers the type of the prediction e.g. in case of a binary problem where the $\hat{y}$ is either 0 or 1.

1. Define $\hat{y}, y$
2. Initialize γ factor
3. Initialize r reward signal estimation
4. Initialize Trust reward.
5. **Repeat:**
    a. **Repeat:**
        i. Evaluate $\hat{y}$
            If $\hat{y} = True$ && $\hat{y} = Predicted\ Class$
                Estimate r, γ
                Increase Trust
            else, decrease Trust
        Until test observations end
    Until agents end

Algorithm 5 - Trust Reward Adaptation based on the prediction class

Three trust reward update processes were described in this section. Firstly, the blackboard updated the trust without having any prior information about the problem. Secondly, it considered the provenance of the observation and updated the trust according to the validity of the related observation. Also, the concepts of reward signal estimation using Thomson's sampling and gamma factor optimisation were introduced. Lastly, a nearest neighbour approach was employed enabling the blackboard to adapt dynamically without constructing a model to identify the data patterns from the data. Trust has an immediate impact to the vote each agent has. In the next paragraphs, social score and social welfare functions for voting are described and how trust reward impacts the influence of the specific voting procedures towards the final decision.

3.5 Optimization

The main purpose of the proposed approach is to approximate a global minimum of the defined cost function. Also, speed of convergence into a global minima classification error cost in a reasonable time is a major requirement. In this thesis, two groups of parameters are optimized. Firstly, depending on the performance measure (accuracy, precision, recall), grid search methods are performed for hyperparameters optimization. The second group of parameters are related to the proposed RL trust function. During learning process three are the parameters that need manual configuration by the expert, learning rate μ, observed current reward or reinforcement signal r and the the gamma factor γ. The next two sections describe the optimization process of the two last parameters. Firstly, a novel application of a reinforcement learning technique namely Bayesian randomized sampling or Thomson sampling is employed for updating the current reward r. Then, an algorithm named Simulated Annealing is applied for finding the near optimal values from gamma factor search space.

3.5.1 Reinforcement Signal Optimization

According to (Leslie Pack Kaelbling, 1996), there are many ways a reward function can be defined. The blackboard manager attempts to learn and map the state-action representation for every problem by choosing an action given the current state of

each agent. In every problem there two spaces one of states and one for actions. At the same time, for every actions taken there is a reward assigned which is positive or negative according to the outcome of the action taken. Given a state there is a limited number of actions that maximize the reward. The formal definition of that is as follows:

$$(\mathcal{R} : \mathcal{S} \times \mathrm{A} \rightarrow \mathbb{R})$$

Figure 14 – Markov Decision Process

Where $\mathcal{R}$ is the space where $\mathcal{S}$ reflects the current state of the agent and A the action of given the current state. The action given the state each agent is in, results in $\mathbb{R}$, which is a scalar that reflects the reward of the action taken. For this research, we propose the use of the Thomson Sampling algorithm as a function for updating the rewards r.

3.5.1.1 Thomson Sampling

The CHIMACS reinforcement learning algorithm can be considered as a Multi-Armed Bandit Problem (MAB), thus blackboard updates the reward factor in Trust function using a Bayesian randomized probability matching approach. Each agent is considered as a bandit with a posterior distribution of $p(\vartheta\_a \,|\, r)$ given the observation up to time t. Although that distribution is unknown in the beginning, we set its uncertainty by assuming it has a uniform distribution $p(r \,|\, \vartheta\_a) \sim U([0,1])$ which is the prior distribution. Agent α receives rewards r from Bernoulli distribution $p(r \,|\, \vartheta\_a) \sim B(\vartheta\_a)$. By applying the Bayes Rule, we approach $p(\vartheta\_a)$ by the posterior distribution (Komiyama, Honda, & Nakagawa, 2015):

$$\underbrace{p(\vartheta_a|r)}_{Posterior\ \ Distribution} = \frac{p(r|\vartheta_a)\,p(\vartheta_a)}{\int p(r|\vartheta_a)p(\vartheta_a)d\vartheta_a} \propto \underbrace{p(r|\vartheta_a)}_{Likeli\,hood\ Distribution} \times \underbrace{p(\vartheta_a)}_{Prior\ \ Distribution}$$

Equation 5 - Bayes Rule

At each round *n*, we get a random draw $\vartheta_a(n)$ from the posterior distribution in equation (7), for each agent *a.*

$$p(\vartheta_a|r) \sim \beta(N_a^1(n) + 1 \mid N_a^0(n) + 1)$$

where $N_a^1(n)$ and $N_a^0(n)$ is the number of times the agent a got reward 1 and 0 accordingly depending on the validity of their prediction.

Usually, at each round n the algorithm selects the bandit with the highest $\vartheta_i(n)$. However, the random draw output from the distribution is integrated as the reward signal in Trust in (2). This way the system can adapt the reward signals with regards to the expertise of every agent in any knowledge area. At the same time, empirical evaluation showed faster convergence using discounted TS future rewards. The reward assignment using TS and reward gradient over states is formulated as follows:

$$r_a^i = \begin{cases} \vartheta_a(n)/i^2, & correct\ prediction\ at\ i \\ -\vartheta_a(n)/i^2, & incorrect\ prediction\ at\ i \end{cases}$$

3.5.2 Gamma Factor Optimization

The gamma parameter acts as a weight factor in regard to the probability estimate provided by the agent classifiers. The gamma factor constitutes a search space that includes trust functions for all the agents and the knowledge source of their expertise. To that end, an optimization algorithm named simulated annealing was applied. SA is a descent general purpose algorithm whose main advantage is that is able to escape from local minima by random ascent moves. The annealing algorithm aims to minimize a cost function. The solutions by this technique are close to the global minimum(Kirkpatrick, Gelatt, & Vecchi, 1983). The optimization of gamma

parameter is the key to minimize the convergence time and get a near optimal minimal error rate.

3.5.2.1 Simulated Annealing

Annealing is a probabilistic technique for approximating the global optimum of a given cost function. It simulates a physical/chemical process during which when tempering certain alloys of metal, glass, or crystal by heating above their melting point and then cooling it very slowly until it solidifies into a perfect crystalline structure, it produces high-quality materials. The simulation of this process is known as Boltzmann or simulated annealing (SA) (Černý, 1985; Kirkpatrick et al., 1983). Global minimum energy configuration reflects the defect-free crystal state corresponds of the material. For this thesis, the analogy of SA with gamma parameter optimization procedure is as follows. The states of the physical material correspond to classification solutions (Accuracy, Recall, Precision), the energy of a state to cost of a solution depicted in equation (6) and the temperature to a control parameter.

$$e = 1 - Cost_{new}$$

Equation 6 - Classification Cost Function

The Metropolis algorithm is a simple method for simulating the evolution to the thermal equilibrium of a solid for a given temperature (Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953). SA is a variant of the Metropolis algorithm (Kirkpatrick et al., 1983), where the temperature is decreasing. SA is based on two stochastic processes: The first process is for generating solutions and the other for the accepting or rejecting these solutions.

The probability P_a of a physical system, being in state a with energy E_a at temperature T satisfies the Boltzmann-Gibbs distribution.

$$E_a = \frac{1}{Z} e^{\frac{-E_a}{k_B} T}$$

Equation 6 - Energy Function

where k_B is the Boltzmann's constant, T is the absolute temperature and Z is the partition function, defined by:

$$Z = \sum_{\beta} e^{-\frac{E_\beta}{k_B} T}$$

Equation 7 - Partition Function

the summation being taken over all states β with energy E_β at temperature T . Regardless of the energy. At high temperatures, the Boltzmann distribution exhibits uniform preference for all generated states. When T approaches zero, only minimum energy states have nonzero probability of occurrence.

SA, theoretically guarantees that CHIMACS will reach a global minimum with high probability. T is the artificial temperature parameter which is reduced over time and controls the number of the iterations of the energy function $E(x)$. The probability of a state change is determined by the Boltzmann distribution of the energy difference of the two states:

$$P = e^{-\frac{\Delta E}{T}}$$

At high temperatures, the algorithm attempts to improve on by occasionally taking a risk to accept a worse solution. SA is described below in algorithm (6) (Kirkpatrick et al., 1983).

1. Initialize $step$
2. Initialize γ population with random values increasing by $step$.
3. Initialize T with a large enough temperature.
4. Initialize threshold THR.
5. **Repeat:**
    a. **Repeat:**
        i. Apply random perturbations to the state $\gamma = \gamma + \Delta\gamma$
        ii. Evaluate the energy $\Delta E(\gamma) = E(\gamma + \Delta\gamma) - E(\gamma)$:
        if $\Delta E(\gamma) < 0$, keep the new state;
        else, accept the new state with probability $P = e^{-\frac{\Delta E}{T}}$.

        until accepted transitions < THR
    b. Set $T = T - \Delta T$
    Until T is close to 0

Algorithm 6 - Simulated Annealing

The population of the gamma values is predefined into a range of $\gamma \in [-40, 40]$ with the step defined at $step = 0.000001$. Temperature is initialized at $T = 10000$, while the threshold's value depends on the time we want to spent letting the system learn.

The rate at which Temperature T is reduced is critical to the efficiency of SA. If the reduction rate of T is too slow, the algorithm takes too long to converge, on the other hand, if the reduction rate is too fast SA is probably prematurely converging to a local minimum. Based on (Geman & Geman, 1984) T should be decreased based on a cooling schedule that ensures the convergence to a set of globally minimum cost states with probability one, given that the initial temperature is large enough. The cooling rate is defined in equation (10) below:

$$T(t) \geq \frac{T_0}{ln(1+t)}, t = 1,2,3, \ldots$$

Equation 8 - Temperature Cooling Rate

According to (Hajek, 1988), given enough time and a logarithmic reduce rate, simulated annealing is proved to converge with high probability to the global

minimum. However, it is too slow and practically a reduce rate described in equation (11) `ensures a near optimal solution in a good amount of time.

$$T(t) = a(t-1), 0.85 \leq a \leq 0.96$$

Equation 9 - Practical Cooling Rate

Simulated annealing is particularly effective when applied to combinatorial or discrete problems (Nikolaev & Jacobson, 2010). Although it is not guaranteed that it will find the best optimum, it will converge to a near optimal state.

3.6 Impact of Voting in the Overall Predictive Performance

From a social choice theory point of view, CHIMACS can be considered as a "society" (Endriss, 2011, 2013) with a finite set of ML agents as its electorate or voters $N = 1,2, \ldots, n$ voting over an finite set of alternatives $A = \{1,2, \ldots m\}, m \geq 2$. Each agent $i \in N$ holds a quasi-order Pi (ranking of the alternatives) over A, where transivity, totality, reflexivity and anti-symmetry are satisfied (Brandt et al., 2016). The set of all quasi-orders is denoted as $L = LA$, so for $i \in N$, $\mathcal{P}^i \in \mathcal{L}$ throughout. Thus, for each ML agent a preference rankings profile vector $i \in N$ is constructed.

In this section, the impact computational voting has on the improvement of the overall classification accuracy will be discussed. Agents are the electorate, where each one of them is entitled with a vote over what it believes is the solution to the examined problem. This section describes five different voting configurations based on majority voting rule, where each individual knowledge source assigns a vote in different forms and the decision is made based on those votes. Every vote impacts differently on blackboard's final choice and thus the result of the overall accuracy.

3.6.1 Prediction utilizing Voting Processes

In this research, a variety of social scoring functions (SCF) and social welfare functions (SWF) have been used but only the ones that produced significant increase in the overall prediction performance have been described. According to May's Theorem (Endriss, 2014), considering two alternatives and any number of voters, majority rule declares as winner the most dominant between the two alternatives.

Knowledge aggregation from heterogeneous KS is one of the core concepts of CHIMAS framework. Agents participate in different variations of voting procedures that allow them to express their opinion and suggest a solution to the currently examined problem. The blackboard controller utilizes voting procedures to aggregate the acquired knowledge that later considers to conclude to a decision.

Voting is separated into two broad categories, the majority and the judgment aggregation voting. On the one hand majority voting nominates an individual knowledge source as dominant and considers only its opinion for the result. On the other hand, judgment aggregation voting sums up the votes and utilizes the collective results to decide about the final solution. In the next paragraphs, different voting configurations are formally described.

3.6.2 Voting Mathematical Formulation

Following the classic example of political elections, where every political candidate standing for election is an alternative and voters express their preferences on the ballot sheet. In the same way, a given set of available alternatives between a solution, a group of voters (KS) and their preferences over these alternatives, voting is a framework for choosing a best alternative.

Social choice theory provides a simple mathematical (Milgrom∗, 2004) framework for modelling the process of voting. Following the formal framework, CHIMCAS voting process formulation is represented as showed below:

- Electorate - A finite set of voters:

$$Knowledge\ Source\ = \{v\ \in \mathbb{N}^{+}\}$$

- A finite set of alternatives:

$$Alternatives = \{a\ \in\ \mathbb{R}\}$$

- A finite set of votes:

$$Vote = \{g\ \in\ \mathbb{R}\}, \mathbb{R}_{\geq 0}$$

The mentality behind voting is to take into consideration every expert's opinion, exploit and aggregate their knowledge which in the end results into a higher quality solution. The next paragraph describes in detail different configurations of voting CHIMAS utilizes over an execution cycle.

3.6.3 Simple Majority SCF – Hard Voting

Simple majority social choice function voting lets the experts express their opinion and according to their estimations their vote counts for one unit. Let $a_1, a_2, a_3, \dots, a_n$ denote the set of n number of answers provided by the KS hosted in the agent repository regarding an examined problem. Simple voting procedure is based on the plurality rule that counts the number of each individual answer and sums them up. For each answer the agent takes one vote that counts as one. Then blackboard collects the results, compares them and decides which alternative has the most votes. After each voting cycle finishes, it validates the final result and updates the accuracy over this procedure accordingly. The vote with the higher value wins. Hard voting is the simplest case of majority voting. The mathematical formulation of simple voting procedure is described in equation (12):

$$\hat{y} =\ mode\{C_1(x), C_2(x), \dots, C_n(x)\ \}$$

Equation 10 - Simple Majority SCF Voting

Assuming three classifiers are combined to classify a training sample as follows, $C_1(x) = \{0\}, C_2(x) = \{0\}$ and $C_3(x) = \{1\}$, then via majority vote, we would classify the sample as $P_f = \{0\}$.

3.6.4 Simple Majority SWF with Probability Estimate – Soft Voting

The probability estimate of the currently predicted outcome resembles the opinion and the vote of each knowledge source. Blackboard proceeds with the voting procedure during which the process is similar to a tournament selection and the answer is that the higher vote wins. Using uniform weights for w_j, we compute the average probabilities utilizing function in Equation (13):

$$\hat{y} = \arg\max_i = w_j Pr(Y = y|X)_i^j$$

Equation 11 - Simple SWF with Probability Estimate

Assuming the example in the previous section was a binary classification task with class labels $i \epsilon \{0,1\}$ the ensemble could make the following prediction, $C_1(x) \rightarrow \{0.9,0.1\}, C_2(x) \rightarrow \{0.8,0.2\}$ and $C_3(x) \rightarrow \{0.4,0.6\}$, $w_j = 1$:

$$\hat{y} = \arg\max_i [\, P(i_0|x), P(i_1|x)] = 1$$

In algorithm (7), the pseudo code describes simple SCF voting process utilizing confidence as the base for measuring agent's competence.

1. Initialize vote for $Agents_n$
2. Predict $test\ data\ observations$.
3. **Repeat:**
    a. **Repeat:**
        i. $Agent_i\ Alternative_1 = Porb.Estimate_1$
        ii. $Agent_i\ Alternative_2 = Porb.Estimate_2$

    until Voting End.
    b. Validate $Agent_i\ response\ Alternative_1$, $Agent_i\ Alternative_2$

Until Test Observations End.

Algorithm 7 - Majority SCF Pseudo code

Majority voting SCF declares a single alternative as the winner, whereas in judgment aggregation SWF every vote has its impact to the outcome. Regardless the structure of the voting process both functions provide an output that is feed into equation (14) where the vote with the highest value is the winner.

$$Winner\ Vote = \underset{\mathcal{P}^i}{\operatorname{argmax}}\, \mathcal{P}^i$$

Equation 12 - Dominant Vote

Literature and experimentation indicate that weak experts provide biased results with misleading confidence levels. Blackboard is responsible for handling the produced bias and enhancing the evaluation metrics. The next chapter introduces a trust measurement based on reinforcement learning techniques that enhances the voting processes. This metric enables blackboard to capture and exploit each individual expert's capabilities and influence the impact each vote has during voting process by observing individual classifier's performances.

3.6.5-Majority SWF – Aggregative Voting with Probability Estimate

Majority social welfare function in this setting sums up the votes regarding the alternatives offered as a solution to a problem. However, this time voting is related to

the summation of the probability estimation provided by the agents. The experts do not compete each other, but rather contribute to the solution as a team. The formula in equation (15), depicts the majority SWF, that aggregates the confidence of the provided answer. The alternative with the highest aggregated vote is declared winner by this voting procedure.

$$\hat{y} = \arg\max_{i} \sum_{j=1}^{m} w_j Pr(Y = y|X)_i^j$$

Equation 13 - Simple SWF with Probability Estimate

Assuming the example in the previous section was a binary classification task with class labels $i \epsilon \{0,1\}$ the ensemble could make the following prediction, $C_1(x) \rightarrow \{0.9,0.1\}, C_2(x) \rightarrow \{0.8,0.2\}$ and $C_3(x) \rightarrow \{0.4,0.6\}$, $w_j = 1$:

$$P(i_0|x) = \frac{0.9 + 0.8 + 0.4}{3} = 0.7$$

$$P(i_1|x) = \frac{0.1 + 0.2 + 0.6}{3} = 0.3$$

$$\hat{y} = \arg\max_{i} \ [P(i_0|x), P(i_1|x)] = 1$$

1. Initialize vote for $Alternative_1, Alternative_2$
2. Predict $test\ data\ observations.$
3. **Repeat:**
    a. **Repeat:**
        i. $Alternative_1 = Alternative_1 + Porb.Estimate_1$
        ii. $Alternative_2 = Alternative_2 + Porb.Estimate_2$
    until Voting End.
    b. Validate $Alternative_1, Alternative_2$

Until Test observations End.

Algorithm 8 - Majority SWF Pseudocode

3.7 Conclusion and Discussion

Collective intelligence is of major importance that reflects the diversity in capabilities both in peoples and agent's society. A number of societal voting methods were transferred in a computational form and reinforcement learning functions were constructed in order to explore and exploit expert's knowledge (Procaccia, 2008). In this Chapter, first, an intelligent agent structure was described, then five social choice and welfare functions were formally described as voting procedures which the framework employs. Lastly, an adaptive metric that captures the predictive performance is introduced. Blackboard utilizes that metric and enables it to dynamically identify each agent competence by adapting in every epoch. Two version of reinforcement trust measure were applied, the first adapts based only on the validity of the test observations. The second takes into account the depended variable $\hat{y}$, too. Blackboard combines and utilizes all these tools to boost the overall predictive performance.

To sum up, this Section introduced the core modules of the proposed multi-agent framework, then it proposed two trust metrics based on reinforcement learning methods and three different social and welfare voting methods. Combined all together the system can increase the overall predictive performance on different problem domains.

3.8 Summary

A novel general classification framework, CHIMACS, was proposed in this Chapter. The framework adopts and utilizes a number of societal tools able to capture agent and problem properties and fuse the acquired knowledge. The patterns emerge from the data represented in knowledge areas each expert performs. Their performance and relationship with the knowledge areas is facilitated through reinforcement learning trust metric by exploring and exploiting. The concept of computational trust has been introduced along with the calculation and update process. Two optimization techniques have been described which have been used for optimizing essential parameters of the trust function (Nikolaev & Jacobson, 2010). The applicability of the framework will be demonstrated in the next chapters using a number of different datasets. In the next Chapter, three different versions using different blackboard structures and two voting function influenced by reinforcement trust are discussed.

Chapter 4 Blackboard Reasoning & Expertise Reconnaissance

In this Chapter, a variety of implementations of the proposed framework are explored. Three conceptually different learning modes for training the blackboard, a static, a knowledge area aware and an adaptive knowledge area learning mode are proposed. Two update approaches for the trust measurement are also proposed. Finally, a probability estimate calculation method of the k-nearest neighbours is presented. As a result, an intelligent blackboard based system that utilizes reinforcement learning techniques for boosting the predictive power is proposed.

To begin with, a brief introduction on pattern identification from the data is presented in Section 4.1. Section 4.2, describes the trust function along with the pseudocode of the naïve version of CHIMACS. Secondly, the impact of pattern discovery in the data used unsupervised methods have in trust measurement and voting procedures is discussed in Section 4.3. The resulting knowledge area aware approach combines the information related to the extracted patterns from the data and the provenance of the observations with dedicated trust calculation and voting procedures in order to enhance the overall predictive performance. Proceeding, the last implementation of the proposed framework is described. A component that enables blackboard to dynamically identify patterns from the data on the fly is introduced in Section 4.4. Next, Section 4.5 describes the update methods for the trust measurement according to the enhanced performance metric. Finally, Section 4.6 and Section 4.7 provide a conclusion and a summary of the Chapter.

4.1 Introduction

Ensemble classification is considered as the aggregation of predictions of multiple classifiers with the goal of improving prediction performance.

EM became popular as a relatively simple device to improve the predictive performance of a base procedure. There are different reasons for this: the bagging procedure turns out to be a variance reduction scheme, at least for some base procedures. On the other hand, boosting methods are primarily reducing the (model) bias of the base procedure. This already indicates that bagging and boosting are very different EM. We will argue in sections 4.1 and 4.7 that boosting may be even viewed as a non-ensemble method which has tremendous advantages over ensemble (or multiple prediction) methods in terms of interpretation.

There is extensive research on revealing and exploiting patterns from data using reinforcement learning methods in order to increase the overall classification performance, such as adaptive agent-based heterogeneous CBR framework (Teodorescu & Petridis, 2013) and Q-learning agent-based multi-agent system for data classification. These approaches employ different homogeneous KS and experts to increase the global predictive performance. However, there are some shortcomings in such approaches. Firstly, both frameworks were trained homogeneous learning models. Secondly, at the end of the training each framework became as good as its best trained.

The first shortcoming, training a pool of the same type of learning agents, although blackboard did manage to recognize agents' expertise, does not produce much diversity in terms of capabilities. Individual classifiers members that constitute a type of ensemble need to meet a necessary and sufficient condition need to be accurate and diverse (Zhou et al., 2002). Even though different hyper-parameters where used to train the models, the ML algorithm remained the same. Thus, there is high probability the errors made by the same type of classifiers are correlated.

In this Chapter, the proposed framework provides flexibility and scalability where a diversity of different heterogeneous and homogeneous models and training sets can be used. The system, which is coordinated by the blackboard controller, is able to partition the data into homogeneous groups of observations into distinct areas which

share similar properties. The blackboard utilizes reinforcement learning trust reward method in order to capture the expertise area of each agent by evaluating the agent's competence. Thus, the controller after a number of iterations is more intuitive in regard to a more individualized task assignment plan. Utilizing dedicated trust calculation methods and voting procedures the blackboard is capable of capturing the relationship between the expert's capabilities and the knowledge areas properties.

The second shortcoming of the proposed frameworks is that of capturing the characteristics of the best trained agent only. That is reasonable since the agent with the best performance feels very confident and overshadows the rest. However, that is not ideal since opinion diversity is essential. In this chapter, the proposed framework captures the opinions of every expert and combines them in an effective way. As a result, the proposed framework allows the combination and exploitation of different and diverse capabilities in such a way that increases the overall predictive performance.

4.2 CHIMACS

Figure 20, depicts the reasoning scheme of the basic version of CHIMACS. The information the blackboard receives from the environment is restricted. The blackboard has no prior knowledge about the nature of the examined problem or about the agent's capabilities and expertise. The only input it receives is the individual predictions along with their estimated probability. Thus, the agents are treated as black-boxes where the blackboard is called to distinguish which are the diverse capabilities and exploit the existing knowledge.

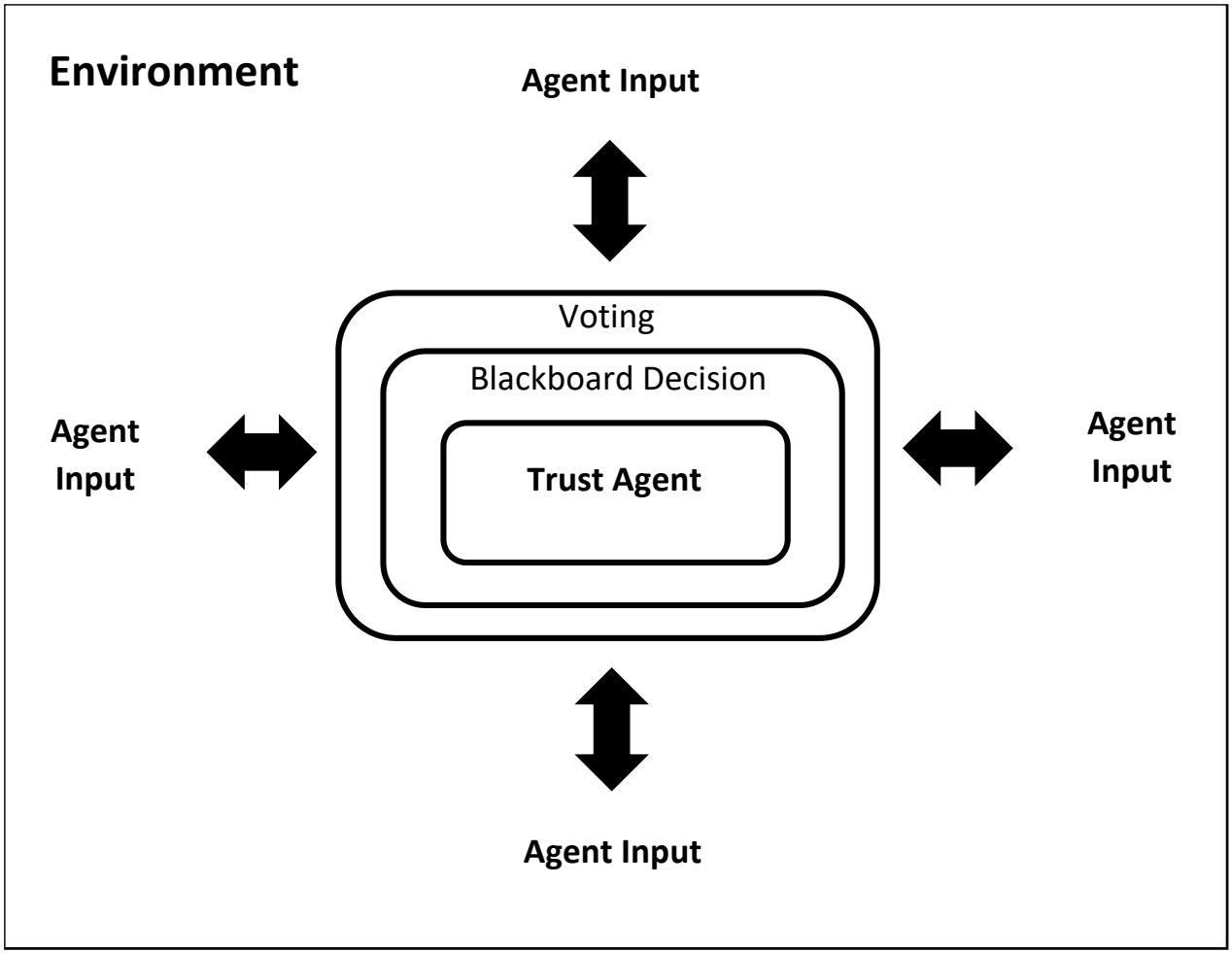


Figure 15 - Simple CHIMACS Reasoning Scheme

This is the first version of CHIMACS and is considered as the basis with which the rest of the implementations are compared to. The validity of the answers the agents provide is the only criterion the trust value is updated. In case of a correct or wrong answer, the trust measurement is updated according to the equation (6) in section 3.4.1.1. As the process begins it attempts to identify the capabilities and the expertise of each individual. Also, this version is focused to boost only accuracy performance metric and did not care about the rest of the performance metrics e.g. specificity and sensitivity. Thus, the framework deals only with problems that there is balance between the depended feature elements.

4.2.1 Trust Measurement and Voting

Given the predicted $\hat{y}$ from the entire agent group, the final decision is derived from voting procedures. Voting formulas are applied and they are only related to the predictions collected. Blackboard does not have any insight regarding the agent's structure, capabilities nor the nature of the problem. Trust is introduced as a new parameter in voting functions that represents the system's knowledge regarding an agent expertise. The system draws insights related to agent's performance and attempts to boost the overall predictive power. Trust value influences the voting procedure acting as a weight factor that affects the individual agent vote by enhancing or reducing the votes' importance. Trust towards an agent is updated only through the validity of the response given. Consequently, the final choice of the system is affected as the value of the trust is adapted during the training process. Pseudo code in algorithm (9) describes the above process in more detail.

Algorithm CHIMACS

Input: agents, hyper-parameters of agents, number of clusters training samples, test samples, gamma population, gamma initial values, reinforcement signal, trust initial values.
Output: Performance indicators

Determine initial trust, reinforcement signal and gamma parameters.

For each agent:
For each training sample:
Calculate Normalization Value
For each agent:
For each training sample:
Normalize Est. Probability Value

For each test sample:
For each agent:
Predict class a_i^a , $Pr(Y = y|X)_{a,i}^c$
Initialize Simple SWF Voting
If $a_i^a = 0$ $vote_0 += 1$ **Else** $vote_1 += 1$
Initialize SWF Voting with Prob. Est.
If $a_i^a = 0$ $vote_0 += Pr(Y = y|X)_i^a$ **Else** $vote_1 += Pr(Y = y|X)_i^a$
Initialize SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Pr(Y = y|X)_i^a$
Initialize Trust SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Trust_a * Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Trust_a * Pr(Y = y|X)_i^a$
Initialize Trust SWF Voting with Prob. Est.
If $a_i^a = 0$ $vote_0 = Trust_a * Pr(Y = y|X)_i^a$ **Else** $vote_1 = Trust_a * Pr(Y = y|X)_i^a$
End for
Winner SCF Voting = The agent with the highest vote.
Winer SWF Voting = The alternative with the highest vote.
Update Trust
Update Confusion Matrix
End for

Algorithm 9 - CHIMACS Pseudo Code

4.3 Clustered CHIMACS

C-CHIMACS utilizes unsupervised learning methods to capture meaningful and useful patterns in the data. Blackboard is aware of the extracted patterns which then used as knowledge areas from the system to identify the expertise of each of each KS. Voting process is related the identified knowledge areas. Figure 22 depicts the reasoning scheme under which the clustered CHIMACS operates. This process works with observed input patterns x_i, which are often assumed to be independent samples from an underlying unknown probability distribution $P_I[x]$, and some explicit or implicit a priori information as to what is important. Hence, knowledge area identification is a process that aims to analyse the data and extract information that describe the relationship among the data observations inside a dataset ("Cluster Analysis," 2007). The analysis of the relevant knowledge areas of the dataset is a descriptive task performed by the blackboard controller itself.

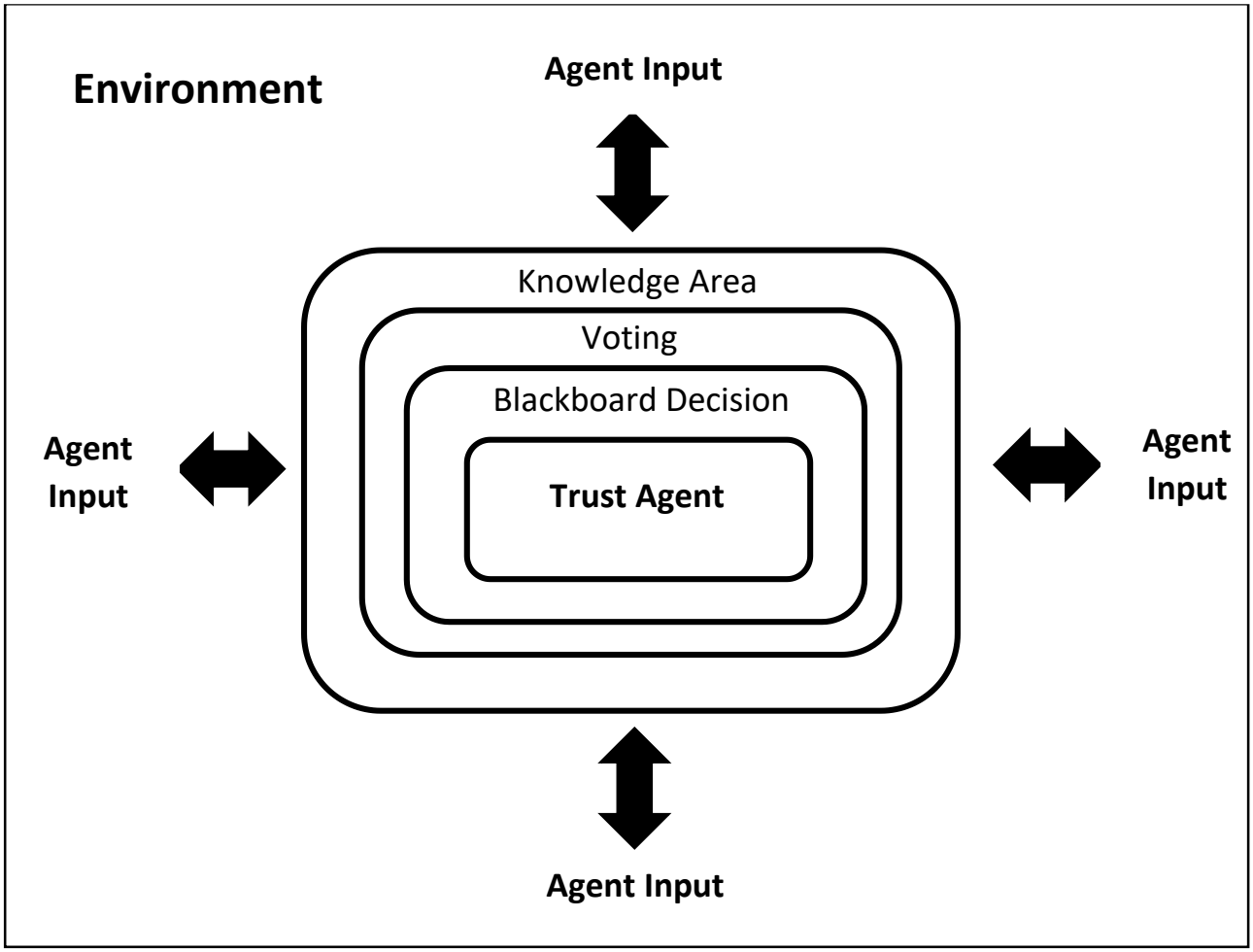


Figure 16 - C - CHIMACS Reasoning Scheme

4.3.1 Knowledge Extraction

Cluster analysis is one of the unsupervised techniques used to analyse and capture the underlying relationships among the data observations. In more detail, the K-Means algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields.

That task seeks to identify homogeneous group observations to partition the data into distinct areas which share similar properties. Knowledge areas or patterns are the

terms assigned to these groups since in later section they are going to be used as the area of expertise for each knowledge source individually. Clustering algorithms are the mechanisms employed for this task by the blackboard. K-means is one of the algorithms responsible for the identification of the different knowledge patterns in CHIMACS.

The k-means algorithm divides a set of N Samples X into K disjoint clusters C of equal variance by minimizing a criterion known as the inertia or within-cluster sum-of-squares, where each cluster is described by the mean m_j of the samples in the cluster. The means are commonly called the cluster *"centroids"*. Note that they are not, in general, points from *X*, although they live in the same space. The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum of squared criterion in equation (16) ("Cluster Analysis," 2007).

$$argmin_{\{r_{nk},\mu_k\}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

Equation 14 - K-Means Objective Function

where x_n is the value of variable *n, r* is the weight and μ is the mean of the variable over the examined cluster (Mooi & Sarstedt, 2011). This implicit objective function above measures the sum of distances of observations between their knowledge areas (cluster) centroid and is called Within-Cluster-Sum-of Squares (WCSS). WCSS is a measure of variation within a knowledge area and thus, minimizing the WCSS is the methodology that provides valuable information in regard to the optimal number of different knowledge areas.

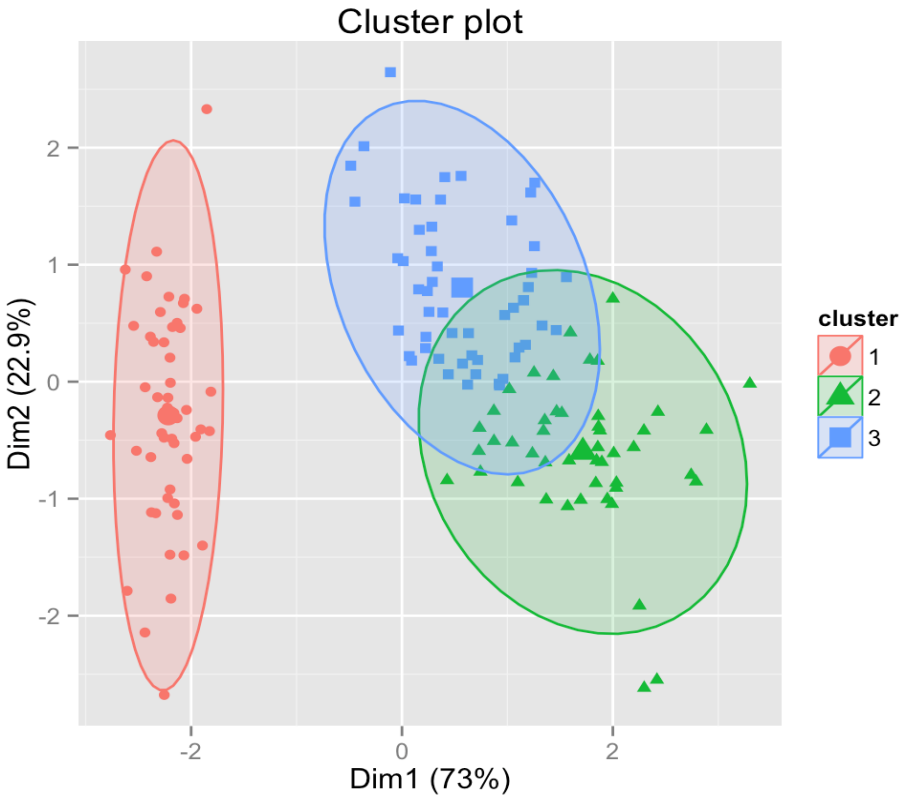


Figure 17 - Knowledge Areas (Hofmann, 2001)

In figure 23, is an example of discrete clusters of the Iris dataset (Lichman, 2013). The knowledge areas are determined through an iterative process that tries to minimize the distances between the candidates that have common characteristics as described above. Silhouette analysis is performed in order to determine the optimal number of clusters ("Cluster Analysis," 2007). Blackboard is now aware of the patterns that emerge from the data and is capable of performing dedicated voting and trust update.

4.3.2 Trust Measurement and Voting

Trust still updates according to the validity of each data observation. However, now the blackboard is aware of the provenance of the observation and updates the measurement for the specific agent in regard to the knowledge area the observation belongs to. The following equation describes the trust update function in regard to a cluster, where c denotes the cluster assignment:

$$Trust\ (i)_a^c = Trust_a^c(i-1) \pm \mu\left(r_a^{i,c} + \gamma \max_{Pr}\left(Pr(Y=y|X)_a^i\right)\right)$$

Equation 15 - Clustered Trust Update

The knowledge areas affect the way trust is updated. Hence only the voting processes that are influenced from trust parameter are described in this section. Majority SCF and SWF voting formulas that are not influenced by trust measurement are applied as described in sections 3.6.3, 3.6.4,3.6.5. The rest are described below. Majority SWF and SCF functions are described in equations (18) and (19), where c is the parameter that reflects a specific cluster and influenced by the trust factor.

$$Majority\ Vote\ A_{n,a,c}\ SWF = \sum_{i=1}^{n,a,c}\left(Pr(Y=y|X)_i^{a,c} * Trust_a^c(i-1)\right)$$

Equation 16 - Clustered Majority SWF

$$Majority\ vote\mathcal{A}_{i,a,c}\ SCF = Pr(Y=y|X)_i^{a,c} * Trust_a^c(i-1)$$

Equation 17 - Clustered Majority SCF

In both cases, current vote value is calculated based on the previous trust state, where the value equals to, $Trust\ (i-1,a)$. Utilizing trust, blackboard is now based on the objective opinion it has drawn observing the performance of the experts and is capable of evaluating the quality of the vote in order to conclude to a final result. Thus, the clusters represent the diversity of the data patterns, which the agents act on. The blackboard monitors the performance for each cluster and learns it through the computation of the trust value. The process is described in more detail in Algorithm (10).

Algorithm C - CHIMACS

Input: agents, hyper-parameters of agents, number of clusters training samples, test samples, gamma population, gamma initial values, reinforcement signal, trust initial values.
Output: Performance indicators

Determine initial trust, reinforcement signal and gamma parameters.

For each training sample:
Assign Cluster

For each agent:
For each training sample:
Calculate Normalization Value
For each agent:
For each training sample:
Normalize Est. Probability Value

For each cluster:
For each test sample:
Assign cluster C_a^i
For each agent:
Predict class a_i^a , $Pr(Y = y|X)_{a,i}^c$
Initialize Simple SWF Voting
If $a_i^a = 0$ $vote_0$+= 1 **Else** $vote_1$+= 1
Initialize SWF Voting with Prob. Est.
If a_i^a = 0 $vote_0$ += $Pr(Y = y|X)_i^a$ **Else** $vote_1$+= $Pr(Y = y|X)_i^a$
Initialize SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Pr(Y = y|X)_i^a$
Initialize Trust SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Trust_a^c * Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Trust_a^c * Pr(Y = y|X)_i^a$
Initialize Trust SWF Voting with Prob. Est.
If $a_i^a = 0$ $vote_0 = Trust_a^c * Pr(Y = y|X)_i^a$ **Else** $vote_1 = Trust_a^c * Pr(Y = y|X)_i^a$
End for
Winner SCF Voting = The agent with the highest vote.
Winer SWF Voting = The alternative with the highest vote.
Update Trust
Update Confusion Matrix
End for
End for

Algorithm 10 - C-CHIMACS Pseudo Code

4.4 Dynamic CHIMACS

Uncovering distinct knowledge patterns from the data can facilitate the blackboard learning process towards an optimal exploitation of the knowledge each agent process. Indeed, utilizing unsupervised ML methods to extract such information is very important for the learning process. However, there is a shortcoming using unsupervised methods. More data observations require an expert to train the model again and evaluate its performance. Thus, the blackboard relies on a component that is practically static. Part of its reasoning mechanism is not dynamic and flexible enough to adapt to the environment changes. The problem addressed in this Chapter is how to identify data patterns dynamically based on k-nearest neighbour's algorithm that is capable of updating and adapting in every execution cycle. Figure 30 depicts the reasoning scheme under which the clustered D - CHIMACS operates.

First, a brief introduction on nearest neighbours and pattern identification using k number of similar observations is introduced in Section 4.4.1. The proposed adaptive approach to dynamically extract useful patterns and dynamic knowledge areas is discussed in Section 4.4.2. The process for calculating the probability estimates of the retrieved neighbours is presented in Section 4.4.3. Section 4.4.4, describes the trust calculation and voting processes based on k-Nearest Neighbours. Finally, Section 4.6 and Section 4.7 provide a conclusion and a summary of the Chapter.

4.4.1 Structure

In chapter 3, the concept of static knowledge areas identification as homogeneous groups with similar properties and the unsupervised tools used for revealing such patterns from the data were described. Also, the importance of the impact of such procedures and how they were used to help increase the collective prediction performance were also described. In this context, the utilization of unsupervised learning techniques for extracting meaningful patterns is closely related with model creation. That, in the future the model is going to need retraining by an expert. However, despite its usefulness in the collective classification process, the lack of

adaptability is the major drawback in this process since in case of additional incoming data the model needs reconstruction.

To that end, CHIMACS employs a more sophisticated meta-reasoning approach to dynamically capture the underlying trends in knowledge patterns using k-nearest neighbour approach that is adapted during the process. Blackboard can retrieve k-nearest most similar cases from the dataset and construct the essential knowledge area based on the predefined number of number of neighbours. Figure 24 presents the reasoning scheme of D-CHIMACS.

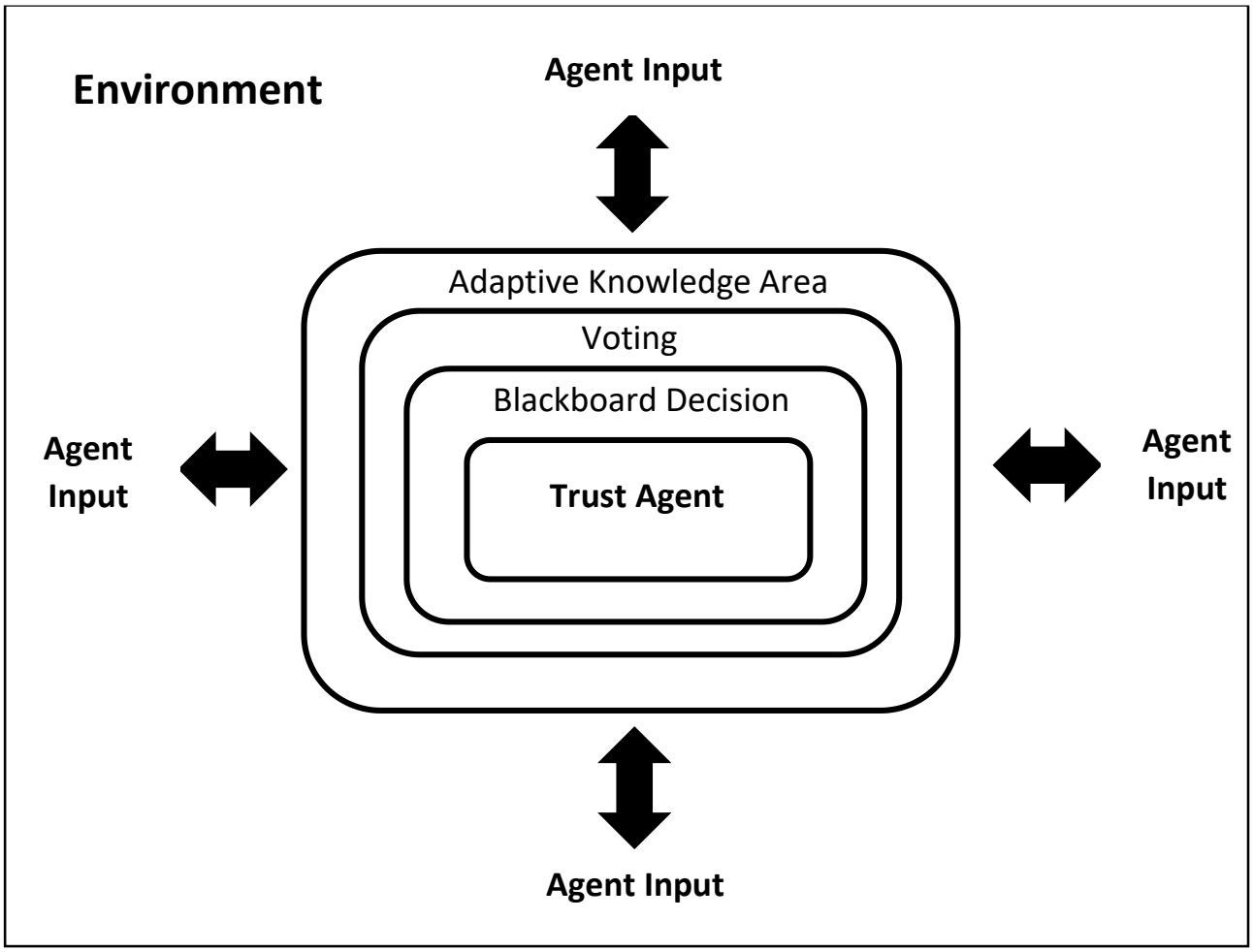


Figure 18 - D-CHIMACS Reasoning Scheme

4.4.2 Adaptive Knowledge Extraction

For this version of knowledge pattern recognition task, blackboard utilizes the k-nearest neighbour's algorithm in an out-of-the-box manner. In most of the cases a supervised ML algorithm is trained to approximate a function $h : \mathcal{X} \longrightarrow \mathcal{Y}$, so that given an unseen observation $\mathcal{X}$,$h(x)$ can confidently predict the corresponding output $\mathcal{Y}$. Thus, a datum object is classified by a majority vote of its neighbours considering the class membership, with the object being assigned to the class most common among its k nearest neighbours. Equation (20), estimates the class conditional probability:

$$P(y = j|X = x) = \frac{1}{K}\sum_{i \in A} I\left(y^{(i)} = j\right)$$

Equation 18 - Class Conditional Probability

where $I(x)$ is the indicator function which evaluates to 1 when the argument x is true and 0 otherwise (Murty & Devi, 2011).

In this case, we only care to capture the relationship between $\mathcal{X}$ and $\mathcal{Y}$ and retrieve the predefined $\mathcal{K}, \mathcal{K} \in \mathcal{Z}^+$ nearest neighbors from the feature space and stop before the final prediction outcome. The flow chart in figure 25 describes the dynamic CHIMACS process.

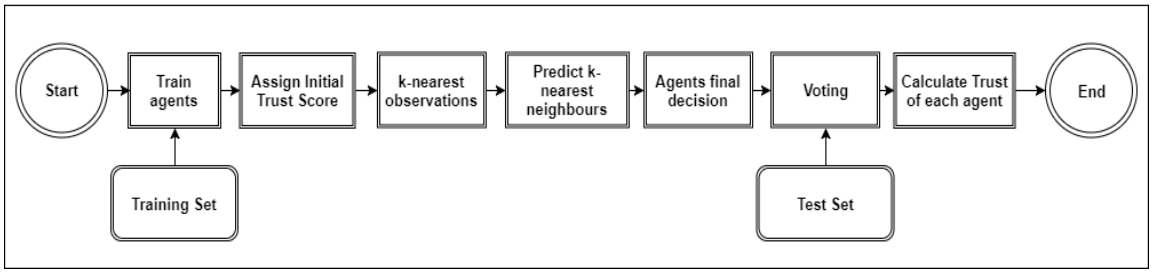


Figure 19 - D-CHIMACS Flowchart

CHIMAS framework uses an enhanced k-nearest Meta-Reasoner version controller as its backbone. The structure is based on a customized k-nearest neighbour

algorithm that updates its knowledge dynamically in every execution cycle and can learn and reason based on the old and new acquired knowledge.

The main advantage of the enhanced version of blackboard controller is that it does not need to build a new model every time a new experience is saved. In simple terms, due to its structure it can process any type of problem and adapt to it. In the previous approach, probability estimate $Pr(Y = y|X)_a^i$ was taken into consideration individually after the agent's classifier prediction outcome. In this case, the probability estimate calculation follows an entire different process that is described in the following sections of this chapter. In this point, it is essential to state that k-NN algorithm is not used for prediction purposes. Its only use is for retrieving groups of data observation for the data repository that share similar attributes.

4.4.3 K-Nearest Neighbour Probability Estimate Calculation

Probability estimate calculation is performed in an unconventional manner. As mentioned earlier, k-NN is not employed for prediction purposes thus we only care for the k data observations. The retrieved data observations are delivered from the blackboard controller to the classifiers and the prediction probability estimates are returned to the blackboard. Following the process described, the equation (21) describes the calculation of the probabilities estimates the prediction outcomes in case of a binary problem for each individual expert:

$$\mathcal{E}_a^i = \begin{cases} \sum Pr(Y = y|X)_a^i, & y = 1 \\ \sum Pr(Y = y|X)_a^i, & y = 0 \end{cases}$$

Equation 19 - Probability Estimate Summation

where $1 \leq \mathcal{E}_a^i \leq 0$ is the aggregated value of the returned probabilities estimates $Pr(Y = y|X)_a^i$, of the prediction outcomes and $y \in [0,\infty)$ is the class label resulted from the agent classifiers.

Retrieving the optimal number of closest data observations is a process that as a hyperparameter needs optimization. However, probability estimate calculation is closely related to the results obtained by the agent experts and not related to the

previous process. A specific number needs to be defined using conventional hyperparameter optimization techniques that depend entirely on the data, but since it's not a classification procedure the k-NN component will not provide us with a final classification outcome. The next section describes the trust measurement calculation based on the knowledge area acquisition described earlier.

4.4.4 K-Nearest Neighbour Trust Measurement & Voting

Trust evaluation based on the $\mathcal{K}, \mathcal{K} \in \mathcal{Z}^+$ nearest neighbours is the same as described in chapter 3. The trust update function is adapted in regard to $\max_{Pr}\left(Pr(Y = y|X)_a^i\right)$ factor. In every execution cycle, the grouped instances are classified by the agent members and the average probability estimate results are then fed into to the equation (22) while $r_{a,i}^{\varkappa}$ is calculated as described in section 3.5.1.1.

$$Trust_a^{\varkappa}(i) = Trust_a^{\varkappa}(i-1) \pm \mu\left(r_{a,i}^{\varkappa} + \gamma \max_{Pr}\left(Pr(Y = y|X)_{a,i}^{\varkappa}\right)\right)$$

Equation 20 – k-NN Trust Measurement

where $Pr(Y = y|X)_{a,i}^{\varkappa}$ the average probability estimate is calculated based on the predicted class label obtained by the classifier. As expected, the validity of the individual predicted outcomes influences the trust value that reflects the capabilities in regards of the agent performance. In the next paragraph, equations (23) and (24) depict the implemented voting functions.

$$Majority\ Vote\ A_{n,a}\ SWF = \sum_{i=1}^{n,a}\left(Pr(Y = y|X)_{i,a}^{c} * Trust\ (i-1, a)\right)$$

Equation 21 - Majority Vote SWF Function

$$Majority\ vote \mathcal{A}_{i,a}\ SCF = Pr(Y = y|X)_{i,a}^{c} * Trust\ (i-1, a)$$

Equation 22 - Majority Vote SCF Function

Consequently, $\mathbb{C} = k,\ k \in [1,\infty)$ parameter is integrated in the SWF and SCF functions pointing out the knowledge area constituted by the k-nearest neighbours. Pseudo code in Algorithm (11) describes in more detail the process described above. The next chapter presents the results from experimentation using different setups as described on the previous chapters. CHIMACS is tested in several different datasets with different structures and characteristics. Most of them were taken from UCI ML Repository and one was extracted from StackOverflow Q&A forum.

Algorithm D - CHIMACS

Input: agents, hyper-parameters of agents, number of clusters training samples, test samples, gamma population, gamma initial values, reinforcement signal, trust initial values.
Output: Performance indicators

Determine initial trust, reinforcement signal and gamma parameters.

For each training sample:
Assign Cluster

For each agent:
For each training sample:
Calculate Normalization Value
For each agent:
For each training sample:
Normalize Est. Probability Value

For each knowledge area:
For each test sample:
For each agent:
Predict k-nearest neighbors class a_i^a , $Pr(Y = y|X)_{a,i}^c$
Average Predictions
Initialize Simple SWF Voting
If $a_i^a = 0$ $vote_0 += 1$ **Else** $vote_1 += 1$
Initialize SWF Voting with Prob. Est.
If $a_i^a = 0$ $vote_0 += Pr(Y = y|X)_i^a$ **Else** $vote_1 += Pr(Y = y|X)_i^a$
Initialize SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Pr(Y = y|X)_i^a$
Initialize Trust SCF Voting with Prob. Est.
If $a_i^a = 0$ $vote_a^0 = Trust_a * Pr(Y = y|X)_i^a$ **Else** $vote_a^1 = Trust_a * Pr(Y = y|X)_i^a$
Initialize Trust SWF Voting with Prob. Est.
If $a_i^a = 0$ $vote_0 = Trust_a * Pr(Y = y|X)_i^a$ **Else** $vote_1 = Trust_a * Pr(Y = y|X)_i^a$
End for
Winner SCF Voting = The agent with the highest vote.
Winer SWF Voting = The alternative with the highest vote.
Update Trust
Update Confusion Matrix
End for
End for

Algorithm 11 - D-CHIMACS Pseudo Code

4.5 Distributed Trust Measurement & Voting

Analysis on the unstructured data provides extremely useful information in regard to the hidden patterns. Trust measurements and voting processes this time are dedicated to specific knowledge areas which enables blackboard to distinguish the expertise of its individual KS. Blackboard now has prior knowledge of the identified areas and can categorize the examined observations into these areas. Using this information blackboard initiates the voting procedure that is dedicated to the current knowledge area. Blackboard assigns trust values to KS by capturing the quality of the individual solutions belonging to a specified knowledge area.

Majority SCF and judgment aggregation are used by the blackboard in order to conclude to an alternative either by nominating an agent as a winner in every epoch or by exploiting the diversity in agent capabilities. $Trust_a^c(i)$, that the system has developed up to time t_i-1, is now updated by incorporating $\mathbb{C}$. An agent now is considered an expert in a specific knowledge area having major impact to its vote. Q-Trust is now updated according to the validity and the quality of the answer that belongs to a dedicated knowledge area. Dedicated trust formula is depicted as follows:

$$Trust_a^c(i) = Trust_a^c(i-1) \pm \mu \left(r_{a,i}^c + \gamma \max_{Pr} \left(Pr(Y = y|X)_{a,i}^c \right) \right)$$

Equation 23 - Trust Function

Signal estimation for updating the reward r is modified accordingly. Posterior distribution of $p(\vartheta_a|r)$ given the observation up to time t and cluster $c, where\ c \geq 1$ is now adapted into $p(\vartheta_a^c|r)$. Its uncertainty is assumed to have a uniform distribution $p(r|\vartheta_a^c) \sim \mathcal{U}(\lceil 0,1 \rceil)$ which is the prior distribution. Agent α receives rewards r from Bernoulli distribution $(r|\vartheta_a^c) \sim \mathcal{B}(\vartheta_a^c)$. By applying the Bayes Rule, we approach $p(\vartheta_a^c)$by the posterior distribution (Komiyama et al., 2015):

$$\underbrace{p(\vartheta_a^c|r)}_{Posterior\ \ Distribution} = \frac{p(r|\vartheta_a^c)\,p(\vartheta_a^c)}{\int p(r|\vartheta_a^c)p(\vartheta_a^c)\vartheta_a^c} \propto \underbrace{p(r|\vartheta_a^c)}_{Likeli\,hood\ Distribution} \times \underbrace{p(\vartheta_a^c)}_{Prior\ \ Distribution}$$

Equation 24 - Posterior Distribution

At each round *n*, we get a random draw $\vartheta_a^c(n)$ from the posterior distribution in equation (26), for each agent *a.*

$$p(\vartheta_a^c|r) \sim \beta(N_a^1(n) + 1 \mid N_a^0(n) + 1)$$

Equation 25 - Posterior Distribution Range

where $N_a^1(n)$ and $N_a^0(n)$ is the number of times the agent a got reward 1 and 0 accordingly depending on the validity of their prediction and the data observation provenance.

Consequently, SWF and SCF functions integrate the $\mathbb{C}$ parameter pointing out the knowledge area voting is performed. Equations (28) and (29) describe majority SWF and SCF that are influenced by the $\mathbb{C}$ factor.

$$Majority\ Vote\ A_{n,a}\ SWF = \sum_{i=1}^{n,a}\left(Pr(Y = y|X)_{i,a}^c * Trust\ (i-1,a)\right)$$

Equation 26 - Majority SWF with Trust

$$Majority\ vote \mathcal{A}_{i,a}\ SCF = Pr(Y = y|X)_{i,a}^c * Trust\ (i-1,a)$$

Equation 27 - Majority SCF with Trust

4.6 Conclusion

In this Chapter, firstly, a simple version of the proposed framework was introduced, then a cluster-based component for extracting knowledge patterns from the data was proposed. Exploiting the extracted information from the data gave the blackboard an advantage that can leverage in order to personalize the task assignment. Cluster analysis, using unsupervised ML techniques, enables blackboard to capture and relate agent and problem properties. Existing implementations that employ computational trust methods measure the trustworthiness and reputation in actions in terms of maliciousness and ignore the performance. The proposed framework is able to fuse the acquired knowledge and it is not bounded to any domain explicitly. However, every version adds complexity in terms of time. Nevertheless, regardless of the added time complexity, every version of the framework presented useful results in capturing diverse skills and exploiting them to boost the predictive performance. The goal of introducing such framework is to provide a flexible and adaptive approach for boosting predictive performance by capturing and exploiting diverse capabilities through voting in the environment.

Each version of the proposed framework acts as a basis for the next. Practically the final version incorporates all necessary components. The goal of introducing a decentralized intelligent framework is to provide the means to boost predictive performance in an adaptable and flexible manner.

Firstly, the proposed framework can facilitate the transparency of the results. Indeed, observing the trust values related to every agent-knowledge area facilitate the exploration of agent's capabilities related to a specific cluster. Secondly, it is capable of boosting predictive performance by enhancing the votes of SCF and SWF using trust measurements. Trust is a metric that evaluates the performance of the agent-members in the environment. Trust value is accumulated during the entire process of classification and acts as a weight that influences the probability estimate of the answer the agent provides. Hence, trust either enhances or diminishes the impact of the individual votes. Finally, as discussed earlier, trust is a representative measure of the agent's competence. That provides a solution to the problem of new incoming

data points. The blackboard will automatically pick the need for training by monitoring the trust levels of the agents.

To sum up, this Section described three different versions of CHIMACS. Initially, simple CHIMACS was introduced, where the blackboard has no prior information regarding the experts and the examined problem. That version is used as a base for comparing the performance of the next versions. Clustered CHIMACS introduced an unsupervised blackboard capable of extracting information from the data exploit them. The blackboard has the ability to observe and infer after trial and error which agent is more competent and enhance their opinion through trust. Finally, a dynamic and adaptable blackboard, which eliminates the need for an unsupervised model creation and re-construction during the process, was presented.

4.7 Summary

In this Chapter, a novel hybrid and intelligent classification framework is presented. The proposed framework provides the means to capture and combine the knowledge from different ML KS. The problem of choosing a framework set-up to boost predictive performance is explored. This includes testing, comparing and then choosing a configuration that provides the best results. Firstly, a novel method to efficiently capture the performance of each agent in terms of accuracy, specificity, sensitivity and precision was proposed. The framework employs reinforcement learning methods which facilitate the exploration and exploitation of diverse capabilities and increase the overall predictive performance. Secondly, Thomson Sampling for efficient optimisation of reward signal estimation and of the trust function is discussed. Thirdly, five different majority voting formulas were applied to let every expert express its opinion to the blackboard. Also, a normalization process is applied to allow slower progress and convergence. Finally, the approach presented yields promising results since it is capable of capturing the correct voting behaviour through the use of computational trust methods.

Chapter 5 CHIMACS Setup for an Efficient Prediction Boost

In this Chapter, the performance of the three set-ups of the proposed general-purpose framework is explored. The framework's performance is evaluated against a number of datasets where metrics like accuracy, sensitivity, specificity and precision are evaluated. The consistency of the results is compared and verified for a few groups of datasets from the literature, one dataset from Stack Exchange, namely Stack Overflow and Stack Maths and three datasets from UCI ML Repository. Experimentation is divided into two main groups where different evaluation metrics are examined each time. Hence, the results from three distinct set-ups of a multi-agent blackboard ML framework for boosting the overall predictive performance are presented and discussed.

In the beginning, a brief introduction of the three set-ups of the proposed framework are introduced in Section 5.1. The experiment outline is discussed in Section 5.2. Experiments of comparing different learning and voting modes implementing case study I and II are presented in section 5.3. The results of both case studies are discussed in section 5.3. Finally, section 5.4 provides a conclusion and a summary of the Chapter.

5.1 Introduction

Tremendous amounts of data are generated as a result of human activities. The application of ML and statistical methods emerged from the need of extracting useful information from the data produced. Due to the fast growth in the amount of the produced data, as well as the increase in the domains which demand knowledge extraction from their data, there is the problem of choosing an appropriate ML

algorithm and achieving valuable prediction performance. As a result, many researchers have worked on developing ensemble ML methods (Drucker et al., 1994; Polikar, 2009). At the same time research is also focused on scalable multi agent ML approaches for boosting prediction performance (Kokorakis, Petridis, & Kapetanakis, 2017; Pourpanah et al., 2017; Teodorescu & Petridis, 2013).

In supervised learning, ensembles are considered a set of classifiers whose individual decisions are combined in some way typically by weighted or unweighted voting in order to classify new examples. The main discovery is that ensembles predictions are often much more accurate than the individual classifiers (Dietterich, 2000). However, there are some disadvantages using ensembles for boosting predictive performance. The application of an ensemble method requires insight of the problem at hand. For example, models with high variance are likely to benefit from bagging. On the other hand, biased models are better used combined with Boosting. At the same time, complexity of the classification and computational time increases. An ensemble is not interpretable, a fact which makes it hard to explain the deduction process of the constructed model. Also, whenever there are additional observations re-training process is time consuming.

However, there is a question about how effective trust reward, diversity and voting can help perform better in terms of classification and probably faster training (Kokorakis et al., 2017). To answer this question a series of experiments are presented in this chapter. In order to prove the efficiency and efficacy of the proposed framework, a number of different datasets were used as test bed. The consistency of the results is verified for a few datasets, one dataset from Stack Exchange, namely Stack Overflow and Stack Maths, five datasets from UCI ML repository and one from Kaggle repository. At the end of each experiment the classification results and the time performance of training and testing is compared against the individual classifiers and known literature.

5.2 Experiment Outline

A series of noisy and noise free benchmark problems were employed to evaluate the efficiency and efficacy of the proposed framework. Firstly, the noisy benchmark problem, waveform dataset, three biomedical, one credit card approval from the UCI ML repository (Lichman, 2013) were employed. Also, an SMS spam dataset from Kaggle. Details of the data sets are shown in Table 2. The results from CHIMACS, C-CHIMACS and DN-CHIMACS were compared with those from its constituents as well as other models reported in the literature.

As far as classifier's hyper-parameters are concerned, grid search optimization is applied. The models used vary from traditional ML models like LR, DT and SVM's and ensembles like Adaboost, bagging, SGD and random forests. As a result, the experimental parameters change for every experiment setup.

The experiments are divided into two main case studies. The first case study is focused on prediction tasks where accuracy performance metric is improved. The performance of the proposed framework is evaluated through the function in equation 26. Balanced datasets were used where the specified metric makes sense (Gopalakrishna, Ozcelebi, Liotta, & Lukkien, 2013).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Equation 28 - Accuracy performance metric

On the other hand, the second case study deals with highly imbalanced datasets where accuracy does not produce any valuable inference. The framework enhances the predictive performance by boosting specificity, precision and sensitivity metrics. Recall or sensitivity is the ratio of correctly predicted positive observations divided to all observations in actual class. In short, recall is the fraction of relevant instances that have been retrieved over the total amount of relevant instances (Gopalakrishna et al., 2013).

$$Recall = \frac{TP}{TP + FN}$$

Equation 29 - Recall Performance Metric

Precision is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. In more detail, precision is the ratio of correctly predicted positive observations divided by the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP}$$

Equation 30 - Precision Performance metric

True negative rate or Specificity calculates the correctly classified proportion of negatives e.g. for datasets related to medicine, specificity reflects the percentage of healthy people who are correctly identified as not having the condition. The function in equation 24, describes specificity metric (Gopalakrishna et al., 2013):

$$Specificity = \frac{TN}{TN + FP}$$

Equation 31 - Specificity Performance metric

Finally, hyper-parameter optimization through grid search is based on f-measure in case of class imbalance. The F1 score can be interpreted as a harmonic mean of the precision and recall, where an f_1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the f_1 score are equal. The formula for the f_1 score is (Gopalakrishna et al., 2013):

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Equation 32 - F1 Score

Concluding, different variations of f-measurement are $f_{0.5}$ which weighs precision more than recall and f_2 that weighs recall more accordingly. The use of each variation of f score depends on the problem in hand.

For every dataset the features described are aggregated in a feature matrix of size n × m, where n is the number of features F, m is the number of instances Q and C is the number of predicted classes, where $C \in [0,1]$. This matrix is used for training and then testing every classification model (agent) built by a ML algorithm. For certain regularized classification models such as k-NN and SVM, it is critical to apply regularization and normalization methods. Whenever necessary, in order to avoid features in greater numeric values to dominate others, the features are normalised to an interval [0, 1]. Also, the features are standardised by removing the mean value μ = 0 and scaling variance to one σ = 1. Also, this data transformation can improve the training time and prediction performance for SGD classifiers. In the end, variable encoding for categorical features was applied to convert numeric features. The prediction tasks are formulated as binary classification tasks and then executed across all the above-mentioned datasets.

Finally, three experiments are conducted. Same architecture is followed for each of the three experiments. However, the blackboard kernel is changed. In the first, a kernel without knowing a priori the existence of the knowledge areas is constructed. That version is used as the base line which the rest of the results are compared with. A blackboard kernel that is aware of the distinct knowledge areas is constructed using un-supervised ML methods for the needs of the second set-up and finally, a kernel based on k-NN method for identifying dynamically the knowledge areas is configured. In every case, a number of classifiers were tested and the best five performing agents (LR, k-NN, DT, SVM, SGD) were chosen to represent it as experts for the examined problem. For every experiment, the algorithms were trained with the same dataset following the standard 10-fold cross validation training process. Each algorithm is executed ten times and then the average and standard

deviation of these independent runs is recorded. CHIMACS: The number of iterations is not predefined; the execution is continued until convergence. Through trial and error the learning rate μ is set to be equal to '1'. Agent Learning Algorithms: Since, this is a general-purpose framework hyper-parameters depend on the examined data. Hence, grid search is performed in every occasion for optimal configuration.

Table 2 - Datasets Description

Dataset	Number of Data Samples	Number of Features	Number of Classes
Waveform Database Generator (version 2)	3350	40	2
Q&A	1.000.000	62	2
SPECT	267	22	2
Pima Indian Diabetes (PID)	768	8	2
ILPD	583	10	2
German Credit Card Approval	690	15	2
Hill-Valley	606	101	2

5.2.1 Case Study I

Four problems from a variety of domains were used: (i) Waveform database generator (a problem with artificially generated noisy data), (ii) the wine quality dataset, (iii) Q&A from StackExchange and finally, (iv) a credit card approval

dataset from a German bank. The classification task for the Waveform dataset is to predict the class of the artificial wave. Wine dataset includes red and white wine quality datasets and the problem was transformed into binary where the classification task is to predict the colour of the wine. Q&A another noisy dataset was extracted and pre-processed for the needs of Nikolay Burlutskiy's research (Burlutskiy, Petridis, Fish, Chernov, & Ali, 2016). The classification task was to predict the response time of a potential answer that was given to a question, whether it is going to be answered in less or more than an hour. The last dataset examined in this case study is data related to credit check where a potential client's credit risk is evaluated by the bank and a credit card is approved or not. For each dataset examined, agent classifiers received a 10-fold cross validation training and grid-search hyper parameter optimisation. As far as framework training is concerned, 10% of the test set is used for training the framework's trust measurements and avoiding the cold start problem.

5.2.2 Case Study II

Three medical benchmark problems taken from public domain repository, including heart disease, liver disease, diabetes and one SMS spam benchmark problem from Kaggle were used in this case study. One heart disease data sets (SPECT) was taken from the UCI ML repository. The Pima Indian Diabetes (PID) was taken from the UCI data repository, too. The classes of each data set were set to 0 and 1 for absence and presence of the target disease, respectively. The details of all data sets are shown in Table 2.

Following the same experimental procedure as before, the 10-fold cross validation and grid search optimisation were applied. However, in addition to accuracy, sensitivity, precision and specificity are part of the performance evaluation process. Class imbalance and the context of the datasets dictate that these metrics are now more useful performance indicators to evaluate the performance of classifiers for medical binary problems.

In the next sections, the results of the experimentation are presented. In every section, the visualisation figures present the performance of either the individual

agent classifiers or the proposed frameworks performance. The results presented in the tables follow the experimentation process described above. Firstly, the results of the experiment without distinct knowledge areas are described, followed by the results of the second set-up and lastly the results of the k-NN based approach are presented. At the end of each experiment an evaluation of the results is discussed and compared to related literature.

5.3 Results and Discussion

The results for the three experiment set-ups, comparing different voting modes across the fourteen datasets, are presented in the tables in the section followed. The results for the change in the accuracy of predictions with respect to the number of iterations, knowledge areas and number of nearest neighbours are in the next figures for every examined dataset correspondingly.

5.3.1 Accuracy Boost

5.3.1.1 Waveform Data Generator (Version 2)

Waveform dataset was pre-processed to include only two of the three classes of waves. Each class holds 33% of the dataset and each instance was created with zero mean and variance equal to one. Boxplots in figure 27 visualize the 10-fold cross validation scores using accuracy as scoring function. It is observed that median accuracies vary between 89% and 92% and the highest standard deviation 0.003. It is obvious that we deal with models that have low bias and low variance. It is expected that the models will perform good on the test set.

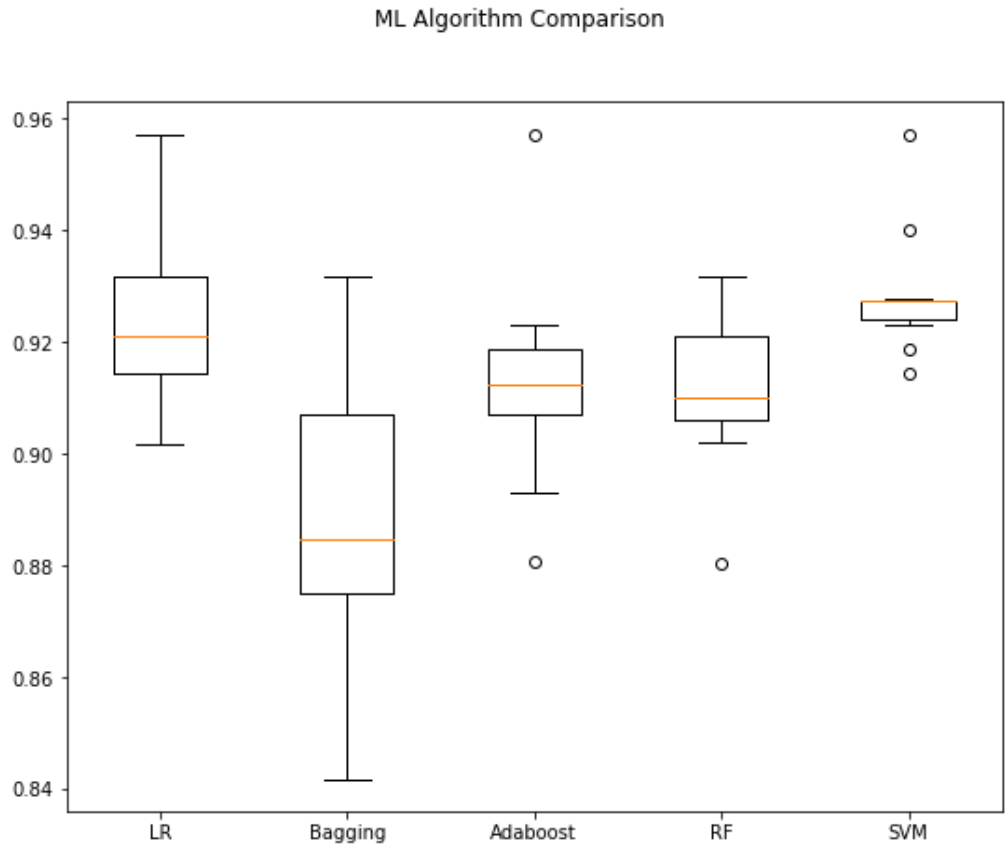


Figure 20 - Agent Training Performance Comparison

Accuracies in table 3 meet the expectations stated above. All agents performed very well in regard to the data observations used as test bed.

Table 3 - Accuracy rates (%) on the test set, training and testing time in milliseconds for Waveform

Agent	**Accuracies**	**Training Time, s**	**Testing Time, s**
LR	92.43	0.25	0.352
Bagging	89.24	11.4	360.2
Adaboost	90.23	0.39	0.456
Random Forest	92.23	0.73	0.969
SVM	91.63	3.0	0.955

Tables 3,4 and 5, along with the figures below them, describe the results in regards with accuracy performance. More specifically, the accuracy performance of three different kernel set-up of the proposed system along with different voting methods is presented. The rest of the figures depict the system progress in terms of epochs until convergence. One epoch is considered as a full classification cycle that depends on the number of test samples.

The first version of the proposed framework describes the results of the five voting methods proposed in section 3.2.1. Accuracies from the first two SCF and SWF voting functions underperform compared to the worst to the individual agent classifier. On the other hand, the results from the last two majority voting that are influenced by trust performed are better than the rest, however, the system became as good as its best agent. According to Figure 28 the system converged around after epoch 100. Training and testing performance are the times in seconds that the highest accuracy or any other performance metric reaches its highest value.

Table 4 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	CHIMACS		49.10		2.76
SCF – Prob. Estimate Simple			92.52		0.44
SWF – Prob. Estimate Simple			92.82		0.51
SCF – Trust Estimate		95.12	93.75	7.137	1.548
SWF – Trust Estimate		95.48	93.90	5.436	3.818

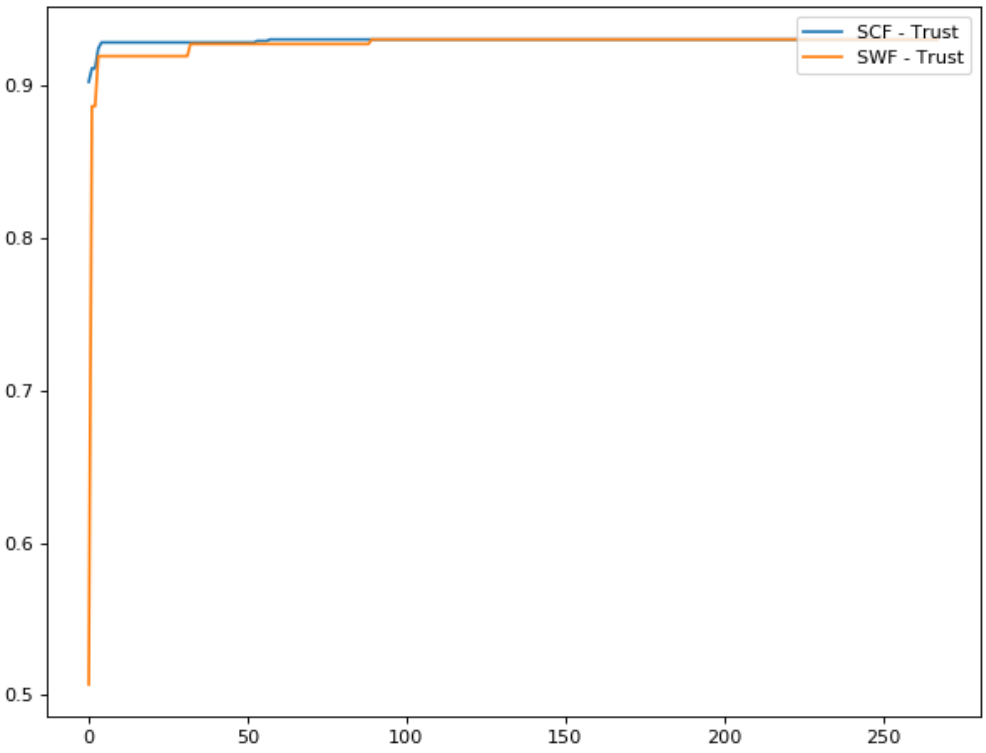


Figure 21 – CHIMACS SWF & SCF Accuracy Performance

Accuracies regarding the last two voting methods in table 6 perform just slightly better than the previous version of CHIMACS.

Table 5 - Accuracy Rates (%) and Testing Time for C-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	C - CHIMACS		49.10		2.76
SCF – Prob. Estimate Simple			92.52		0.44
SWF – Prob. Estimate Simple			92.82		0.51
SCF – Trust Estimate		96.48	94.30	4.958	1.884
SWF – Trust Estimate		95.35	94.02	6.145	3.756

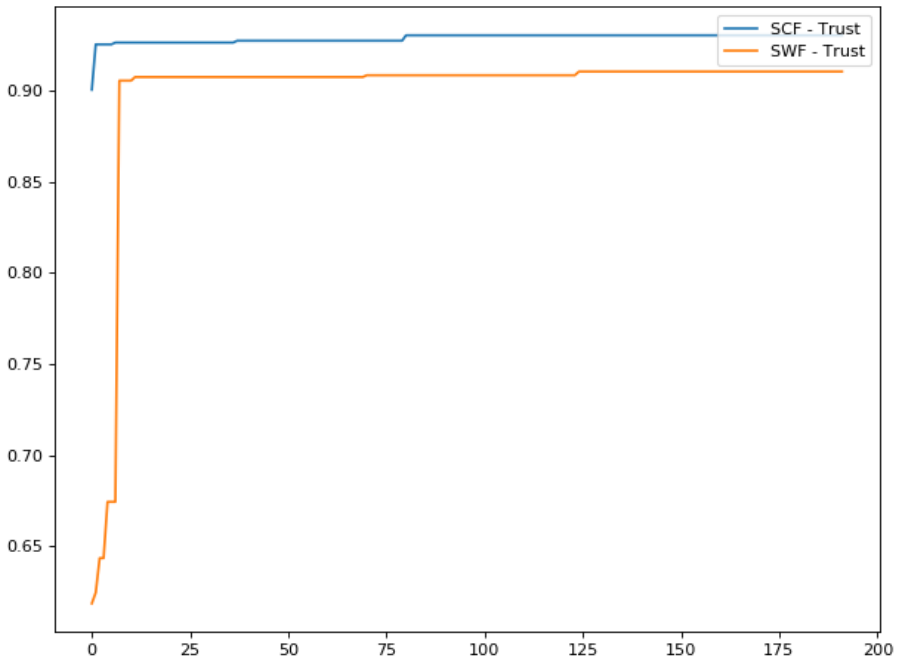


Figure 22 - C-CHIMACS SWF & SCF Accuracy Performance

Although the system took the longest time to complete the simulations run compared to the rest of the set-ups, Dynamic CHIMACS did not converge to better results than the rest of them or the individual models. In fact, SWF function yielded 1.27% worse accuracy compare to C-CHIMACS. The fact that did not presented better results relies probably on some of the reasons stated next. One possible reason is the distribution and the structure of the data. One limitation of this version is that each kernel in based on k-NN which practically is a ML algorithm. That means its behaviour is closely related to the data given. Another reason is the similarity of the metric implemented. The similarity function is kept the same for efficacy reasons in every experimentation which is not the best practice given than the data domain, hence their structure, is changing in every experimentation.

Table 6 - Accuracy Rates (%) and Testing Time for DN-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Test Time, s**
SWF – Simple	DN - CHIMACS		49.10		2.356
SCF – Prob. Estimate Simple			92.52		2.862
SWF – Prob. Estimate Simple			92.82		2.533
SCF – Trust Estimate		97.41	91.03	18.147	13.715
SWF – Trust Estimate		95.834	93.02	16.236	14.715

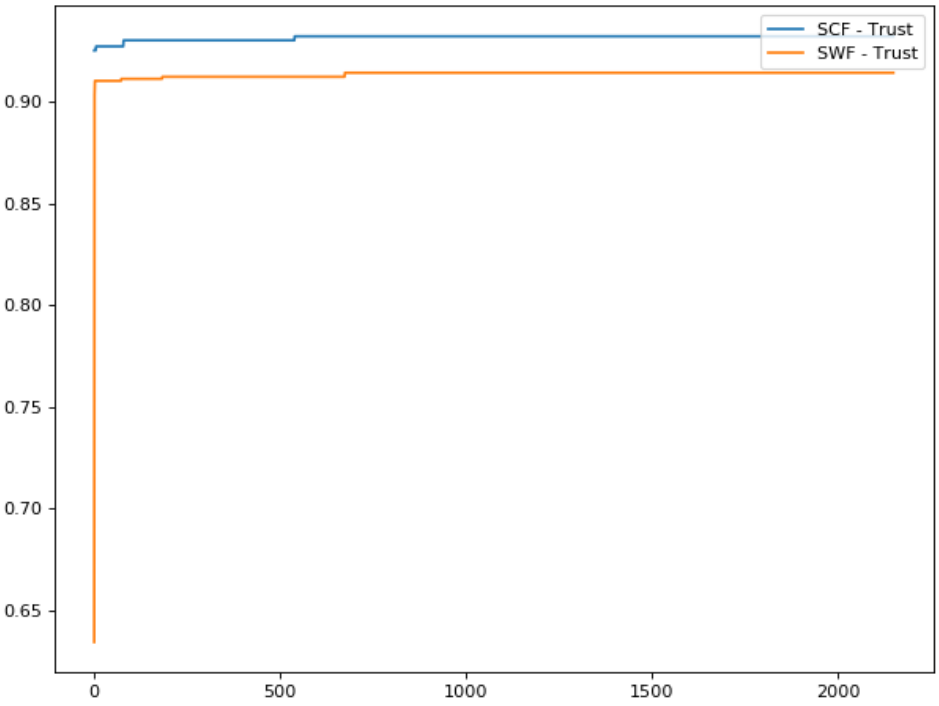


Figure 23 - DN-CHIMACS SWF & SCF Accuracy Performance

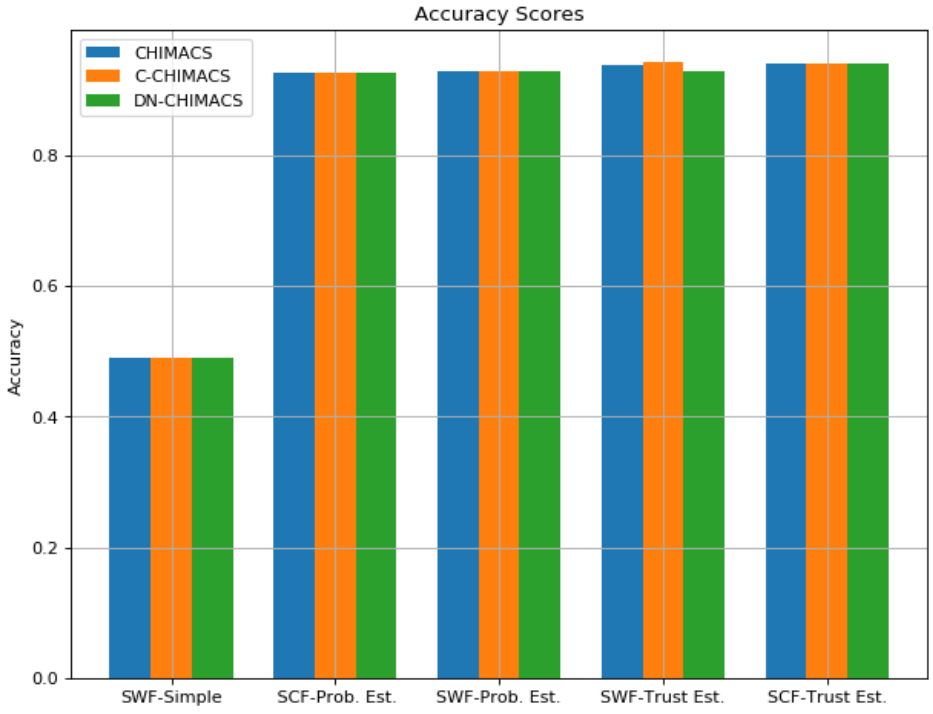


Figure 24 - Accuracy rates (%) for every voting function of the framework for the waveform problem

Accuracy results from the individual ML agent classifiers were already at very good level. Over 92% accuracy in predicting wavefoms. However, none of the

implemented version could boost prediction performance. Accuracy levels from the figure above, indicate that naïve CHIMACS became as good as its best expert inside the MAS environment. Social welfare voting function from C-CHIMACS showed over 2% increase in accuracy, which is not a major increase given the training and testing time.

5.3.1.2 Wine Datasets

This specific dataset is the result of merging two separate datasets, the red and white wine datasets (Lichman, 2013). Both come from UCI ML Repository and originally describe the quality of the wines giving an integer value in the range of 1-7. The two datasets were merged and an extra column was added specifying the colour of the wine. The classification task was transformed into a binary problem in order to fit CHIMAS specifications.

The dataset is not considered very challenging(Lichman, 2013), thus, most classifiers exceed 90% accuracy. The results from classifiers training are presented in Figure 32 and the results from testing in table 7 support that.

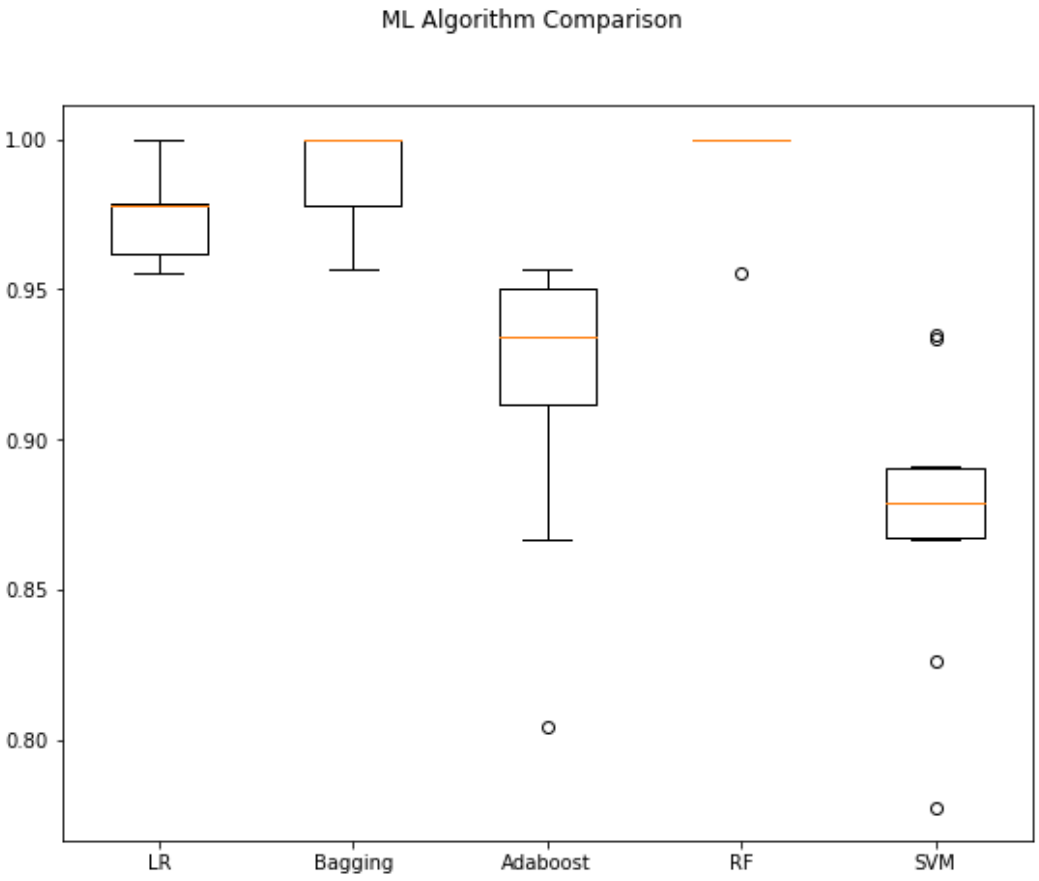


Figure 25 - Agent Performance Comparison

Table 7 - Accuracy rates (%) on the test set, training and testing time in milliseconds

Agent	**Accuracies**	**Training Time, s**	**Testing Time, s**
LR	96.41	1.40	0.010
Bagging	98.41	10.67	0.035
Adaboost	93.33	0.18	0.016
Random Forest	98.46	0.48	0.048
SVM	88.20	22.39	0.065

The system presents almost identical behaviour as before. CHIMACS converged to an accuracy level that is as good as its best agent expert. According to Figure 33 it converged around epoch 150 in a very short time.

Table 8 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	CHIMACS		74.35		2.76
SCF – Prob. Estimate Simple			95.38		0.44
SWF – Prob. Estimate Simple			97.43		0.51
SCF – Trust Estimate		98.84	98.97	5.241	0.548
SWF – Trust Estimate		98.25	98.97	2.365	1.494

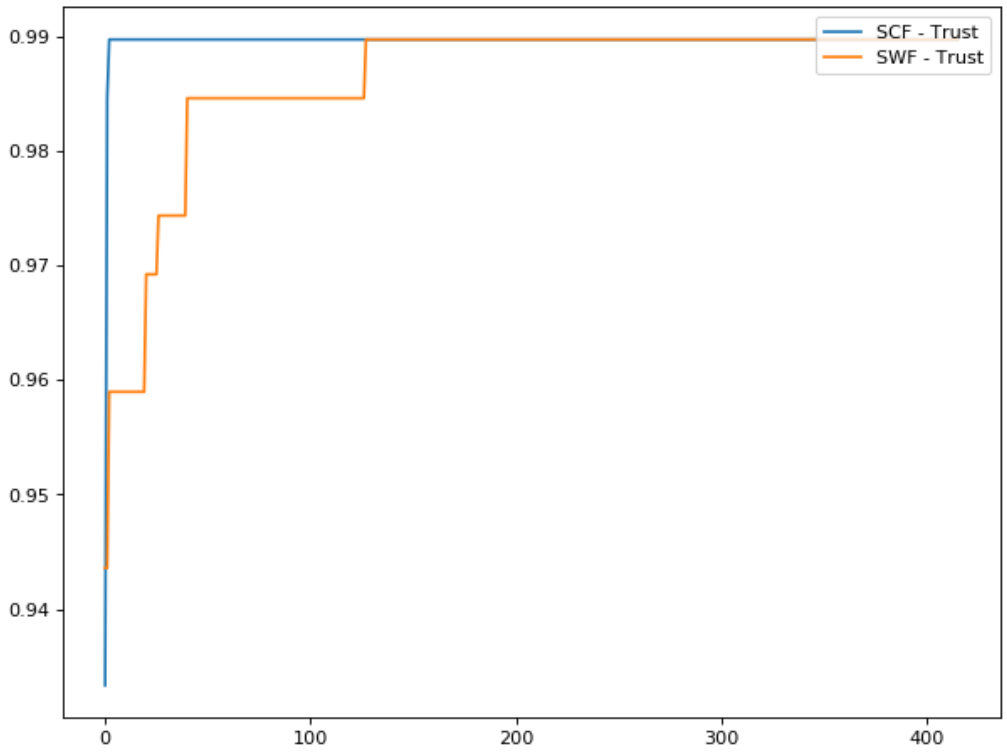


Figure 26 - CHIMACS SWF & SCF Accuracy Performance

Social choice voting exhibited the best performance compared to the social welfare function. Clustered CHIMACS converged around epoch 430 while the time taken remained in low levels.

Table 9 - Accuracy Rates (%) and Testing Time for C-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	C - CHIMACS		74.35		2.76
SCF – Prob. Estimate Simple			95.38		0.44
SWF – Prob. Estimate Simple			97.43		0.51
SCF – Trust Estimate		98.98	99.80	3.982	1.142
SWF – Trust Estimate		98.69	98.36	2.365	2.113

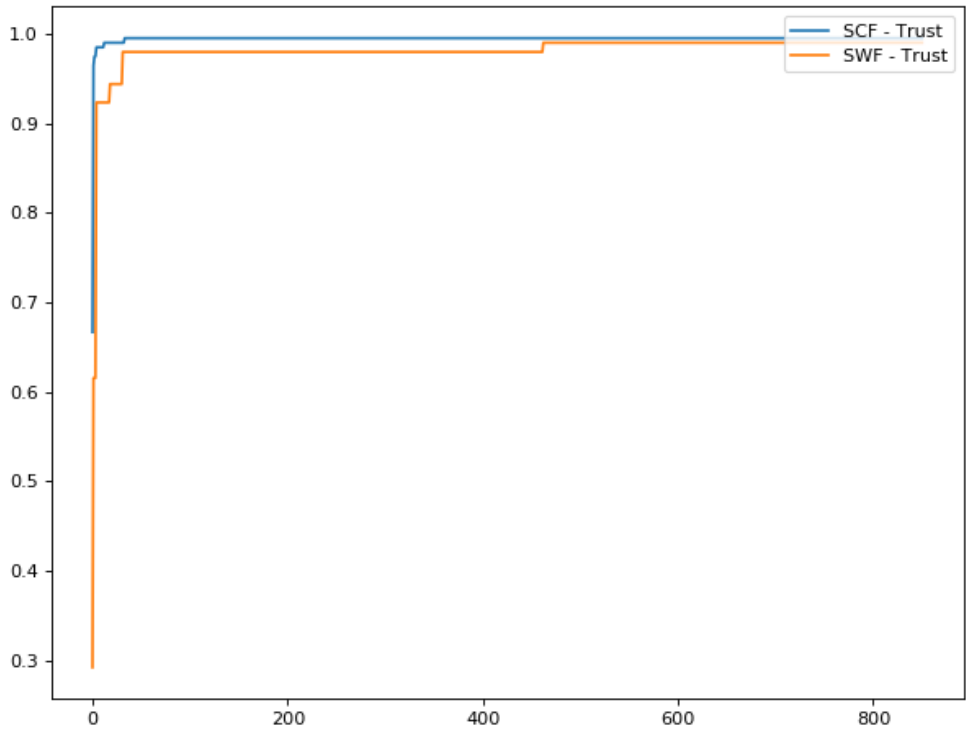


Figure 27 – C - CHIMACS SWF & SCF Accuracy Performance

Dynamic CHIMACS performed as well as the clustered set-up. However, this time SWF voting presented better results than SCF with trust estimate. The results both from clustered and k-NN based versions of CHIMACS are almost the same. However, it took less time for the DN – CHIMACS to converge to optimal accuracy.

Table 10 - Accuracy Rates (%) and Testing Time for DN-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	DN - CHIMACS		74.35		2.356
SCF – Prob. Estimate Simple			95.38		2.862
SWF – Prob. Estimate Simple			97.43		2.533
SCF – Trust Estimate		99.0	99.45	5.396	8.715
SWF – Trust Estimate		97.87	99.58	7.365	4.715

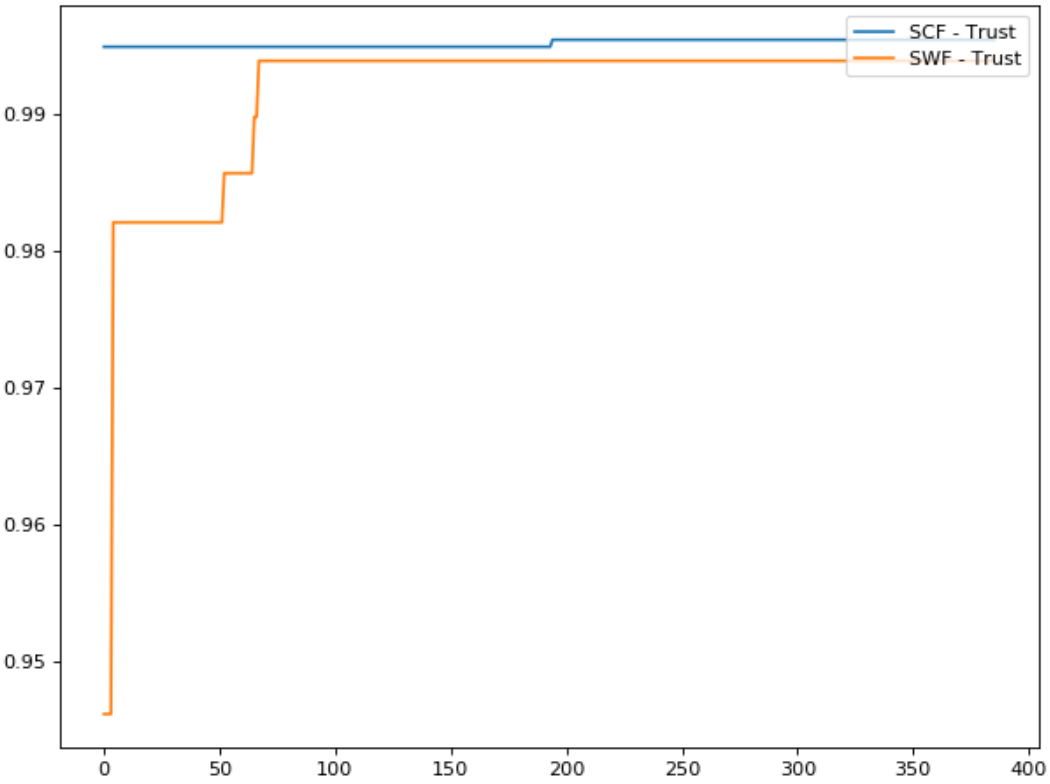


Figure 28 - DN-CHIMACS SWF & SCF Accuracy Performance

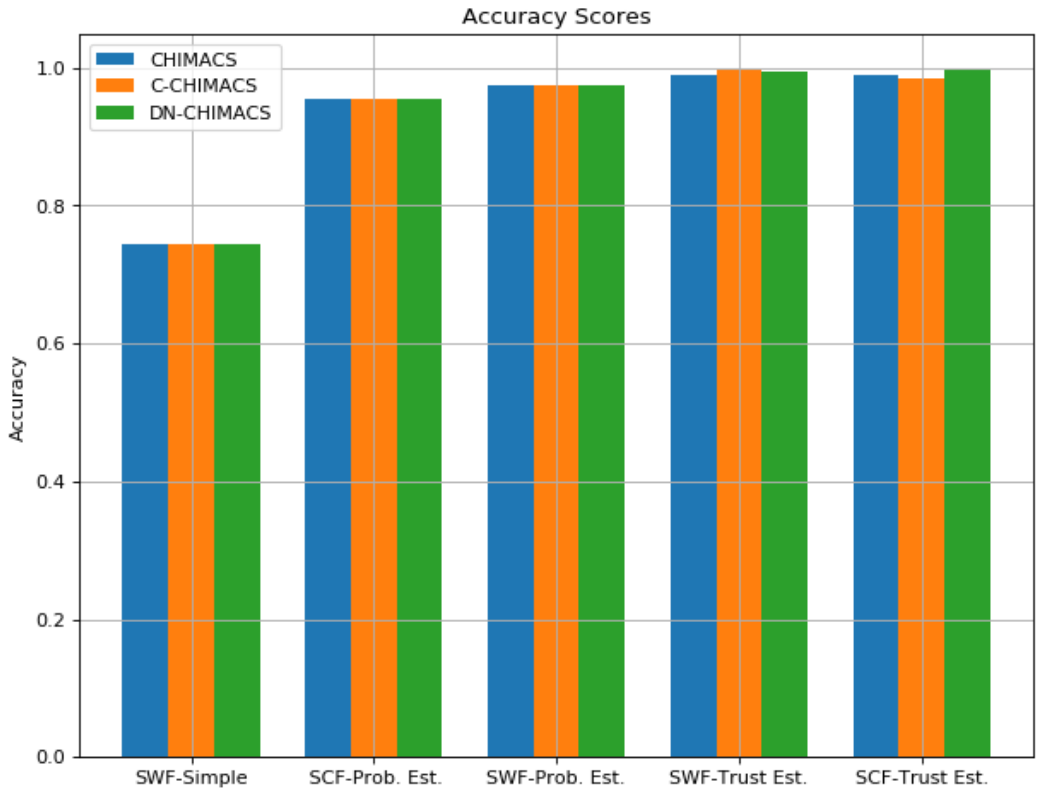


Figure 29 - Accuracy rates (%) of the framework for the waveform problem

Wines are a very well-structured dataset modified into a binary classification task. The results from the individual classifiers are already very good reaching 98% accuracy in predicting and categorizing a wine as red or white. All versions of CHIMACS presented good results with the last two clustered and dynamic behaving best, reaching over 99% prediction accuracy. It is obvious that the last two versions could perform better to the naive since they were given more information e.g. number of clusters regarding the examined problem.

5.3.1.3 Q&A Stack Exchange Dataset

Stack Exchange websites enable users to post questions about a number of topics by posting messages on the platform and receive related answers. Containing almost 8 million questions and over 13 million answers StackOverflow is considered one of the largest Q&A forums of StackExchange that enables access to massive data dumps. Answered questions are considered the questions which have at least one answer accepted by the asker. Related research (Burlutskiy et al., 2016) proposes an ML set-up for efficient user behaviour prediction. The task is approached as a binary classification problem where the class to be predicted is the response time e.g. 0 if the response is less than an hour and 1 if it is more.

The boxplots in Figure 37 and the accuracy results in table 14 present the performance of the classifiers both from training and testing procedures accordingly. The best cross-validation means accuracy is around 75% which is reflected from test set results.

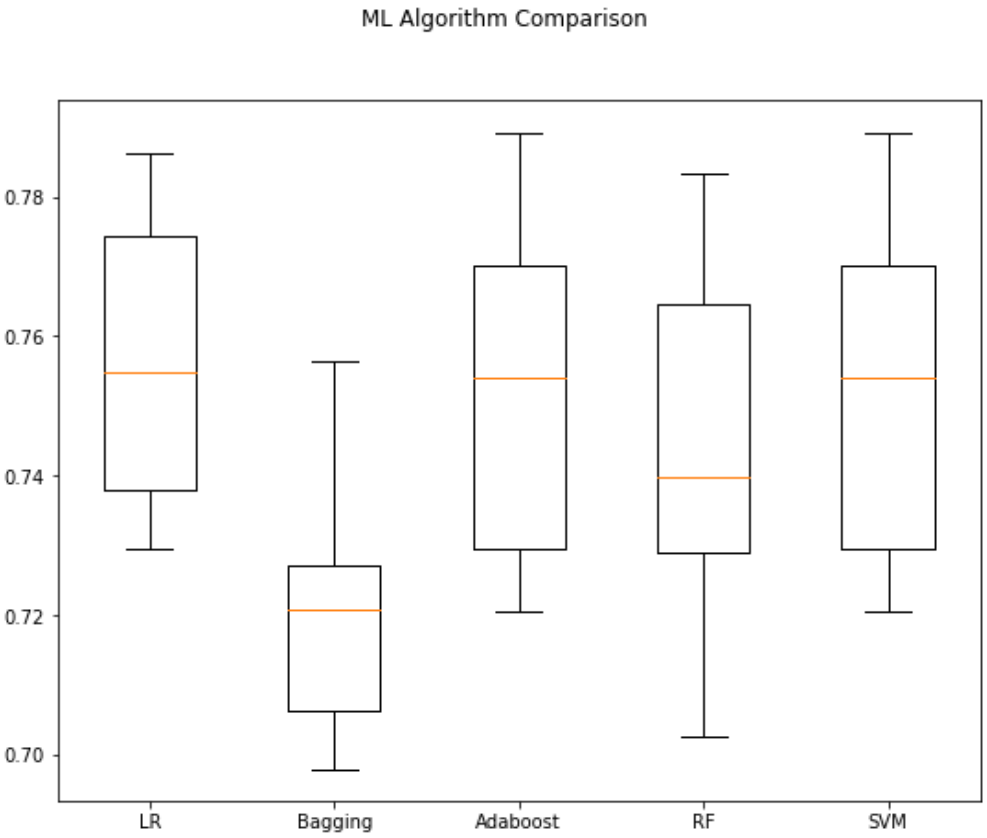


Figure 30 - Agent Performance Comparison

Table 11 - Accuracy rates (%) on the test set, training and testing time in milliseconds

Agent	**Accuracies**	**Training Time, s**	**Testing Time, s**
LR	74.66	0.43	0. 10
Bagging	71.45	10.96	0.95
Adaboost	73.71	1.03	0.67
Random Forest	74.20	1.22	0.048
SVM	73.71	51.96	6.23

The results in table 12 show a small boost in accuracy compared to the results of the simple version of the framework in table 13. According to Figure 38, the system accuracy converged around 430 epochs, without the level of accuracy exceeding by much the level of LR.

Table 12 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	CHIMACS		73.71		18.28
SCF – Prob. Estimate Simple			74.45		1.28
SWF – Prob. Estimate Simple			74.20		1.28
SCF – Trust Estimate		79.85	74.97	11.36	8.648
SWF – Trust Estimate		77.12	74.52	10.957	8.848

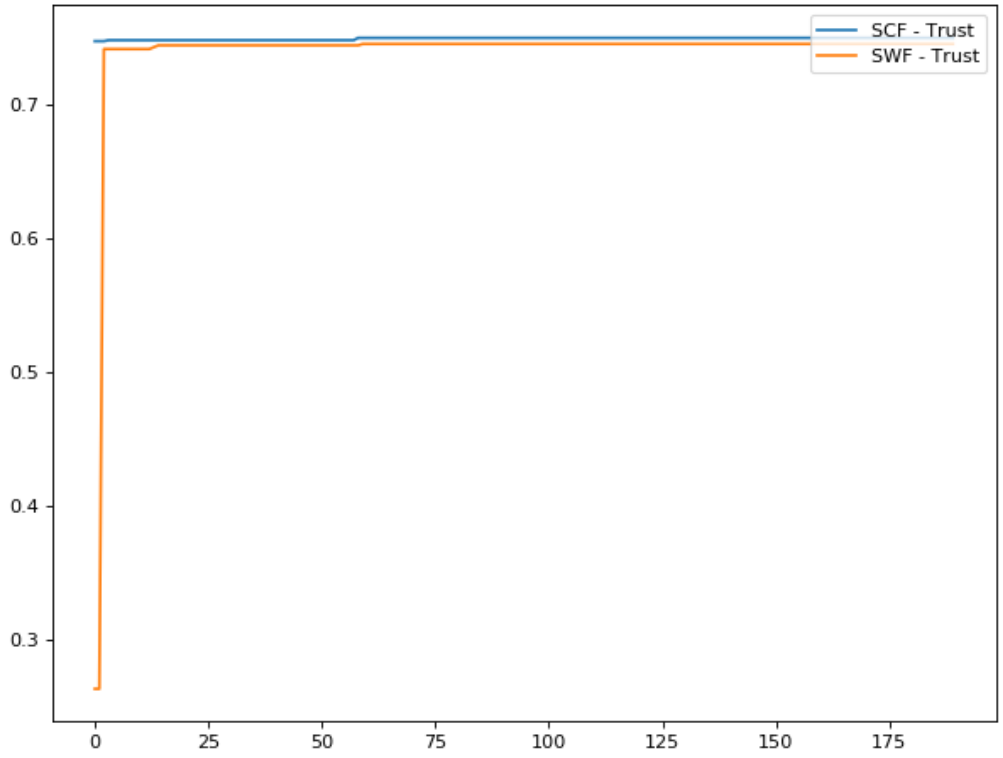


Figure 31 - CHIMACS SWF & SCF Accuracy Performance

Table 13 - Accuracy Rates (%) and Testing Time for C-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	C - CHIMACS		73.71		2.76
SCF – Prob. Estimate Simple			74.45		0.44
SWF – Prob. Estimate Simple			74.20		0.51
SCF – Trust Estimate		78.29	75.0	4.765	6.658
SWF – Trust Estimate		76.37	74.52	7.578	7.412

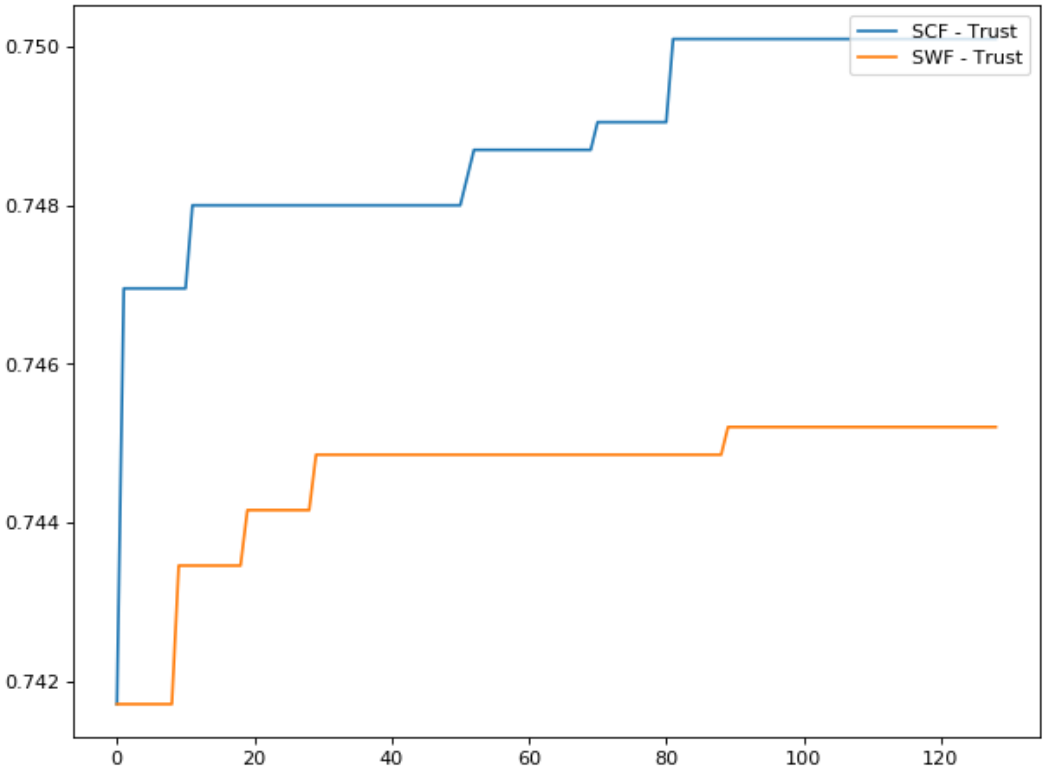


Figure 32 – C - CHIMACS SWF & SCF Accuracy Performance

k-Nearest Neighbour CHIMACS reduced the error in accuracy by 3.75% which can be considered a significant increase in accuracy. It took a sufficient amount of time for the system to converge and around epoch 300 we observed the last increase in accuracy. The main reason for time delay is computational complexity and data dimensionality (Keogh & Mueen, 2017; Nodarakis et al., 2017). KNN employs a number of distance metrics in order to find the k most similar observations. For every dimension added to the dataset calculating, the distance becomes more and more computational expensive. Finally, the search space is an important factor since this is a serial search space implementation.

Table 14 - Accuracy Rates (%) and Testing Time for DN-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	DN - CHIMACS		73.71		2.356
SCF – Prob. Estimate Simple			74.45		2.862
SWF – Prob. Estimate Simple			74.20		2.533
SCF – Trust Estimate		82.36	78.39	14.745	17.715
SWF – Trust Estimate		79.89	76.65	9.156	2.149

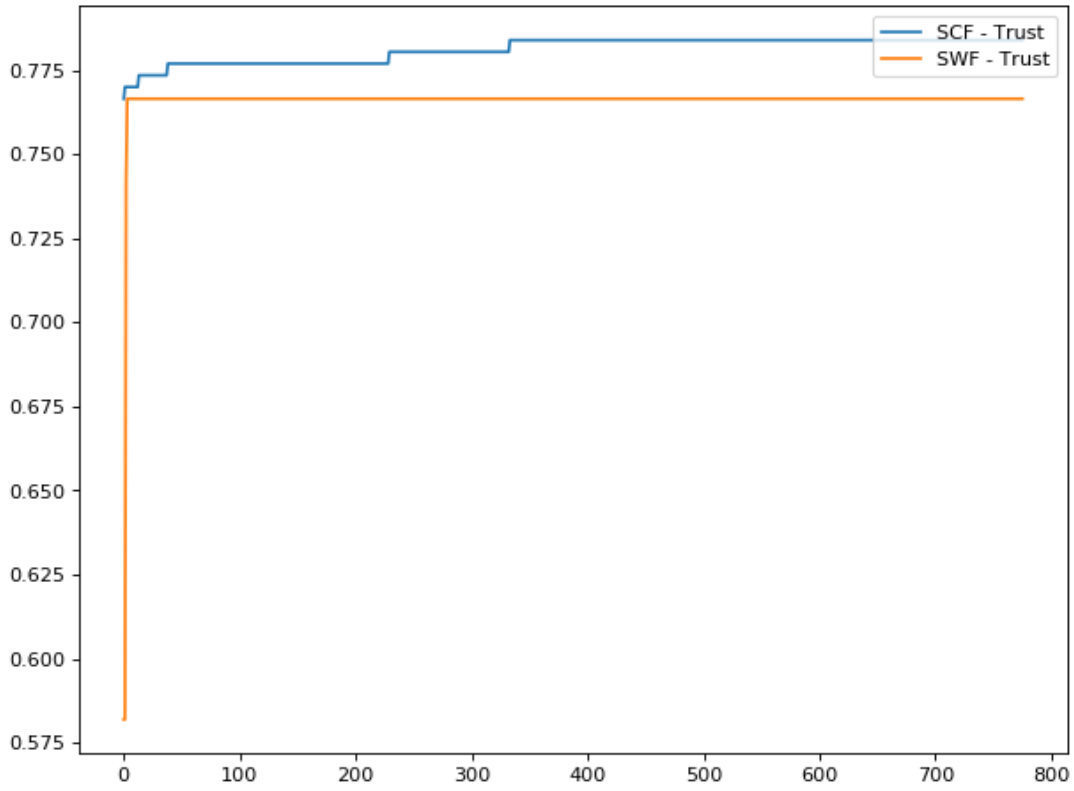


Figure 33 - DN-CHIMACS SWF & SCF Accuracy Performance

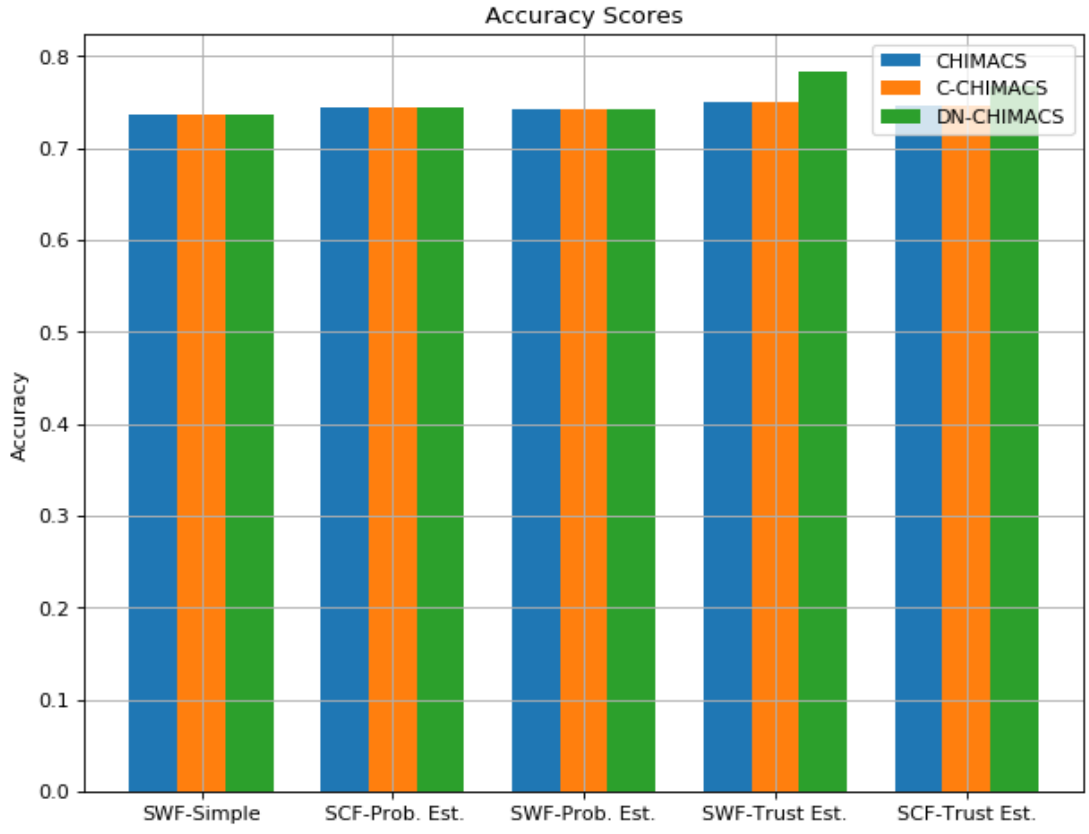


Figure 34 - Accuracy rates (%) of the framework for the waveform problem

Accuracy levels form the bar chart in Figure 41 present SWF voting function and k-NN kernel as the most promising version of CHIMACS. Q&A dataset is extremely noisy and the cluster analysis showed over 350 cluster areas. It is obvious that both naïve and clustered version of the proposed system would not yield good results. Dynamic CHIMACS increased accuracy about 3.79%. A speculation on why this could happen is because k-NN kernel is able to find and retrieve the noisy observations and use the most competent experts in order to classify them.

5.3.1.4 German Credit Card Approval

This dataset came also from UCI ML Repository (Lichman, 2013) and includes details of clients from a German bank. The clients are evaluated and according to the calculated risk a credit card is approved or not. LR performed best with accuracy of 85.48%, while the rest of the model accuracies vary from 64% to 83%. There is not always clear why an ML agent performs better or worse because it depends on the data and the concept under which the algorithm handles the data. A reasonable explanation on the behaviour of LR, is that of the data linearity. Linear models are known to perform better using data than when are linearly separable.

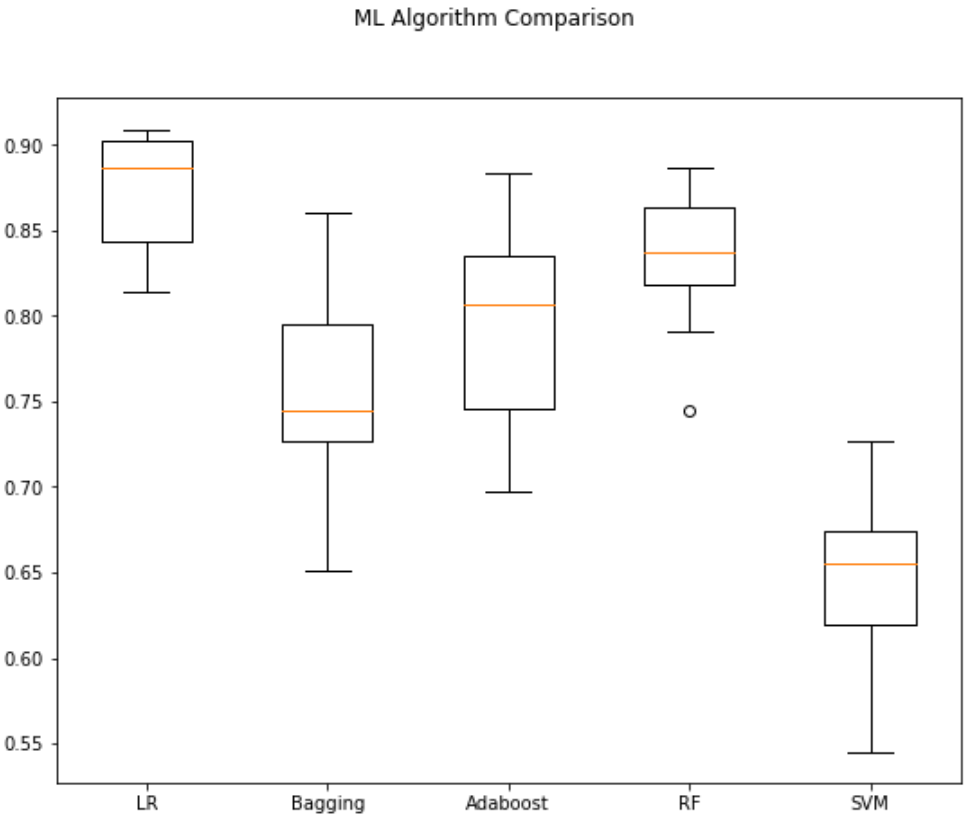


Figure 35 - Agent Performance Comparison

Table 15 – Accuracy (%) on the test set, training and testing time in milliseconds

Agent	**Accuracies**	**Training Time, s**	**Testing Time, s**
LR	85.48	0.01	0.009
Bagging	77.33	11.14	0.047
Adaboost	81.01	0.013	0.084
Random Forest	83.09	0.151	0.658
SVM	64.81	0.467	0.978

Despite the history in results of the simple version of the proposed framework, this time it managed to decrease the classification error and increase the accuracy over 2%. That is quite unusual, since the system does not have any prior knowledge regarding the knowledge areas of the problem. Figure 43 dictates that the system converged after epoch 700 and took a sufficient amount of time (over 10 minutes).

Table 16 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	CHIMACS		59.25		0.151
SCF – Prob. Estimate Simple			86.57		0.08
SWF – Prob. Estimate Simple			85.18		0.073
SCF – Trust Estimate		85.96	87.42	11.584	9.829
SWF – Trust Estimate		87.89	87.88	13.236	10.464

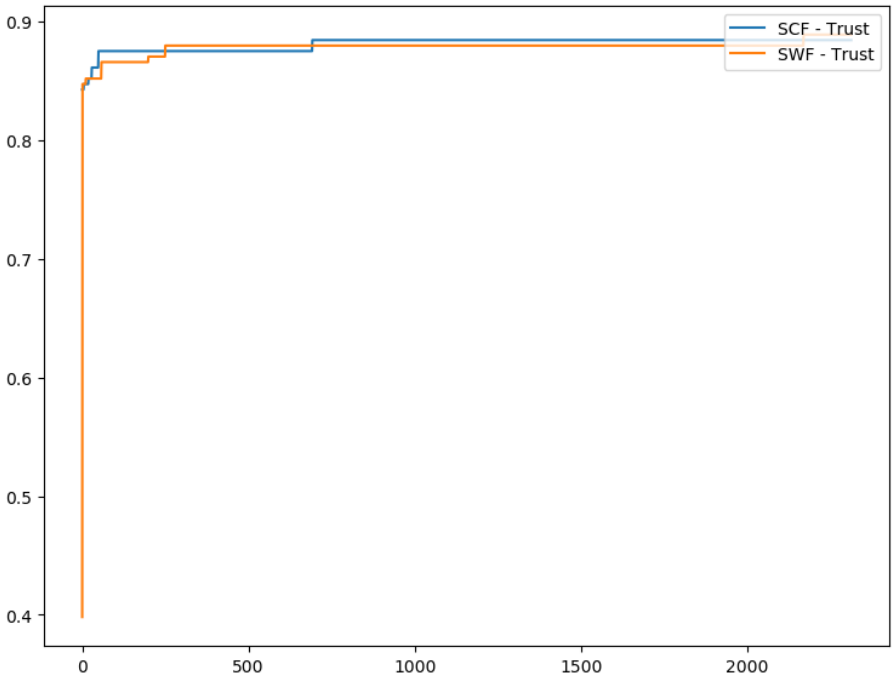


Figure 36 – CHIMACS SWF & SCF Accuracy Performance

Enabling CHIMACS to know about the related knowledge areas from cluster analysis it did not help it to perform much better. SC function increased accuracy just a little bit more than 1.53% compared to the voting function before. Also, it took more time for the system to converge to an optimal state.

Table 17 - Accuracy Rates (%) and Testing Time for C-CHIMACS

Voting	Set-up	Accuracy, Train (%)	Accuracy, Test (%)	Training Time, s	Testing Time, s
SWF – Simple	C - CHIMACS		59.25		0.151
SCF – Prob. Estimate Simple			86.57		0.08
SWF – Prob. Estimate Simple			85.18		0.073
SCF – Trust Estimate		91.35	88.95	7.365	11.846
SWF – Trust Estimate		86.98	85.58	12.879	9.235

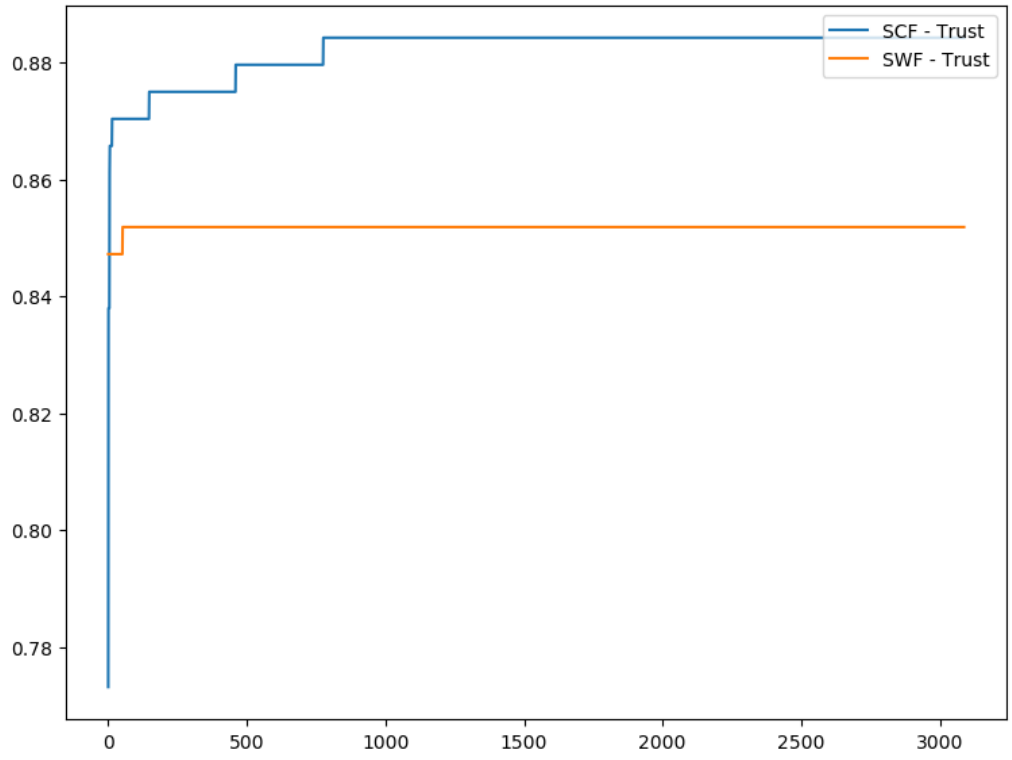


Figure 37 – C - CHIMACS SWF & SCF Accuracy Performance

A significant increase is observed by SCF in table 18 compared to individual classifiers (5%). On the other hand, there was an increase of only 1.05% in accuracy, compared to the rest of kernels. The system was let to run over 6000 epochs, despite the fact that it reached its peak around epoch 1100.

Table 18 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	DN-CHIMACS		59.25		0.156
SCF – Prob. Estimate Simple			86.57		0.546
SWF – Prob. Estimate Simple			85.18		0.08
SCF – Trust Estimate		88.89	90.0	17.26	15.125
SWF – Trust Estimate		88.12	88.0	8.398	5.378

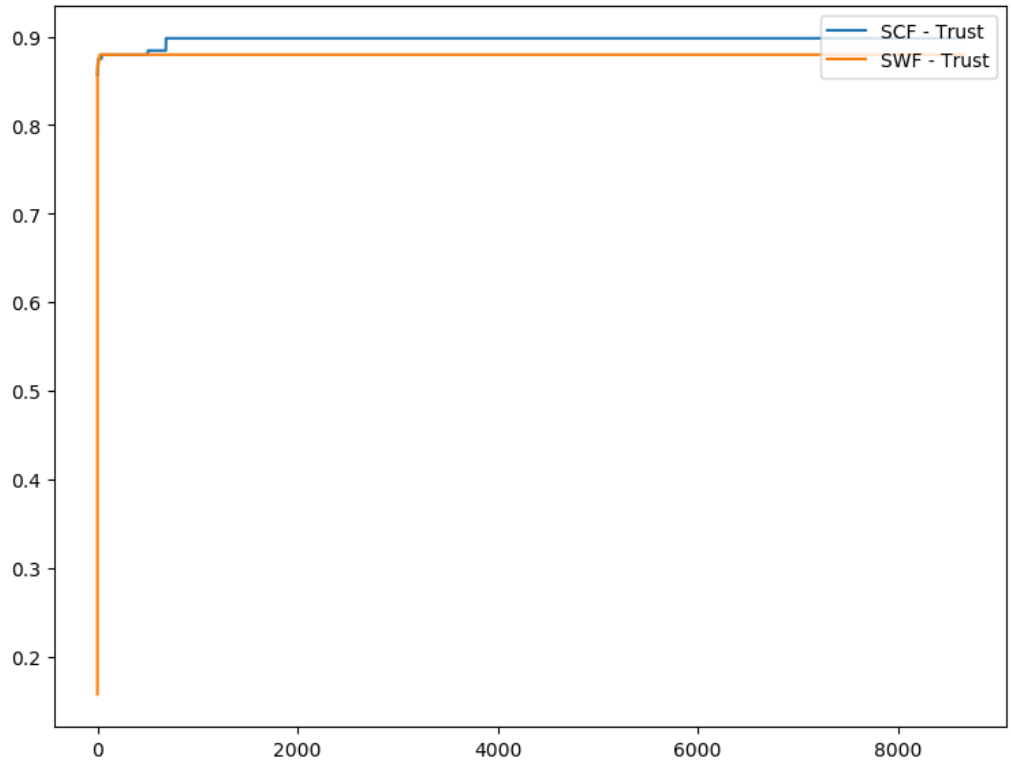


Figure 38 - CHIMACS SWF & SCF Accuracy Performance

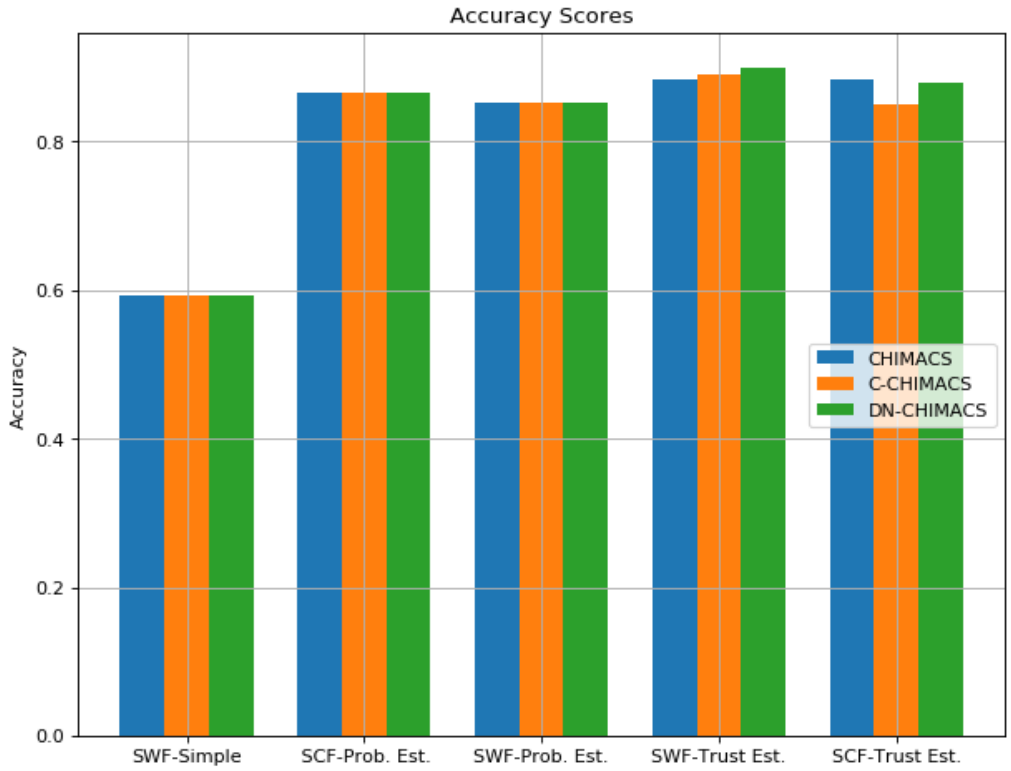


Figure 39 – Accuracy rates (%) of the framework for the Credit Approval problem

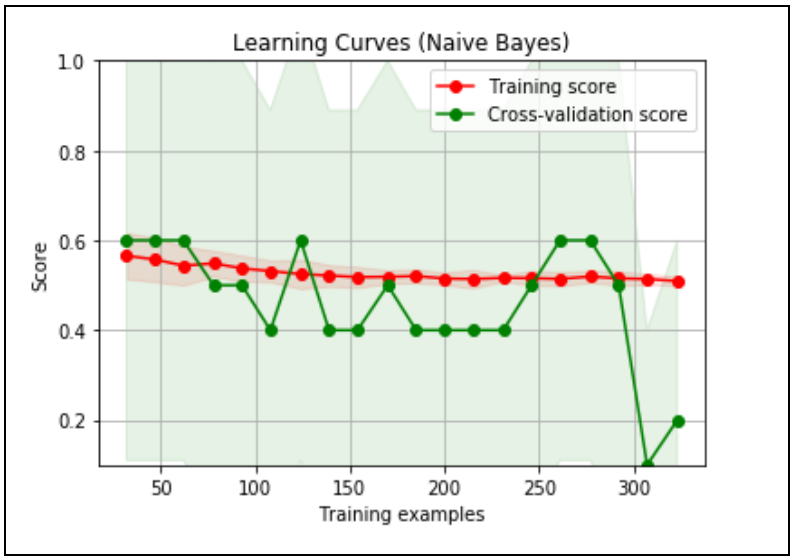
Although the results from cluster analysis indicated that there are two distinct cluster areas where the observations belong to, the best results were yielded from the last version of the proposed framework. The results from SWF function are not far from the results yielded from the previous version. In my opinion, both clustered and dynamic CHIMACS are capable of capturing and fusing diversity in the error rates of the agents. However, k-NN kernel was also able to capture part of the noise in the dataset and assign it to the appropriate agents.

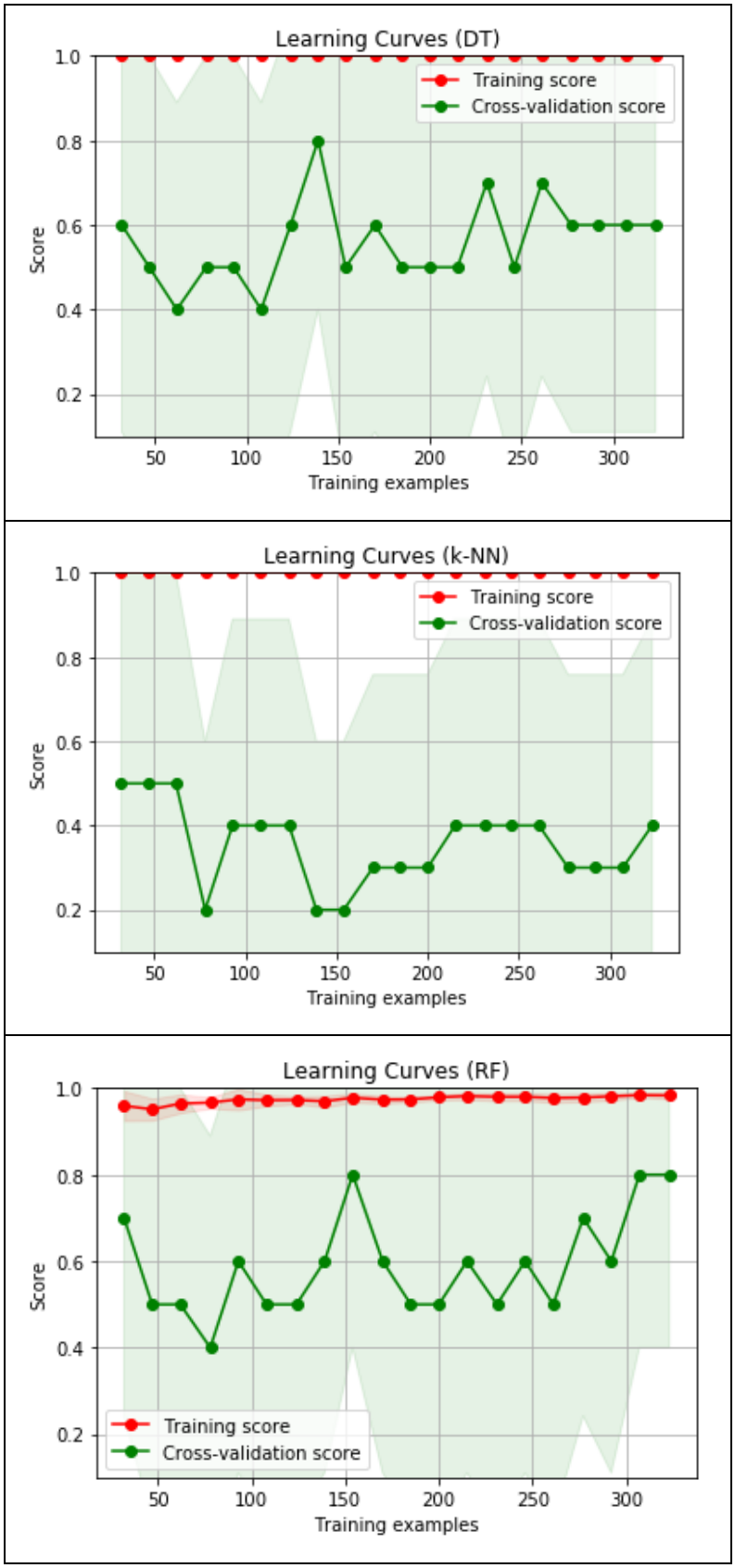
5.3.1.5 Hill-Valley

Each record from this dataset represents a two-dimensional graph. Each observation includes 100 features. Plotting the two-dimensional points in order will eventually form either a “bump” (Hill) in the plot or “dip” in the terrain (Valley),(Lichman, 2013). The prediction task is to classify whether the points form a hill or a valley. The dataset size has been reduced to 45% of its original size.

Figure 47 and table 19 below include the results from training and testing samples accordingly. The trained estimators are clearly underperforming. High training and relatively the same magnitude of validation error indicate that the model is biased. On the other hand, low training and very high validation errors indicate models with high variance.

Figure 40 - Training Results




Learning Curves (DT)
Training score
Cross-validation score
Score
Training examples
1.0
0.8
0.6
0.4
0.2
50
100
150
200
250
300
Learning Curves (k-NN)
Training score
Cross-validation score
Score
Training examples
Learning Curves (RF)
Training score
Cross-validation score
Score
Training examples

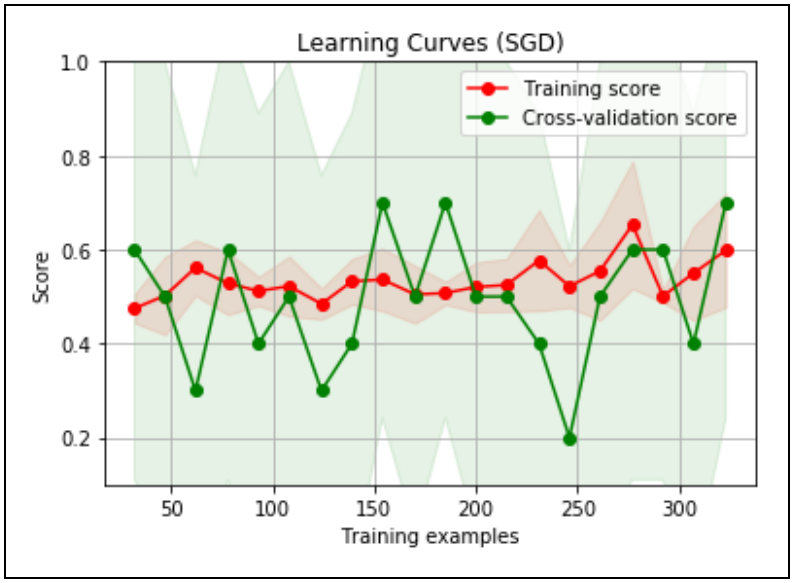


Table 19 - Accuracy rates (%) on the test set, training and testing time in milliseconds

Agent	**Accuracies**	**Training Time, s**	**Testing Time, s**
Naïve Bayes	42.17	0.07	0.004
k-NN	43.41	2.31	0.009
DT	47.76	0.113	0.084
RF	49.24	0.265	0.244
SGD	48.17	0.387	0.171

The results of table 19 above do not follow the statement in section 2.2.1. More specifically, I stated that the individual agent members need to yield error rates at most 50%. However, that does not hold for the specific agents, since in some cases it is observed almost 60% error rate. Despite that statement, it was interesting to observe how the proposed approach behaved and evaluate the results. Thus, that case was not discarded form the experiments.

Table 20 - Accuracy Rates (%), Training and Testing Time for CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	CHIMACS		52.17		0. 008
SCF – Prob. Estimate Simple			52.39		0.03
SWF – Prob. Estimate Simple			57.76		0.093
SCF – Trust Estimate		62.47	64.0	11.584	8.136
SWF – Trust Estimate		61.89	65.0	13.236	11.534

Figure 48 below describes the increase in accuracy. It is observed an increase in accuracy by 5.76% compared to the rest of the voting procedures. Although the accuracy level is still low, the proposed approach managed to increase it significantly. The reason that the proposed system performed better relies on the ability to exploit the diversity in error rates inside the MAS environment through voting and trust.

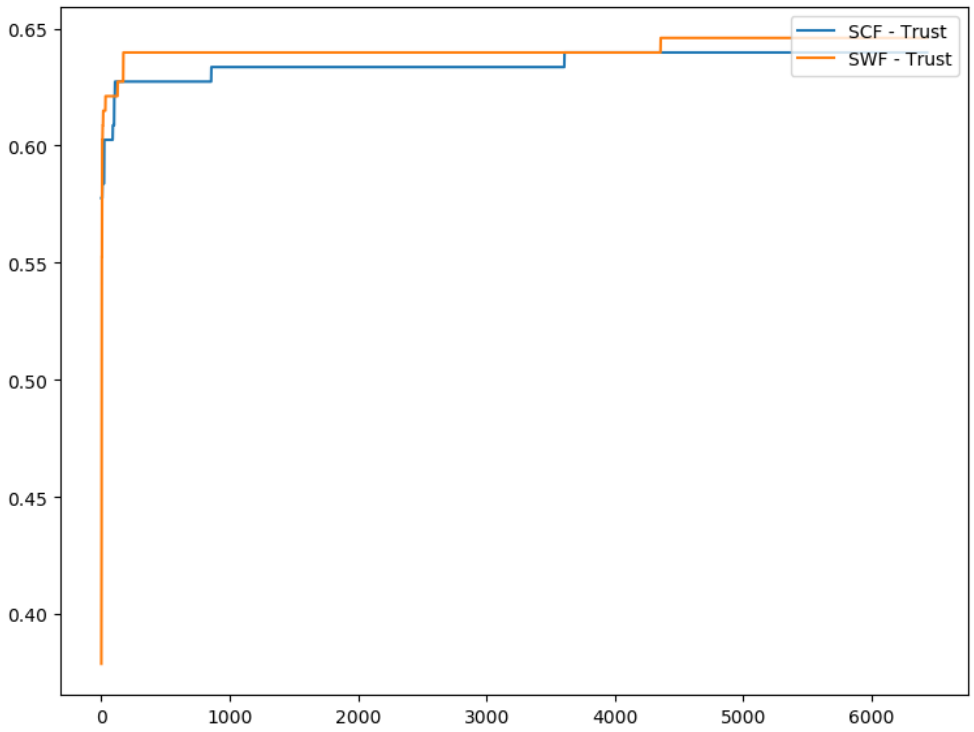


Figure 41 - CHIMACS SWF & SCF Accuracy Performance

Figure 48 and 49 presented the progress of the trust value associated with the performance of LR agent. The first figure describes trust related to SCF voting function, while the second is related to SWF voting function. In the first case, we observe a positive increase in trust measurement which indicated that LR agent is performing quite well as regards the observations belonging to cluster 0. On the other hand, the same agent in the second figure presents a significant negative value in trust.

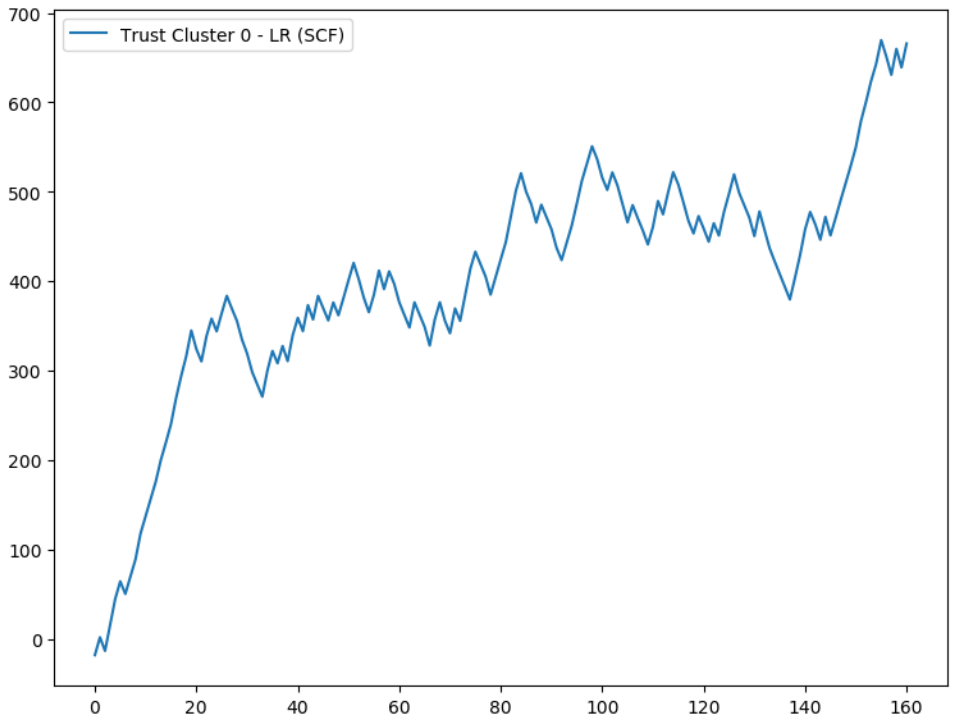


Figure 42 - LR Trust Progress for Cluster 0

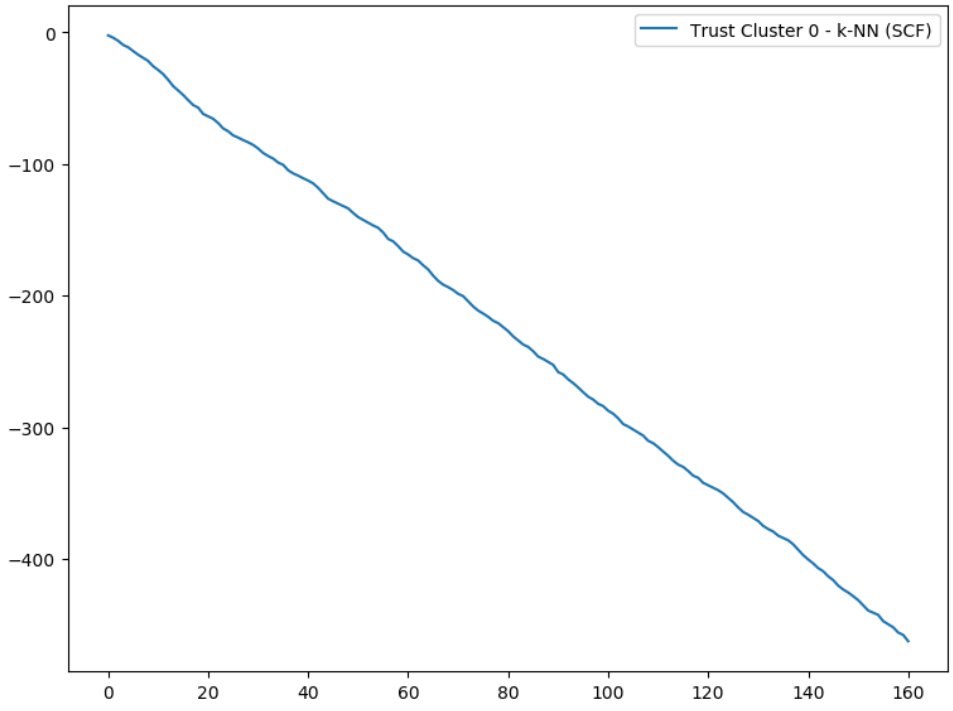


Figure 43 - – k-NN Trust Progress for Cluster 0

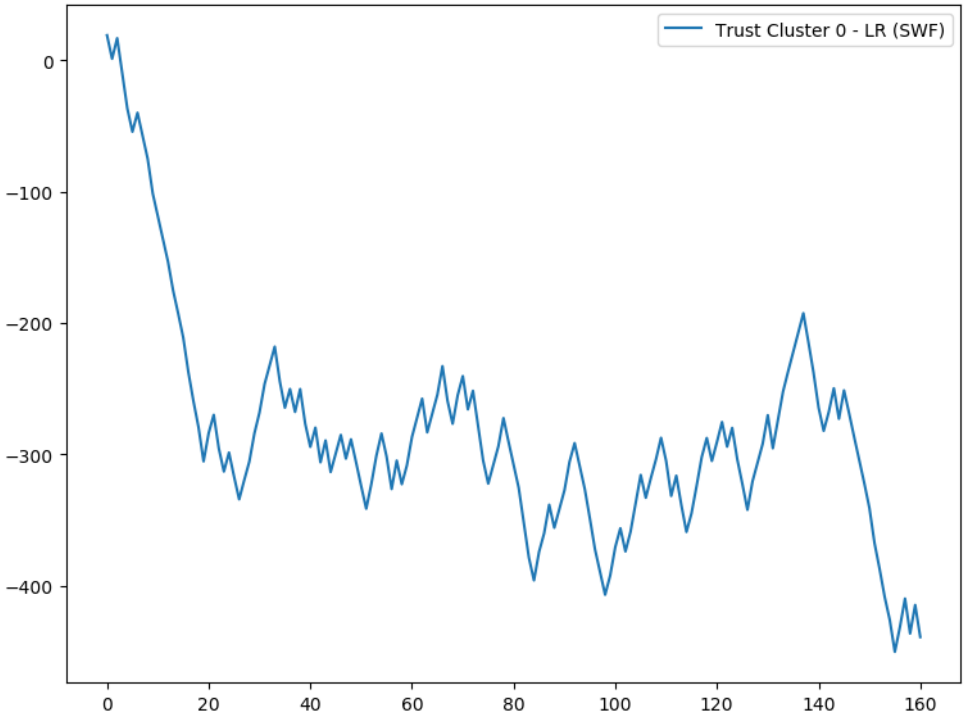


Figure 44 - LR Trust Progress for Cluster 0

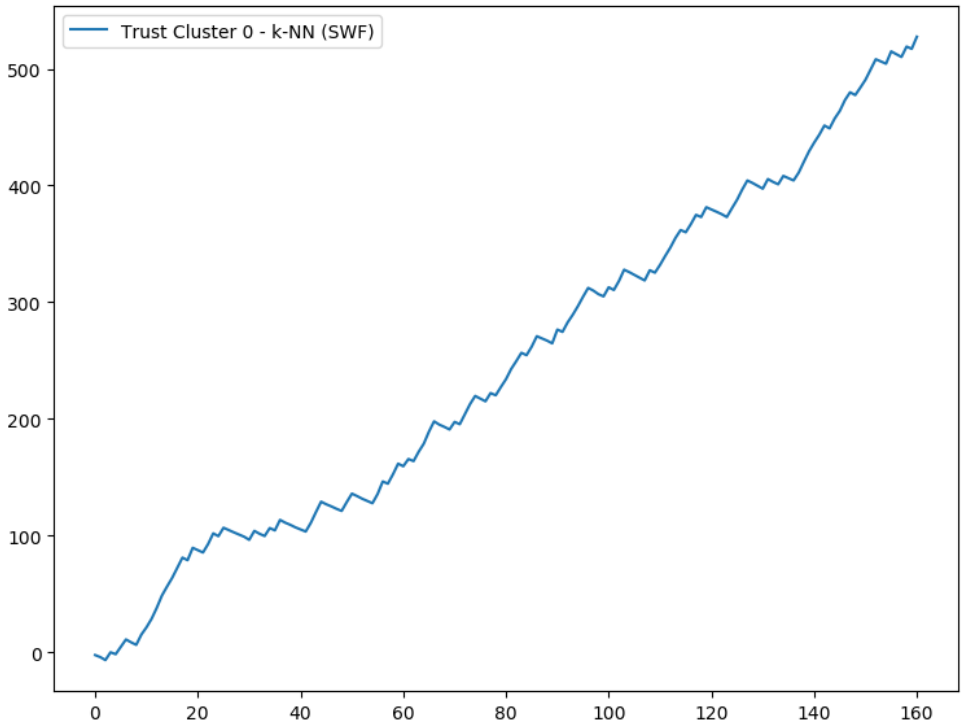


Figure 45 – k-NN Trust Progress for Cluster 0

We can securely infer that in both cases trust has different impact, which is closely related to the current gamma factor and the voting function. At the same time,

transparency and interpretability of the results help the expert to evaluate the produced results and configure the related parameters accordingly.

Clustered CHIMACS presented better results in accuracy in regard to SCF voting function. Also, it increased by 1% compared to naïve CHIMACS. The system was allowed to run for more time, thus it converged into higher level of accuracy.

Table 21 - Accuracy Rates (%), Training and Testing Time for C-CHIMACS

Voting	**Set-up**	**Accuracy, Train (%)**	**Accuracy, Test (%)**	**Training Time, s**	**Testing Time, s**
SWF – Simple	C - CHIMACS		52.17		0. 008
SCF – Prob. Estimate Simple			52.39		0.03
SWF – Prob. Estimate Simple			57.76		0.093
SCF – Trust Estimate		63.42	66.0	13.821	16.012
SWF – Trust Estimate		63.81	65.21	8.634	11.003

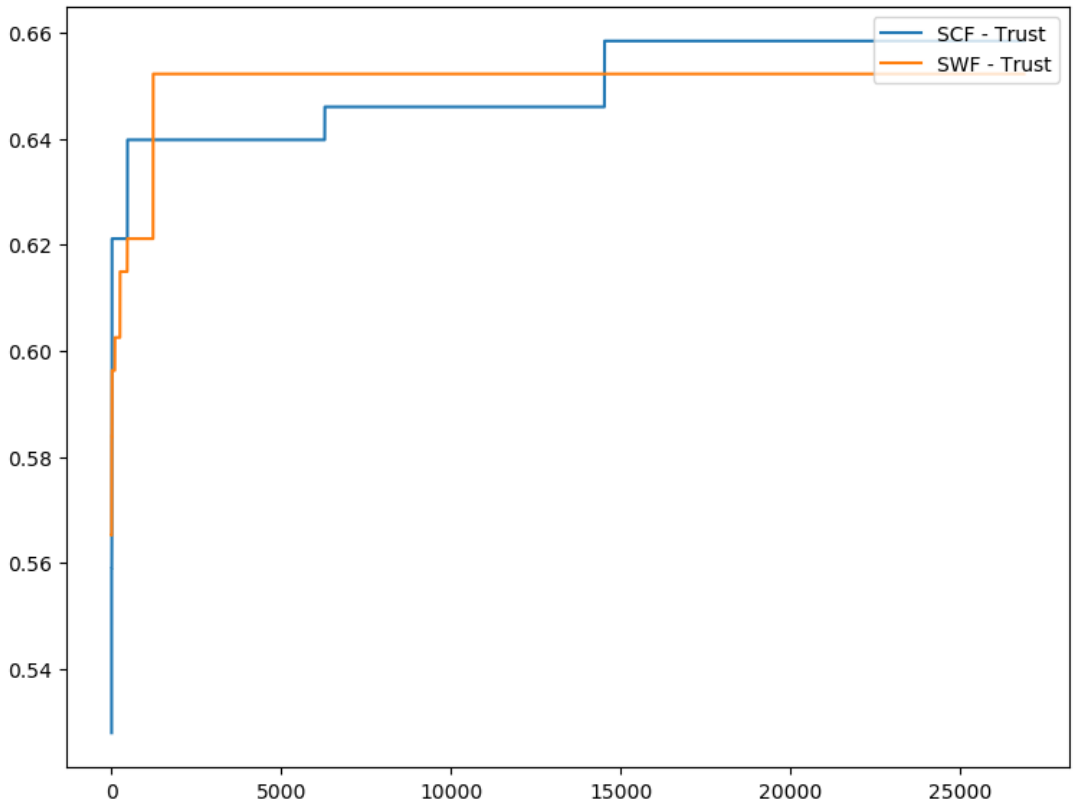


Figure 46 – C - CHIMACS SWF & SCF Accuracy Performance

Figures 54 and 55 present the trust levels both for k-NN and RF ML algorithms for three knowledge areas (clusters) during execution. The same pattern appears in the progress of trust. SCF voting function presents a high positive trust reward and a significant negative for k-NN in regard to cluster 0, while trust levels for the rest of the clusters remain in similar levels for both voting functions.

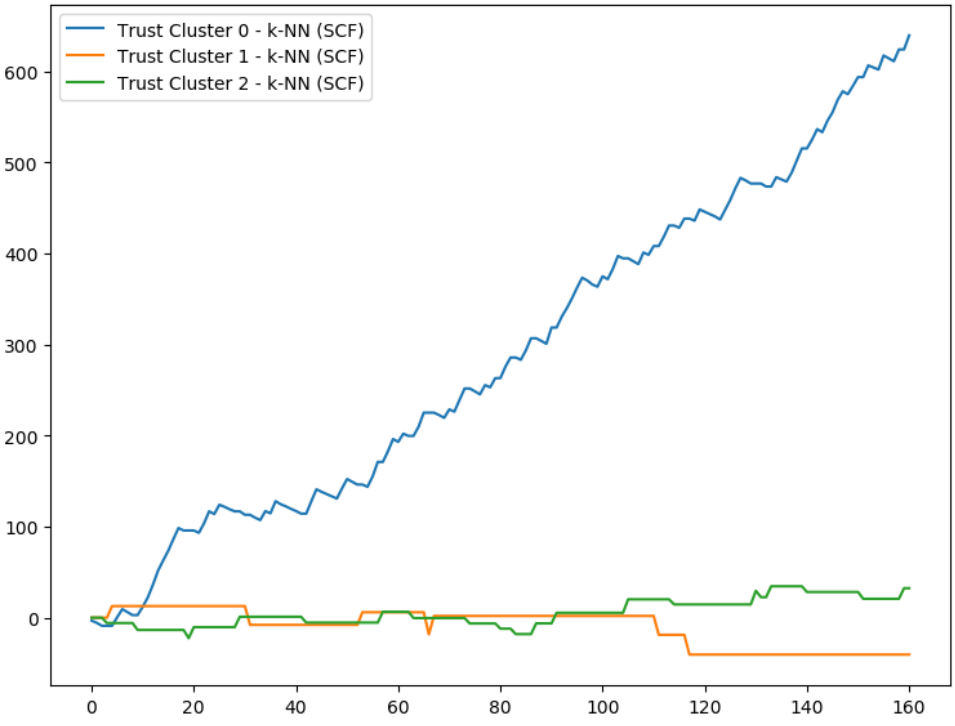


Figure 47 - k-NN Trust Levels for all clusters

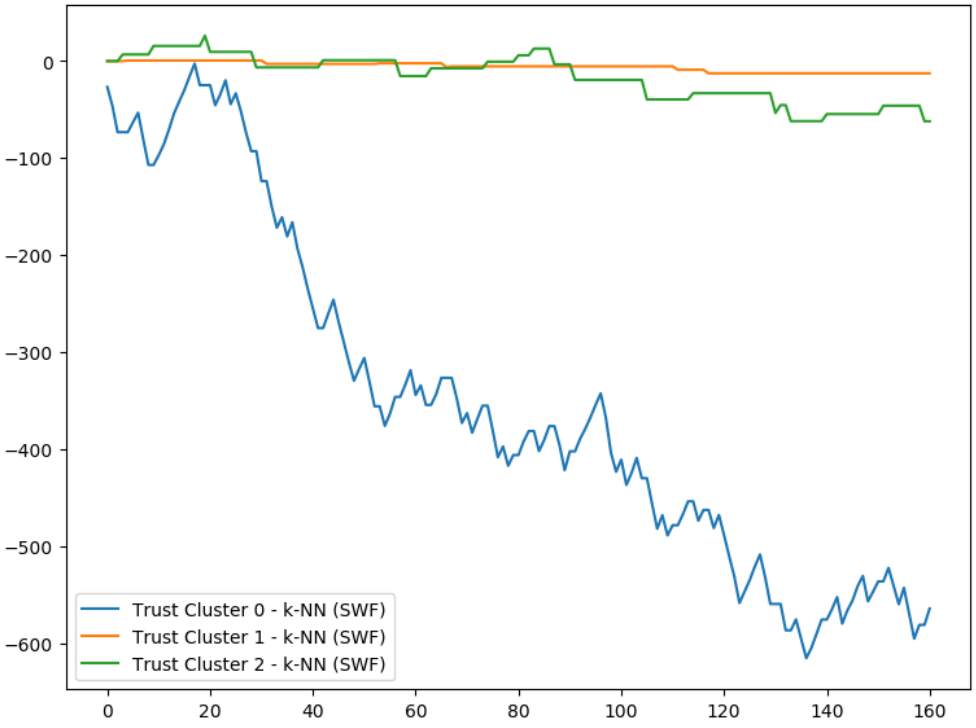


Figure 48 - k-NN Trust Levels for all clusters

Relatively different results are presented in Figures 56 and 57. RF ML algorithm performed better with data samples that come from cluster 0. However, around observation 90, it performed better for data that belongs to cluster 1.

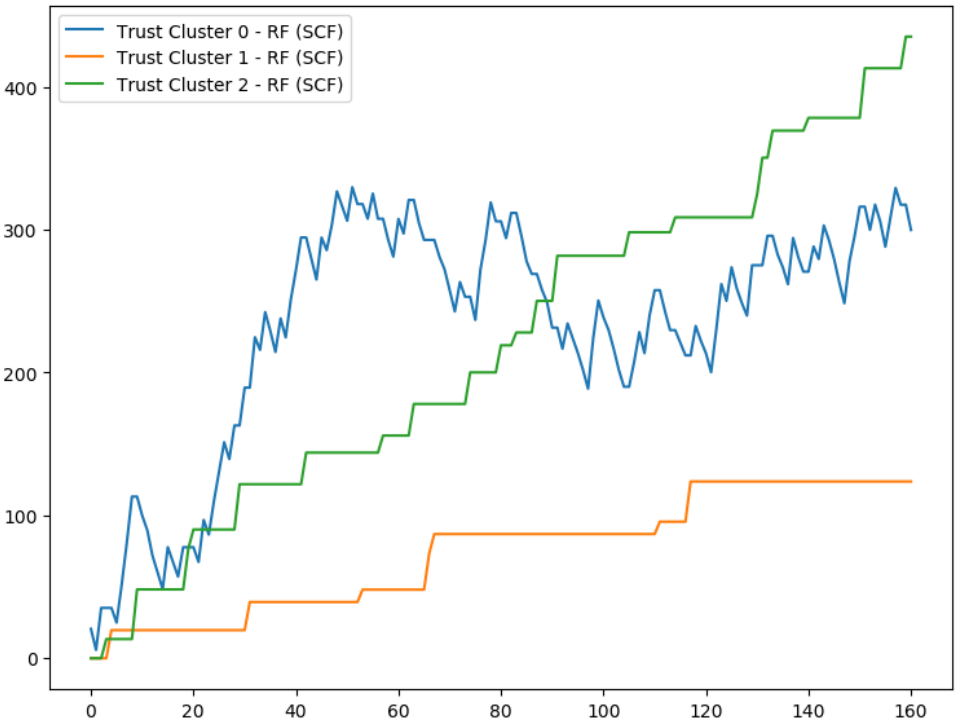


Figure 49 - RF Trust Levels for all clusters

Visualization of SWF function in Figure 57 indicates that trust levels for cluster 2 are almost the same as before. Although trust for data samples belonging to cluster 0 and 1 remain in high positive levels, the algorithm presented better results for the observations belonging to the first cluster.

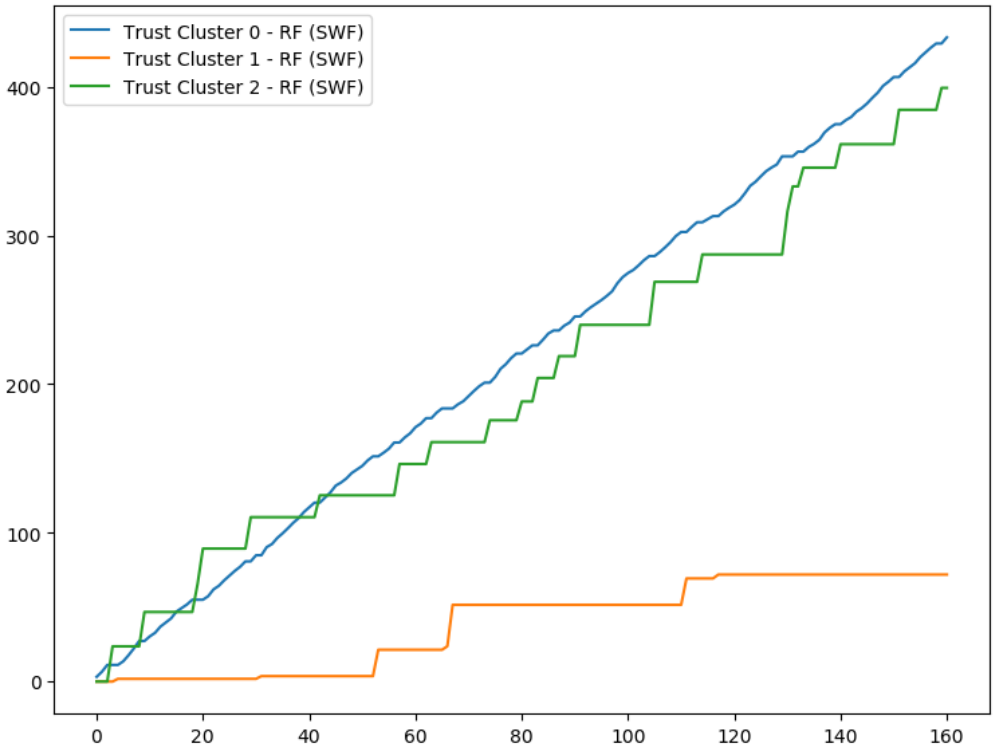


Figure 50 - RF Trust Levels for all clusters

The results in table 22 and the accuracies diagram in Figure 58 indicate that k-nearest neighbours kernel performed almost the same with the rest of kernel implementations, although DN-CHIMACS was expected to perform better than clustered-kernel, accuracy decreased by 1.41% regarding SCF voting. The most reasonable explanation is the distance metric employed which probably is not the most appropriate for the specific data structure.

Table 22 - Accuracy Rates (%) and Testing Time for CHIMACS

Voting	Set-up	Accuracy, Train (%)	Accuracy, Test (%)	Training Time, s	Testing Time, s
SWF – Simple	DN-CHIMACS		52.17		0. 008
SCF – Prob. Estimate Simple			52.39		0.03
SWF – Prob. Estimate Simple			57.76		0.093
SCF – Trust Estimate		61.89	64.59	7.897	11.367
SWF – Trust Estimate		62.12	63.35	7.162	16.143

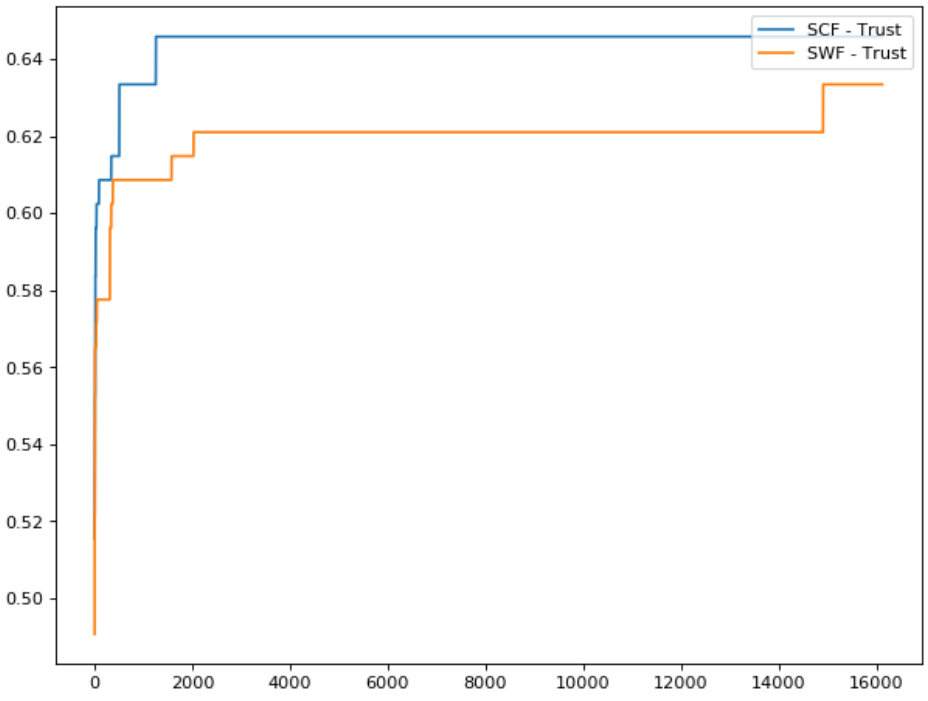


Figure 51 – DN - CHIMACS SWF & SCF Accuracy Performance

Figure 59 and 60 depict trust progress during execution. It is obvious that there is a diversity in trust levels, which is related to the partial randomness of the gamma factor and the performance of each individual classifier. Despite that, RF ML algorithm presented a positive linear increase in both cases.

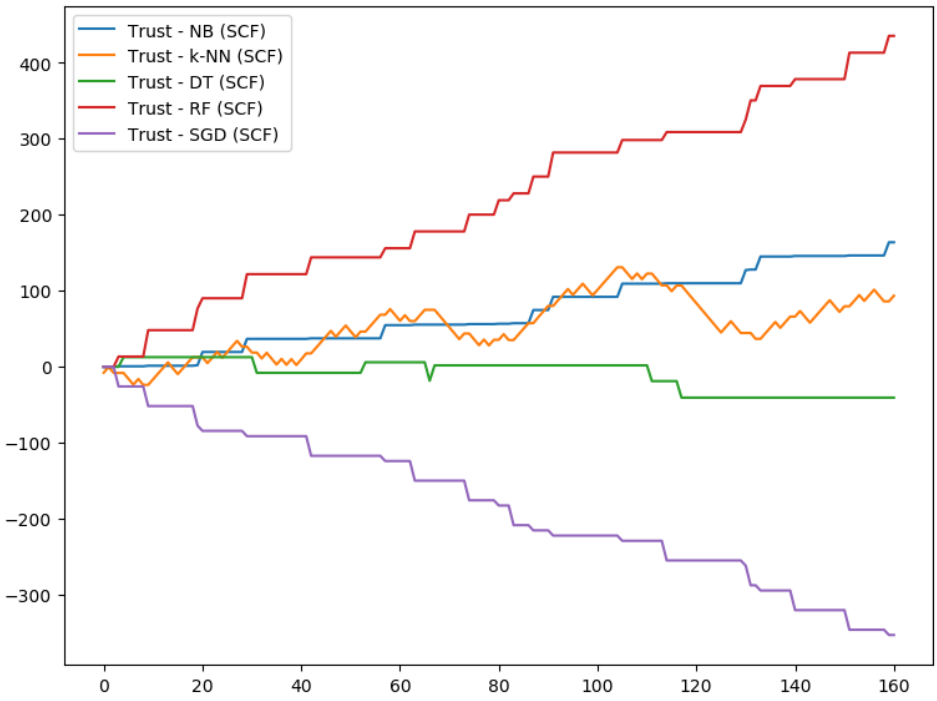


Figure 52 - Trust Levels for SCF

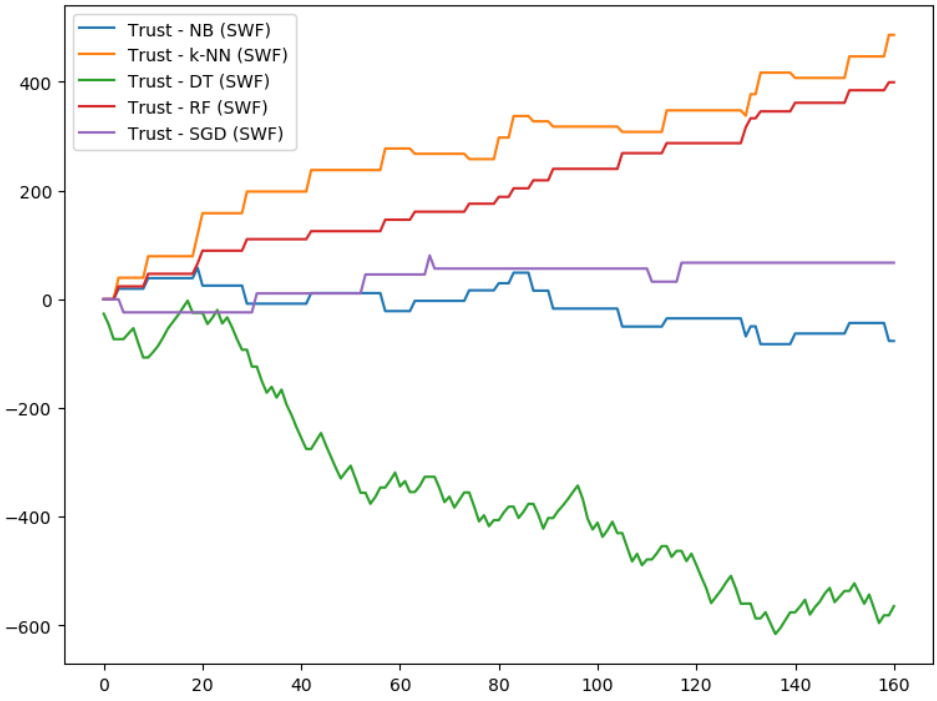


Figure 53 - Trust Levels for SWF

The bar chart in Figure 61 describes the performance of all voting functions. Although the performance of the agent's classifiers is worse than random, we can

securely infer that the proposed approach reduced bias by 17%. The overall classification accuracy still cannot be trusted since it reached only up to 66%. However, the results are very promising and future research can help increase the performance even further.

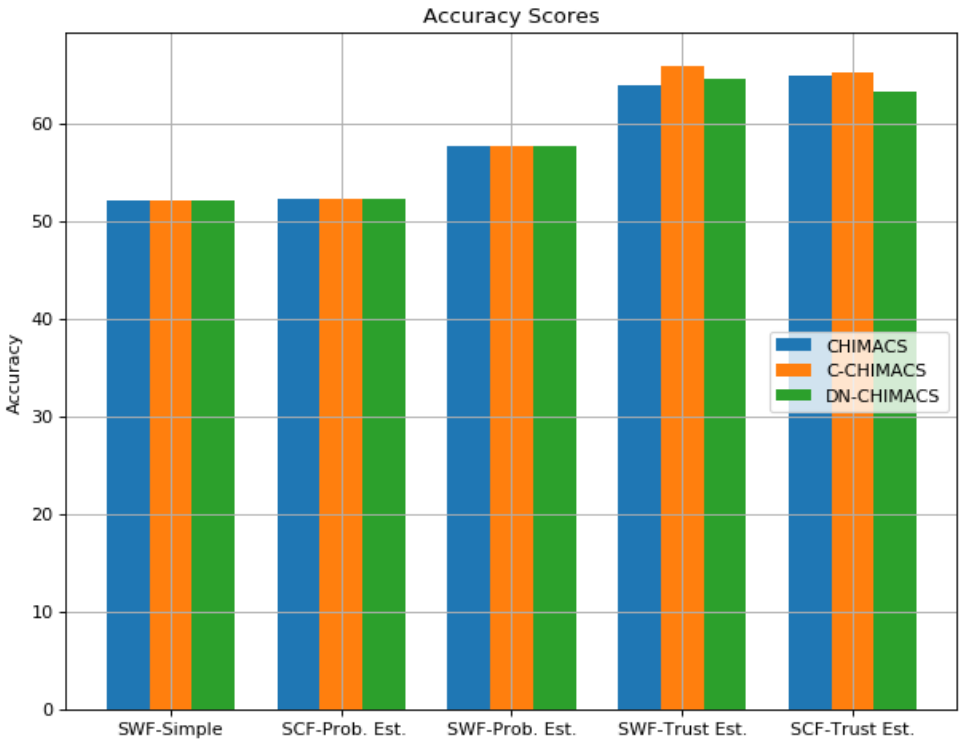


Figure 54 - Accuracy rates (%) of the framework for the Hill-Valley Problem

Cluster analysis showed that there are at least three distinct clusters inside the dataset. Hence it is reasonable to expect that C-CHIMACS would present better results in accuracy as it did. Social scoring function prevailed, yielding 66% accuracy which is 24% increase from the worst individual algorithm. At the same time, investigating the results yielded from the individual ML agents it is observed that k-NN performed 1% better than the worst agent. That could provide a possible explanation as to why k-NN based kernel did performed as expected.

5.3.2 Recall, Precision, Specificity Boost

5.3.2.1 Medical Datasets – ILPD Dataset

Indian Liver Patient Disease dataset contains 583 data observations in total, where 416 of the records are liver patient records and 167 non-liver patient records. At the same time, it contains 441 male patient records and 142 female patient records. It is a binary classification problem, where the task is to predict whether someone is a liver patient or not. Datasets in this case study are imbalanced and accuracy does not yield any valuable or trustworthy information.

Figure 62 visualizes class distribution. It is obvious that accuracy is not a useful metric. In this case, the true positive rate that measures the proportion of positives that are correctly identified as such, in other words sensitivity or recall, seems more pertinent to boost.

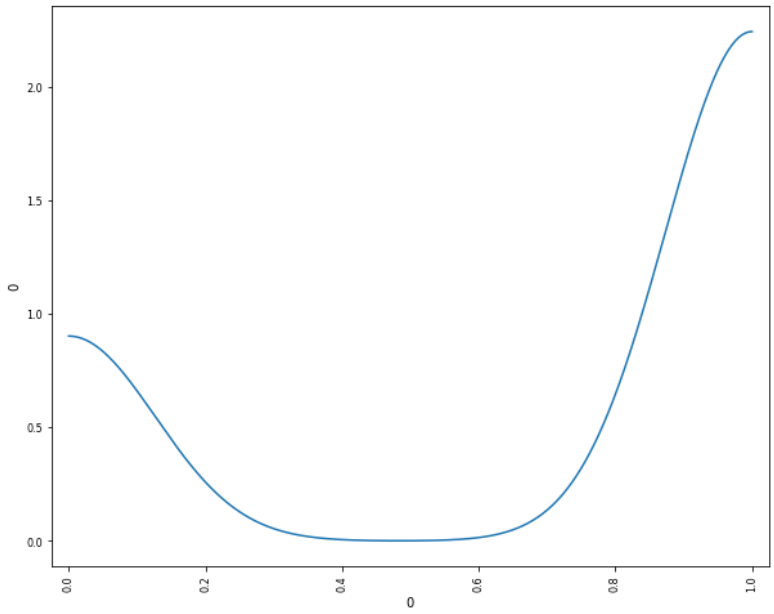


Figure 55 - Class Distribution

Also, the performance of some of the trained models employed is bad. The fraction of positive predictions among the total number of positive class values predicted (PPV) is very low in many cases. Some models like LR yield recall equal to 1, however it "is lying", since the specificity equals to 0.0.

Table 23 – Accuracy, Sensitivity, Specificity and Precision rates (%) on the test set, training and testing time in milliseconds

Agent	**Accuracies**	**Sensitivity**	**Specificity**	**Precision**	**Training Time, s**	**Testing Time, ms**
LR	73.14	1.0	0.0	73.14	0.006	0.001
Bagging	71.99	95.13	8.5	73.39	0.365	0.003
Adaboost	70.85	78.15	51.06	81.30	0.789	0.005
Random Forest	74.28	85.15	44.68	80.07	0.896	0.002
SGD	73.14	1.0	0.0	73.14	0.359	0.06

There is an increase of 1% in accuracy yielded from the second voting function. However, significant increase in recall, specificity and precision presented the voting functions that are influenced by trust reward. SCF with trust estimate yields 1.0 recall and 81.25% specificity, while precision is increased by 3.29% compared to the best individual model

Table 24 - Performance Rates (%) for Training and Time

Voting	**Set-up**	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**	**Precision (%)**	**Training Time, s**
SCF – Trust Estimate	CHIMACS	74.12	80.87	15.25	65.89	10.985
SWF – Trust Estimate		69.94	75.36	54.49	65.94	11.721

Table 25 - Performance Rates (%) and Testing Time

Voting	**Set-up**	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**	**Precision (%)**	**Testing Time, s**
SWF – Simple	CHIMACS	73.14	1.0	0.0	73.14	0.078
SCF – Prob. Estimate Simple		73.14	1.0	0.0	73.14	0.08
SWF – Prob. Estimate Simple		73.70	99.21	0.0	72.98	0.098
SCF – Trust Estimate		79.42	89.84	4.25	72.53	8.648
SWF – Trust Estimate		77.14	80.46	17.02	72.53	8.848

Significant increase was observed in all performance metrics by feeding the system the number of clusters. Both voting functions influenced by trust estimate performed very well, with SCF yielding 6.54% increase in accuracy and 36.17% in specificity compared to random forests, while precision increased by 7.42% compared to support vector machines. On the other hand, SWF boosted accuracy by 5.53%, sensitivity 1.0, specificity and precision increased by 6.39% and 9.61% compared to the specificity and precision rate accordingly of SCF.

Table 26 - Performance Rates (%) ant Training Time

Voting	**Set-up**	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**	**Precision (%)**	**Testing Time, s**
SCF – Trust Estimate	C - CHIMACS	70.14	56.71	78.02	36.23	11.65
SWF – Trust Estimate		70.28	85.47	22.96	67.45	15.42

Table 27 - Performance Rates (%) Rates (%) and Testing Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SWF – Simple	C - CHIMACS	73.14	1.0	0.0	73.14	0.078
SCF – Prob. Estimate Simple		73.14	1.0	0.0	73.14	0.08
SWF – Prob. Estimate Simple		73.70	99.21	0.0	72.98	0.098
SCF – Trust Estimate		77.14	86.71	17.02	75.15	13.648
SWF – Trust Estimate		74.28	96.87	12.76	73.99	13.648

Although the system was allowed to run for a substantial amount of time, k-NN based kernel reached almost the same performance levels as the clustered kernel. It run for 17.500 epochs but the blackboard failed to exploit further the diversity in classifiers. That is entirely related to the similarity retrieval function which most probably needed further optimization as a hyper-parameter.

Table 28 – Performance Rates (%) and Training Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SCF – Trust Estimate	DN-CHIMACS	76.65	80.23	25.52	73.56	8.361
SWF – Trust Estimate		68.25	85.69	19.86	73.89	7.158

Table 29 - Performance Rates (%) and Testing Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SWF – Simple	DN-CHIMACS	73.14	1.0	0.0	73.14	0.078
SCF – Prob. Estimate Simple		73.14	1.0	0.0	73.14	0.08
SWF – Prob. Estimate Simple		73.70	99.21	0.0	72.98	0.098
SCF – Trust Estimate		77.14	93.75	2.12	73.56	7.361
SWF – Trust Estimate		72.57	1.0	2.36	73.89	8.489

It is clear enough that the first two voting functions are biased towards true positive rate function yielding specificity results equal to zero. That means the voting procedures are entirely skewed to the left side of the distribution implying that all the test patients are sick. Results from the last table present that the dynamic version of

the proposed framework reached 1.0% recall where specificity 2.36. This implies that the framework generalized efficiently employing SWF voting function. Silhouette analysis showed that there is not consistency within clusters of data. Hence the data objects did not lie well within its cluster, providing insight on why the clustered version did not perform so well and k-NN did.

5.3.2.2 Heart Disease – SPECT Dataset

The dataset describes the diagnosis of cardiac Single Proton Emission Computed Tomography (SPECT) images. The class attribute of a patient is divided into two categories: normal (0) and abnormal (1). The dataset contains 267 SPECT image sets from patients. Feature extraction to summarize important features of the original SPECT images was performed. As a result, each patient is described by 44 continuous feature patterns. UCI ML repository already provides the examined dataset split into training (80 instances) and testing set (187 instances).

The class distribution in Figure 63 indicates that abnormal heart behavior dominates. The results form training and test samples indicate that some agents are overfitted or under-fitted and show a tendency to classify patients as abnormal most of the times (high positive rate). Hence, one looks to reduce the high false positive rate. The performance metric targeted for boosting by the system is specificity.

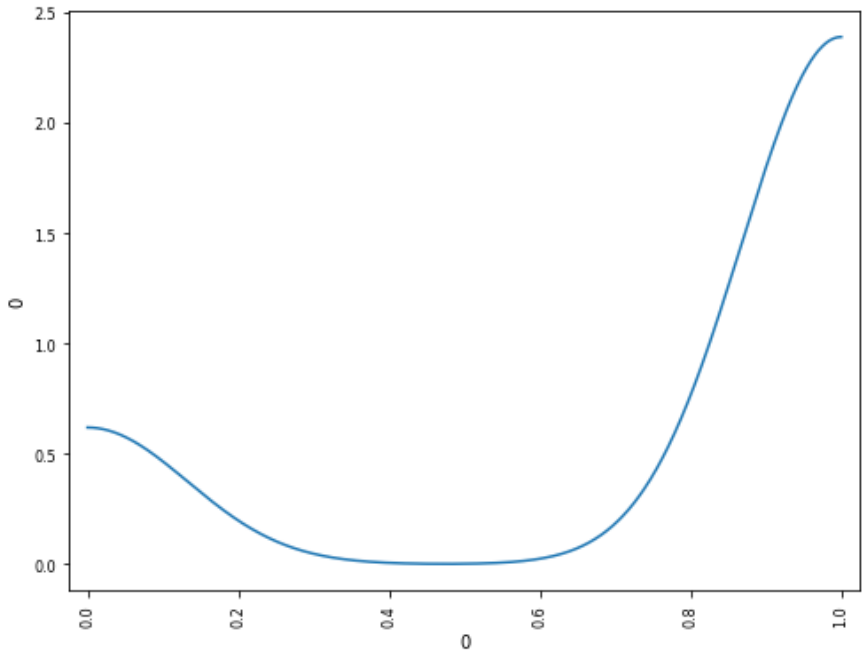


Figure 56 – Class Distribution

Table 30 – Accuracy, Sensitivity, Specificity and Precision rates (%) on the test set, training and testing time in milliseconds

Agent	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**Training Time, s**	**Testing Time, s**
LR	60.49	77.41	50.0	48.97	0.003	0.045
Bagging	69.13	54.38	78.0	54.38	10.81	0.073
Adaboost	64.19	64.51	64.0	64.51	0.04	0.065
Random Forest	71.40	58.06	80.0	64.28	0.42	0.452
SGD	67.90	32.25	90.0	66.66	0.006	0.866

As described in case study II, the evaluation of the trust function does not consider only the validity of the answer but also the class of the prediction. Although the accuracy from the first evaluation method increased by 11.71% compared to the

worse classifier, the desired metric did not present much better results. The table below presents the results from the second evaluation method where trust update considers the class of the predicted value.

Table 31 - Performance Rates (%) for Training and Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Training Time, s
SCF – Trust Estimate	CHIMACS	73.21	30.45	70.0	30.47	4.945
SWF – Trust Estimate		74.98	25.32	67.45	39.35	7.458

Table 32 - Performance Rates (%) and Testing Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SWF – Simple	CHIMACS	51.23	1.0	0.0	0.0	0.147
SCF – Prob. Estimate Simple		65.14	53.21	12.58	19.54	0.03
SWF – Prob. Estimate Simple		67.0	70.12	54.21	40.25	0.093
SCF – Trust Estimate		77.77	32.25	78.0	33.33	4.571
SWF – Trust Estimate		76.54	25.80	68.0	47.60	6.178

The results below indicate that providing CHIMACS an insight about the existing patterns in the data contributes to even more boosting of the examined performance metrics. Both accuracy and specificity increased compared to the naïve version, which did not outperform the best individual classifier.

Table 33 - Performance Rates (%) for Training and Time

Voting	**Set-up**	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**	**Precision (%)**	**Testing Time, s**
SCF – Trust Estimate	C-CHIMACS	77.12	20.45	79.74	30.47	4.945
SWF – Trust Estimate		64.20	19.35	78.98	39.35	2.458

Table 34 - Performance Rates (%) and Testing Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SWF – Simple	C-CHIMACS	51.23	1.0	0.0	0.0	0.147
SCF – Prob. Estimate Simple		65.14	53.21	12.58	19.54	0.03
SWF – Prob. Estimate Simple		67.0	70.12	54.21	40.25	0.093
SCF – Trust Estimate		79.01	25.80	80.0	25.0	6.548
SWF – Trust Estimate		71.60	6.45	88.0	44.44	5.867

Nearest neighbours kernel outperformed the rest of the implementations and the individual models regarding accuracy. Specificity on the other hand, did not increase more as expected. Time taken to converge both for training and testing is obviously longer without presenting better results in this case.

Table 35 – Performance Rates (%) and Training Time

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Training Time, s
SCF – Trust Estimate	DN - CHIMACS	75.81	14.35	65.0	32.78	5.778
SWF – Trust Estimate		78.36	23.80	68.90	29.77	7.986

Table 36 - Performance Rates (%) and Testing Time for DN-CHIMACS

Voting	Set-up	Accuracy (%)	Sensitivity (%)	Specificity (%)	Precision (%)	Testing Time, s
SWF – Simple	DN - CHIMACS	51.23	1.0	0.0	0.0	0.147
SCF – Prob. Estimate Simple		65.14	53.21	12.58	19.54	0.03
SWF – Prob. Estimate Simple		67.0	70.12	54.21	40.25	0.093
SCF – Trust Estimate		80.24	19.35	74.0	34.78	10.698
SWF – Trust Estimate		80.24	25.80	70.0	31.57	9.378

Enhancing specificity is the target of this classification task. The clustered version of the proposed system and more specifically social welfare function yielded the best results as far as specificity rate is concerned. However, the k-NN version of the system presented better results is accuracy levels. A possible explanation on why clustered CHIMACS performed better is that the data formed more distinct cluster boundaries thus given that information the system could identify efficiently the agent's regions of expertise. On the other hand, k-NN based version possibly presented better accuracy rates due to the distance metric, the dataset is consisted of binary type fields (0 or 1) hence the distance metric was capable of retrieving more similar data instances.

5.3.3 Conclusions

Boosting predictive performance through expertise fusion is challenging. In this Chapter, the challenge is addressed by an attempt to find an efficient general-purpose framework set-up for enhancing the overall performance by applying reinforcement learning techniques. For this purpose, firstly, a basic version of the proposed multi-agent framework is tested, secondly, a knowledge area aware of the blackboard kernel is introduced, and finally, an adaptive kernel based on k-NN retrieval method is tested.

The experiment includes two main parts. The first consists of datasets in which accuracy is considered a valid performance metric indicator. For that purpose, the system focused on boosting the accuracy by applying trust measurement while updating it through single validation. The second part consists mainly of highly imbalanced datasets, for which specificity, sensitivity and precision are the metrics boosted.

The conducted empirical study showed in most cases that the system can increase the overall predictive performance only if it is aware of specific information about the examined dataset. The clustered and k-NN based versions of CHIMACS demonstrated better results in terms of predictive performance, but the time to train both was longer compared to the simplest version of it. However, time taken is minimal compared to traditional ML algorithms like SVMs. Another important

conclusion is that the system is capable of achieving major increase in the global performance when its agent members are underperforming. For example, when the accuracy is higher than 90%, there is not enough room for major improvement. Another important factor is the diversity amongst the classifiers that constitute the ensemble (classifiers with different error rates are a necessary and sufficient condition for an ensemble). Transparency is one of the main contributions of this thesis. A meta-level explanation on why the system performed that is provided through trust levels. However, there is a certain level of uncertainty due to the fact that the system is consisted of traditional ML algorithms in its core. Practically, an in-depth description on how and why the system produced the desired results or not, is closely depended on the algorithms structure and the structure of the data of the problem examined. The table below, depicts the best performance metric results is presented:

Framework Version	Voting Function	Dataset	Accuracy (%)	Recall (%)	Specificity (%)	Precision (%)
		Waveform				
CHIMACS	SCF – Trust Estimate		93.90			
C-CHIMACS	SCF – Trust Estimate		94.30			
DN-CHIMACS	SWF – Trust Estimate		93.02			
		Wines				
CHIMACS	SCF/SWF – Trust Estimate		98.97			
C-CHIMACS	SCF – Trust Estimate		99.90			

DN-CHIMACS	SWF – Trust Estimate		99.58	
		Q&A		
CHIMACS	SCF – Trust Estimate		74.97	
C-CHIMACS	SCF – Trust Estimate		75.0	
DN-CHIMACS	SCF – Trust Estimate		78.93	
		Credit Risk		
CHIMACS	SWF – Trust Estimate		87.88	
C-CHIMACS	SCF – Trust Estimate		88.95	
DN-CHIMACS	SCF – Trust Estimate		90.0	
		Hill-Valley		
CHIMACS	SWF – Trust Estimate		65.0	
C-CHIMACS	SCF – Trust Estimate		66.0	
DN-CHIMACS	SCF – Trust Estimate		64.59	

		ILPD				
CHIMACS	SCF – Trust Estimate		79.42	89.84	4.25	72.53
C-CHIMACS	SWF – Trust Estimate		74.28	96.87	12.76	73.99
DN-CHIMACS			72.57	1.0	2.36	73.89
		SPECT				
CHIMACS	SCF – Trust Estimate		77.77	32.25	78.0	33.33
C-CHIMACS	SWF – Trust Estimate		71.60	6.45	88.0	44.44
DN-CHIMACS	SCF – Trust Estimate		80.24	19.35	74.0	34.78

Table 37 - Best Performance Metrics Results

The main goal of this Chapter was to present the experimentation results and answer one important research question: whether knowledge fusion can be achieved by applying reinforcement learning methods. As a result, a number of framework set-ups are formulated. Firstly, a basic version is formed in order to achieve a fast but small boosting of the overall performance. To achieve better results but considerably slower predictions, one should consider a clustered or k-NN based approaches with a pattern extraction mechanism. Also, the experiments demonstrated that a dataset with well-defined structure did not challenge the system enough. The agents perform considerably well and there is not enough room for improvement. Moreover, using a reinforcement trust based approach allows one not only to achieve higher prediction performance but trust an agent inside a vast multi-agent environment and thus

providing a quality measurement. However, further experiments have to be conducted in order to determine the most efficient way to construct and train a reinforcement learning ML multi-agent environment.

5.3.4 Summary

In this Chapter, the results from experimentation were presented, analysed and discussed. This includes comparing, evaluating and then choosing the set-up that performs the best considering the requirements of the examined problem. Firstly, the proposed approach was applied on a series of datasets improving classification accuracy. Then, the performance of the proposed approach on a number of datasets was evaluated, where other performance metrics other than accuracy are improved. Imbalance is the main characteristic of these datasets; as a result accuracy does not produce any valuable inference as far as classification performance is concerned. Moreover, knowledge aggregation through voting functions and the impact they have in classification performance were explored. Finally, the time performance of the proposed approach was compared to the time taken both for training and testing of traditional ML algorithms used as agents. Clustered and Dynamic CHIMACS outperformed the traditional ML and ensemble algorithms in terms of accuracy, recall, specificity and precision but in most of the cases failed to converge to an optimal state in relatively low times.

5.4 CHIMACS vs Literature Work

5.4.1.1 Multi-Agent Systems

As to date there are only two MAS architectures similar to the one proposed in this thesis. Both of them employ reinforcement learning techniques to compute trust and a central entity to monitor and coordinate the actions inside the MAS environment. The first one utilizes Q-Learning techniques and the predictive power of neural networks (Pourpanah et al., 2017) and the second exploits the flexibility of CBR agents under a custom reinforcement learning function (Teodorescu & Petridis,

2013). The proposed MAS in (Pourpanah et al., 2017; Teodorescu & Petridis, 2013) presented successful results in identifying the most competent individual or group of agent's classifiers. However, both approaches performed as good as its best individual or group of agents from the environment without taking any further steps to enhance the overall predictive power. Another shared feature between the above implementation is the fact that they both use homogeneous agents. More specifically, the ML agents are either neural networks or CBR. Also, both approaches did not take any steps further to optimize the hyper-parameters that influence the trust update process. That includes learning rate a and gamma discount factor γ. Finally, one of the most important features that is not addressed from the research work above is the identification of distinct knowledge patterns inside the data. Until now, the data were fed to the agents as it is without the system taking into account the characteristics of the respected domains.

Current research work alleviates all the issues raised above through the main contributions to knowledge. The proposed approach implements a hybrid blackboard-based architecture capable of accommodating heterogeneous agents which enhances the diversity of the environment. The orchestration of the agent activities, performance monitoring and rest of the system processes is handled by a central manager by utilizing reinforcement learning techniques or more as it was named in this thesis, trust metric. Voting techniques were introduced both as a bargaining system and for aggregating the agent's knowledge. The combination of reinforcement trust and voting results in combining the acquired knowledge and the inherited heterogeneity to boost the global predictive performance. That is one of the major contributions of this work, because instead of nominating just an individual agent as an expert, like in the literature, the final output is produced by all of them. Finally, that last contribution is related to the optimization process. To converge to an optimal value related to the discount and learning factors, someone needs to apply optimization processes during the execution of the system. In the literature, the values for these parameters where manually selected, where is this thesis Thomson Sampling was applied. Proposed a novel application of Thomson Sampling algorithm for optimizing reinforcement signal reward of the trust function. At the same time, simulated annealing was employed to find near optimal weights for gamma factor parameter. Showed that the blackboard empirically converged faster.

5.4.1.2 Ensemble Methods

EM can be proven very effective considering that they can enhance the prediction accuracy of a group of ML algorithms using a variety of methods for exploiting the diversity in prediction error rates. In a way, the proposed approach can be considered as an advanced ensemble methodology since one of its contributions to knowledge is enhanced predictive force of high overfitted or biased models.

EM are considered black boxes and the influence an expert can have is only through the algorithm choice, parameterization of the algorithms and their data. The proposed approach has modular organization and is loosely coupled with its internal modules. Estimators and blackboard are considered that way, in the sense that their procedures do not interfere with each other. The agents act semi-autonomously based on the received input and not on any blackboard instructions. Blackboard takes into account the individual predictions and applies weights voting methods to modify the confidence of the agent suggestions based on the trust it has developed over time. However, it does not interfere directly with their decisions. Someone can say that the agents are cohesive and loosely decoupled at the same time. The utilized multi-agent architecture offers flexibility and scalability since an agent component can leave and another one can take its place or more can be added without interrupting the process.

Voting and reinforcement learning trust evaluation procedures are also an important advantage of CHIMAS compared to the ensemble. Some of the traditional methods perform majority voting procedures like voting ensemble for example. However, it is considered a black box, too. Transparency and interpretability in a meta-level are two of the major advantages of the proposed approach. One can always have access to the trust measurements and weights in case interpretation is important in terms of individual performance.

Taking into consideration the results, especially from the last experiment, we can securely infer that the proposed system performs better for specific problems. Both case studies deal with low or in some case large volume of data and noisy which as a result increases uncertainty as far as ML robustness is concerned. Traditional ML approaches fail to generalize well and as a result they do not perform well on unseen data. Hence, the proposed approach has the ability to exploit the strengths of each

individual agent and increase the overall classification performance. Volume and noise in data result in high uncertainty. Traditional ML algorithms fail to handle that inherited complexity which leads in bad generalization and poor performance. In these cases, the proposed approach has been proven that can handle these issues into a certain extent and improve the overall performance.

1. Proposed a novel approach to improve predictive force using a hybrid, intelligent blackboard based framework.
2. Introduced a novel approach of employing voting functions that works as a bargaining system between the blackboard and the ML agents. The agents act as an electorate and use five proposed majority voting functions. Majority SCF nominate one agent's answer as a winner and SWF one alternative over another.
3. Introduced a novel way of using reinforcement trust reward. The value calculated as a function of each agent performance influences the individual vote power. Trust measurement reflects the competence of the agent classifiers in accordance with the certain patterns revealed from the examined data.
4. Improved predictive force in terms of accuracy, recall, precision and specificity. Showed that by changing the evaluation of individual predictions provided by the agents, the blackboard is capable of improving recall, specificity and precision or accuracy only.
5. Proposed a novel application of Thomson Sampling algorithm for optimizing reinforcement signal reward of the trust function. At the same time, simulated annealing was employed to find near optimal weights for gamma factor parameter. Showed that the blackboard empirically converged faster.

Implemented an intelligent kernel for the blackboard. Proposed a novel approach of revealing and exploiting data patterns using the k-NN algorithm as the backbone of the blackboard supervisor. It provides blackboard the means to adapt by updating the knowledge in every epoch.

Chapter 6 Conclusions

6.1 Introduction

This Chapter summarizes the conducted research work of this thesis, presents the main contributions to knowledge achieved by answering the research questions stated in Chapter 1. Finally, a few ideas for further improvement and research work are described. The Chapter is structured as follows: in Section 6.2 an overall summary of this research work is presented; then the main contributions to knowledge are discussed in Section 6.3; finally, in Section 6.4 a few ideas for future research work and improvement are suggested.

6.2 Summary of Thesis

In this thesis, the problem of boosting prediction performance is investigated. This involves implementation of a trust metric based on reinforcement learning techniques, cluster analysis, agent training, voting, knowledge fusion and boosting prediction performance by exploiting diversity in error rate of the classifiers.

To begin with, the motivation for this research is stated in Chapter 1. This included the research questions and objectives, the methodology under which this research work conducted.

Secondly, literature review is presented in Chapter 2. This included an overview of how hybrid MAS are used for predictions and boosting tasks. Then, a short review of traditional EM is given and how they contribute in classification enhancement. Suitable architectures for constructing multi-agent environments were discussed. Finally, computational choice theory and reinforcement learning trust for enhancing prediction performance were presented.

Then, the individual components that constitute the proposed approach were described in Chapter 3. A brief description of the structure for voting, trust and the blackboard manager is presented (see Figures 16 and 17). Then the agent structure and workflow process is presented in Figure 18. A normalization process in the estimated probability of the individual predictions is applied. The purpose of that process is to twofold, firstly it gives the opportunity to the agents to provide predictions with a more realistic estimated probability and secondly it helps blackboard by making smaller steps during the learning process. Trust estimation and optimization are described in Section 3.4. Trust function, validation and update process along with the reinforcement signal optimization are discussed. The trust metric is formalized. At the same time, an optimization method called Thomson Sampling and discounted future update process are applied in a novel way to update the reward parameter and perform faster convergence. Finally, the definition and formalization of five SCF and SWF variations is given, as well as the impact they have in the overall prediction performance.

Proceeding, an approach for constructing a framework for prediction performance enhancement, more specifically a combination of linked modules for voting, trust calculation and evaluation, cluster analysis and knowledge fusion, is proposed in Chapter 4 (see Figures 20-25). The core of this approach is a novel blackboard based kernel proposed in this thesis. The proposed approach consists of three different kernels, the simple one, a cluster aware and a k-NN based. The proposed approach is capable of capturing properties of the individual classifiers in terms of error diversity as well as of combining the acquired knowledge. Furthermore, the blackboard is capable of evaluating the quality of the answers through the assignment of a trust value. This value facilitates the ability of the blackboard to evaluate through trial and error the competence of each classifier. Then, these values are used to influence the voting functions. In order to demonstrate the effectiveness of the proposed approach, seven different datasets were examined using the proposed framework. The datasets originated from different domains: bio-medical, the largest Q&A platform (Stack Exchange), financial and fraud detection.

Finally, the efficiency and efficacy of three set-ups for enhancing prediction performance and exploiting error diversity are explored in Chapter 5. The results are compared to the individual agent classifiers and to each version of CHIMACS. Firstly, the simple version of the proposed approach was compared to the traditional ML classifiers. In the absence of knowledge regarding the examined problem, the performance compared to the performance of the best classifier. Provided with the number of knowledge areas (clusters) the blackboard was capable of recognizing the most competent agents in these areas and exploit them to increase prediction performance. Finally, an adaptive blackboard kernel was proposed to tackle the disadvantages of cluster analysis. The core of that blackboard version is based on a k-NN ML algorithm which feeds the individual components with the k-nearest neighbours. The kernel's knowledge is updated in every epoch. The comparison was performed for seven datasets, 6 from UCI ML Repository and one from Stack Exchange.

CHIMACS is a dynamic and scalable framework used for boosting prediction performance. The proposed approach offers a number of advantages compared to traditional EM and related literature (Pourpanah et al., 2017; Teodorescu & Petridis, 2013). One of its main advantages is the interpretability of the results. An expert can examine the optimal weights and trust values in order to infer which agents are the most competent in a certain knowledge area. Uncoupling is another advantage offered. Agent classifiers can be considered as plugin-in components, where at any point someone can remove or add classifiers. That is very important, since it supports heterogeneity and diversity in the error rates. Finally, the blackboard applies a trust measurement which enables it to evaluate the competence of each agent by assigning a trust reward. That value related an individual classifier with a specific knowledge area and revealed its capabilities. If at any given point the examined data changed behaviour and characteristics, the blackboard was more than capable of realizing it and adjusted itself. In this way, it presented an autonomic behaviour.

6.3 Findings and Contributions to Knowledge

This section demonstrates the main findings of this thesis, which resulted from answering the research questions. The first addressed research question was the following:

1. How can a centralized and intelligent multi-agent system capture, learn and exploit the diversity in capabilities of the accommodated agent estimators in order to increase prediction performance using model-free techniques?

A hybrid multi-agent approach for increasing prediction performance of poorly performing agent models was developed in Chapter 3. The main reason to propose such a framework was the fact that collective reasoning always produces better results in terms of prediction force. Firstly, the use of a suitable architecture for developing the multi-agent system was introduced. Blackboard architecture allows one to exploit the potential heterogeneity in agent's capabilities, while enabling scalability and adaptability at the same time. The proposed approach consists of one trust function that is based on RL techniques, five majority voting functions that enable cooperation and collaborative reasoning and five agent classifiers. The supervisor, namely blackboard manager, orchestrates the entire execution process. Multi-agent architecture is important for constructing a multi-agent environment. The identified requirements for this thesis implementation is transparency, scalability and the fact that the used architecture should not be tightly coupled with any specific domain.

The second research question concerned the most suitable multi-agent architecture to fulfil the above identified design requirements:

2. In there a suitable multi-agent architecture to accommodate heterogeneous agents under the supervision of a central manager that enables flexibility and scalability?

The blackboard supervisor includes three different kernel versions. The first one is not aware of any kind of information regarding the examined problem. The second kernel analyses and holds the identified knowledge areas (clusters), which indicate an immediate relationship between agent performance and the respective clusters.

Finally, the last kernel can learn and adapt during execution process. The blackboard is based on a k-NN algorithm, which enables pattern extraction dynamically without any prior analysis of the data.

As a result of answering the third research question, a reinforcement learning trust measurement and five majority voting functions are proposed in order to efficiently learn, aggregate and exploit the variation in error rates. The trust function allows the blackboard to learn about the potential capabilities of its agent counterparts and exploit them. Also, trust value reflects the performance of each individual classifier and acts as a weight during voting process.

This sets the path for the third research question:

3. How can a function capture, combine and evaluate the quality of the produced knowledge?
    3.1. How can trust measurement based on reinforcement techniques and voting functions enable collaborative reasoning and contribute in reducing variance and bias in trained models?

Blackboard manager engages with the ML agents by assigning binary classification tasks to them. The agents choose from a number of actions which in this case are 0 or 1 followed by probability estimate (level of confidence). Next, voting procedures are employed in order the central blackboard to conclude to a final decision. Then, the validation process takes place. Each agent's answers are validated and in the case of a false answer a trust measurement is calculated and updated by employing the functions in equations (16), (17) and (25). That trust value then acts as a weight in each agent's vote by influencing the given probability estimate in the next round. Finally, collaborative reasoning is enabled through voting functions which fuse the agents trust weighed probability estimate.

The fourth research question addresses the problem of autonomy. The system must be able to learn and adapt with the least human supervision:

4. Can autonomic principles and behaviours be embedded and help improve the effectiveness of the hybrid system?

Utilizing RL techniques gives the ability to the system self-manage. The blackboard is capable of learning in a dynamic manner without stopping the execution process to re-train. The supervisor is constantly aware of the performance of each agent and is able to self-adapt according to the new incoming data samples. At the same time, on a meta-level, the k-NN based kernel is constantly self-adapting, thus enabling the blackboard to identify patterns in data without much human interaction. However, it was shown that identifying patterns using nearest neighbour methods is a computationally expensive task.

The results in Chapter 5 indicate that an appropriate set-up is essential for boosting prediction performance. Firstly, one needs to closely examine the problem and identify the performance metric that needs to be boosted. For example, accuracy is a good metric for datasets with balanced classes. On the other hand, datasets with highly imbalanced classes require other performance metrics boosted e.g. recall or precision. Another parameter that needs attention is the set-up of the proposed approach. Most of the times, clustered and k-NN based set-ups are more accurate compare to the simple version. However, they are computationally expensive and require more steps before the process begins. Hence, the fifth research question is:

5. Can an intelligent blackboard control the reasoning combination by learning about its environment and adapt during the execution process?
   5.1. What kind of information about the examined problem is essential for the central blackboard supervisor?

   It is essential for the blackboard to be aware of certain information in regard to the dataset examined. The clustered version needs to know about the clusters the observations belong to. On the other hand, dynamic CHIMACS needs to know the most similar data instance by evaluating the distance between the examined data observation and the data instances saved in the dataset.

Finally, the last addressed question was the following:

6. How can one optimize the process of searching near optimal values for the reinforcement signal and gamma parameters?

Reinforcement learning trust function includes two parameters that require manual configuration until the system converges to a near optimal state. Two optimization

methods were applied for optimizing reinforcement signal reward r and discount gamma factor, simulated annealing and a Bayesian method (Thomson Sampling). Thomson sampling is a reinforcement learning approach that is used in multi-armed bandit problems. The purpose of this optimization approach is to maximize the reward depending on the performance of the agent (bandit). At the same time, gamma factor is a massively large search space, making the search for the optimal value tedious and time consuming. Thus, an approach adopted from metallurgy was applied, namely Simulated Annealing. This method takes random steps into a confined search space, which makes convergence to near optimal values very fast and efficient.

6.4 Limitations

Although this research work presented very promising results on how one can increase accuracy, precision, etc., using a number of agents with diverse error rates, there are a certain limitation. Below, a number of limitations of this research work are discussed:

- *Structure*. The architecture utilizes reinforcement learning/trust techniques to dynamically evaluate the agent's performance. During training and testing time the level of the trust value fluctuates and that reflects the performance of each agent. However, the blackboard does not perform any evaluation of its internal state. Hence it cannot change its configuration, deactivate, send for training or completely remove one agent, manually and without disrupting the process of the system.
- *Parallelisation*. Performance is a major issue of this research work. Although the convergence performance of simple and clustered CHIMACS is good enough considering the optimization methods applied, k-NN based kernel took a considerable amount of time to converge. The time taken is related to the structure of the k-nearest neighbour algorithm. The kernel is updated and adapted in every epoch, which makes the entire process very slow. Paralleling the tasks is one of the solutions to improve the performance of the proposed approach. For example, splitting data and distributing computational tasks over multiple clusters, CPUs, GPUs, and threads can

potentially significantly improve the performance and is something that needs to be explored in the future.

- *Optimization*. Further work in the future requires the parameters of the currently implemented trust to function. The overall loss is affected from the parameters of the trust function too. It is essential to evaluate the performance of the proposed approach using other optimization processes. That is important because we need to be sure that the system will not converge to a local minima. The implemented reinforcement trust function includes three parameters that require optimisation. Firstly, the learning rate μ that controls the speed which the blackboard will learn, is one of the parameters. Optimisation methods have been applied for the next two, reward signal r_a^i and gamma factor γ. Thomson Sampling, a Bayesian method for solving Multi-Armed Bandit problems, has been applied. Also, an optimization method named Simulated Annealing was applied to find the optimal gamma factor weights. However, more optimization methods e.g. Genetic Algorithms, Particle Swarm Optimization methods (PSO) etc. need to be explored.

6.5 Future Directions

The research work presented in this thesis demonstrated an approach for enhancing prediction performance by exploiting the diversity in the error rates. Research work and related results showed how one can increase accuracy, precision, etc., using a number of datasets from different domains, including bio-medical, Q&A Stack Exchange, Spam and Financial datasets. Although the results are very promising, several directions for further research do exist. Below, a number of future steps for further research work are proposed:

- *Architecture*. The architecture of the proposed approach can be explored further. For example, there is a variety of blackboard versions e.g. one can apply a reflective blackboard that provides more flexibility and efficiency. Moreover, one can compare the performance of the proposed approach with other state-of-the-art multi-agent implementations. At the same time, the

proposed architecture could be tested under a hierarchical structure. In future work, it could be observed how the proposed approach behaves, when an agent is another CHIMACS framework.

- *Trust*. In future work, it is necessary to test the proposed trust function against other reinforcement learning functions in order to look for more accurate results. For this purpose, one has to do a thorough review of the current literature and document the state-of-the-art RL functions. Then, the structure of the proposed framework is flexible enough to alternate the reward functions.
- *Agents.* Data diversity with regard to training the classifiers can be explored further. For example, one can feed a set of principal components to some of the agent classifiers and some others with the original dataset structure and achieve diversity in data and hence, in the error rate.
- *Dynamic Configuration*. We could use CBR to re-use configurations that has been successful in the past. CHIMACS will automatically change the configuration by adjusting the trust over a period of operation.

6.6 References

Abbeel, Pieter, & Ng, Andrew Y. (2004). *Apprenticeship learning via inverse reinforcement learning*. Paper presented at the Proceedings of the twenty-first international conference on Machine learning, Banff, Alberta, Canada.

Alpaydin, Ethem. (2010). *Introduction to Machine Learning*: The MIT Press.

Bart Jacob, Richard Lanyon-Hogg, Devaprasad K Nadgir,Amr F Yassin (2004). *A Practical Guide to the e to the IBM Autonomic mic Computing Toolkit* ibm.com/redbooks: IBM.

Barthes, E. Scalabrin and J. P. (1993). Osaca : une architecture ouverte d'agents cognitifs independants. *In Actes de la Journee "Systemes multi-agents",Montpellier, France*.

Barthes, Weiming Shen and Jean-Paul. (1995). *DIDE: A multi-agent environment for engineering design.* Paper presented at the Proceedings of the First International Conference on Multi– Agent Systems,, San Francisco, CA.

Bauer, Eric, & Kohavi, Ron. (1999). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Machine Learning, 36*(1), 105-139. doi:10.1023/a:1007515423169

Beran, Rudolf J. (1992). Introduction to Efron (1979) Bootstrap Methods: Another Look at the Jackknife. In Samuel Kotz & Norman L. Johnson (Eds.), *Breakthroughs in Statistics: Methodology and Distribution* (pp. 565-568). New York, NY: Springer New York.

Bonura, S., Morreale, V., Francaviglia, G., Marguglio, A., Cammarata, G., & Puccio, M. (2009). Intentions in BDI Agents: From Theory to Implementation. In Yves Demazeau, Juan Pavón, JuanM Corchado, & Javier Bajo (Eds.), *7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009)* (Vol. 55, pp. 227-236): Springer Berlin Heidelberg.

Book review: Blackboard Architectures and Applications Edited by V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum (Academic Press). (1990). *SIGART Bull., 1*(3), 19-20. doi:10.1145/101340.1056294

Brandt, Felix, Conitzer, Vincent, Endriss, Ulle, J\, \#233, r\, . . . Procaccia, Ariel D. (2016). *Handbook of Computational Social Choice*: Cambridge University Press.

Braubach, Lars, Pokahr, Alexander, & Lamersdorf, Winfried. (2005). Jadex: A BDI-Agent System Combining Middleware and Reasoning. In Rainer Unland, Monique Calisti, & Matthias Klusch (Eds.), *Software Agent-Based Applications, Platforms and Development Kits* (pp. 143-168): Birkhäuser Basel.

Brázdil, Tomáš, Chatterjee, Krishnendu, Chmelík, Martin, Forejt, Vojtěch, Křetínský, Jan, Kwiatkowska, Marta, . . . Ujma, Mateusz. (2014). Verification of Markov Decision Processes Using Learning Algorithms. In

Franck Cassez & Jean-François Raskin (Eds.), *Automated Technology for Verification and Analysis: 12th International Symposium, ATVA 2014, Sydney, NSW, Australia, November 3-7, 2014, Proceedings* (pp. 98-114). Cham: Springer International Publishing.

Breiman, Leo. (1996). Bagging predictors. *Mach. Learn., 24*(2), 123-140. doi:10.1023/a:1018054314350

Brunet, Charles-Antoine, de Lafontaine, Jean, & Lachiver, Gérard. (2003). *Generic Agent Architecture for Embedded Intelligent Systems*, Berlin, Heidelberg.

Burlutskiy, Nikolay, Petridis, Miltos, Fish, Andrew, Chernov, Alexey, & Ali, Nour. (2016). An Investigation on Online Versus Batch Learning in Predicting User Behaviour. In Max Bramer & Miltos Petridis (Eds.), *Research and Development in Intelligent Systems XXXIII: Incorporating Applications and Innovations in Intelligent Systems XXIV* (pp. 135-149). Cham: Springer International Publishing.

Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications, 45*(1), 41-51. doi:10.1007/bf00940812

Chen, Hsinchun, Chiang, Roger H. L., & Storey, Veda C. (2012). Business intelligence and analytics: from big data to big impact. *MIS Q., 36*(4), 1165-1188.

Chen, Jin, Luo, De-lin, & Mu, Fen-xiang. (2009, 25-28 July 2009). *An improved ID3 decision tree algorithm.* Paper presented at the 2009 4th International Conference on Computer Science & Education.

Chen, Tianqi, & Guestrin, Carlos. (2016). *XGBoost: A Scalable Tree Boosting System*. Paper presented at the Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA.

Chevaleyre, Yann, Endriss, Ulle, Lang, Jérôme, & Maudet, Nicolas. (2007). A Short Introduction to Computational Social Choice. In Jan van Leeuwen, Giuseppe F. Italiano, Wiebe van der Hoek, Christoph Meinel, Harald Sack, & František Plášil (Eds.), *SOFSEM 2007: Theory and Practice of Computer Science: 33rd Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 20-26, 2007. Proceedings* (pp. 51-69). Berlin, Heidelberg: Springer Berlin Heidelberg.

. Cluster Analysis. (2007) *Interactive and Dynamic Graphics for Data Analysis: With R and Ggobi* (pp. 103-128). New York, NY: Springer New York.

Cotter, Andrew. (2003). *Stochastic Optimization for Machine Learning.* Chicago University, Chicago, Illinois.

Dasarathy, B. V., & Sheela, B. V. (1979). A composite classifier system design: Concepts and methodology. *Proceedings of the IEEE, 67*(5), 708-713. doi:10.1109/PROC.1979.11321

Decisions in Human Centric Multiagent Systems: Dealing with Softness and Bipolarity in Judgments, Intentions and Evaluations. (2016). *Procedia Computer Science, 102*, 7-8. doi:http://dx.doi.org/10.1016/j.procs.2016.09.361

Dietterich, Thomas G. (2000). Ensemble Methods in Machine Learning *Multiple Classifier Systems: First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings* (pp. 1-15). Berlin, Heidelberg: Springer Berlin Heidelberg.

Drucker, Harris, Cortes, Corinna, Jackel, L. D., LeCun, Yann, & Vapnik, Vladimir. (1994). Boosting and other ensemble methods. *Neural Comput., 6*(6), 1289-1301. doi:10.1162/neco.1994.6.6.1289

Endriss, Ulle. (2011). Computational Social Choice: Prospects and Challenges. *Procedia Computer Science, 7*, 68-72. doi:http://dx.doi.org/10.1016/j.procs.2011.12.022

Endriss, Ulle. (2013). Computational Social Choice (with a Special Emphasis on the Use of Logic). In Guram Bezhanishvili, Sebastian Löbner, Vincenzo Marra, & Frank Richter (Eds.), *Logic, Language, and Computation: 9th International Tbilisi Symposium on Logic, Language, and Computation, TbiLLC 2011, Kutaisi, Georgia, September 26-30, 2011, Revised Selected Papers* (pp. 1-3). Berlin, Heidelberg: Springer Berlin Heidelberg.

Endriss, Ulle. (2014). Social Choice Theory as a Foundation for Multiagent Systems. In Jörg P. Müller, Michael Weyrich, & Ana L. C. Bazzan (Eds.), *Multiagent System Technologies: 12th German Conference, MATES 2014, Stuttgart, Germany, September 23-25, 2014. Proceedings* (pp. 1-6). Cham: Springer International Publishing.

Erhan, Dumitru, Bengio, Yoshua, Courville, Aaron, Manzagol, Pierre-Antoine, Vincent, Pascal, & Bengio, Samy. (2010). Why Does Unsupervised Pre-training Help Deep Learning? *J. Mach. Learn. Res., 11*, 625-660.

Eric Savitz, Gartner. (2012). Top 10 Critical Tech Trends for the Next Five Years. Retrieved from Forbes.com website: http://www.forbes.com/sites/ericsavitz/2012/10/22/gartner-10-critical-tech-trends-for-the-next-five-years/

Eric Savitz, Gartner. (2013). Top 10 Strategic Technology Trends for 2013. Retrieved from Forbes.com website: http://www.forbes.com/sites/ericsavitz/2012/10/23/gartner-top-10-strategic-technology-trends-for-2013/

Erman, Lee D., Hayes-Roth, Frederick, Lesser, Victor R., & Reddy, D. Raj. (1980). The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. *ACM Comput. Surv., 12*(2), 213-253. doi:10.1145/356810.356816

Erman, Lee D., London, Philip E., & Fickas, Stephen F. (1981). *The design and an example use of Hearsay-III*. Paper presented at the Proceedings of the 7th international joint conference on Artificial intelligence - Volume 1, Vancouver, BC, Canada.

Fard, Amir Anvarian, & Rafe, Vahid. (2014). *A Formal Architectural Style for Designing Multi-agent Systems*.

Frank, Eibe, Hall, Mark, & Pfahringer, Bernhard. (2003). *Locally weighted naive bayes*. Paper presented at the Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence, Acapulco, Mexico.

Freund, Yoav, & Schapire, Robert E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences, 55*(1), 119-139. doi:http://dx.doi.org/10.1006/jcss.1997.1504

Fumera, Giorgio, Pillai, Ignazio, & Roli, Fabio. (2004). A Two-Stage Classifier with Reject Option for Text Categorisation. In Ana Fred, Terry M. Caelli, Robert P. W. Duin, Aurélio C. Campilho, & Dick de Ridder (Eds.), *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Portugal, August 18-20, 2004. Proceedings* (pp. 771-779). Berlin, Heidelberg: Springer Berlin Heidelberg.

Fumera, Giorgio, & Roli, Fabio. (2005). A Theoretical and Experimental Analysis of Linear Combiners for Multiple Classifier Systems. *IEEE Trans. Pattern Anal. Mach. Intell., 27*(6), 942-956. doi:10.1109/tpami.2005.109

Funtowicz, S. O., & Ravetz, Jerome R. (1995). Science for the Post Normal Age. In Laura Westra & John Lemons (Eds.), *Perspectives on Ecological Integrity* (pp. 146-161). Dordrecht: Springer Netherlands.

Geman, Stuart, & Geman, Donald. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell., 6*(6), 721-741. doi:10.1109/tpami.1984.4767596

Giacinto, G., Roli, F., & Fumera, G. (2000, 2000). *Design of effective multiple classifier systems by clustering of classifiers.* Paper presented at the Proceedings 15th International Conference on Pattern Recognition. ICPR-2000.

Gómez, David, & Rojas, Alfonso. (2015). An Empirical Overview of the No Free Lunch Theorem and Its Effect on Real-World Machine Learning Classification. *Neural Computation, 28*(1), 216-228. doi:10.1162/NECO_a_00793

Gopalakrishna, Aravind Kota, Ozcelebi, Tanir, Liotta, Antonio, & Lukkien, Johan J. (2013). Relevance as a Metric for Evaluating Machine Learning Algorithms. In Petra Perner (Ed.), *Machine Learning and Data Mining in Pattern Recognition: 9th International Conference, MLDM 2013, New York, NY, USA, July 19-25, 2013. Proceedings* (pp. 195-208). Berlin, Heidelberg: Springer Berlin Heidelberg.

Granatyr, Jones, Botelho, Vanderson, Lessing, Otto Robert, Em, Edson, #237, Scalabrin, lio, . . . Enembreck, cio. (2015). Trust and Reputation Models for Multiagent Systems. *ACM Comput. Surv., 48*(2), 1-42. doi:10.1145/2816826

Hajek, Bruce. (1988). Cooling schedules for optimal annealing. *Math. Oper. Res., 13*(2), 311-329. doi:10.1287/moor.13.2.311

Hall, Mark, Frank, Eibe, Holmes, Geoffrey, Pfahringer, Bernhard, Reutemann, Peter, & Witten, Ian H. (2009). The WEKA data mining software: an update. *SIGKDD Explor. Newsl., 11*(1), 10-18. doi:10.1145/1656274.1656278

Heath, B., Hill, R., Ciarallo, F. (2009). A Survey of Agent-Based Modeling Practices (January 1998 to July 2008). *Journal of Artificial Societies and Social Simulation*, pp 9.

Hofmann, Thomas. (2001). Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning, 42*(1), 177-196. doi:10.1023/a:1007617005950

Huynh, Trung Dong, Jennings, Nicholas R., & Shadbolt, Nigel R. (2006). An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems, 13*(2), 119-154. doi:10.1007/s10458-005-6825-4

Jain, Anil K., Duin, Robert P. W., & Mao, Jianchang. (2000). Statistical Pattern Recognition: A Review. *IEEE Trans. Pattern Anal. Mach. Intell., 22*(1), 4-37. doi:10.1109/34.824819

Jennings, Nicholas R., Sycara, Katia, & Wooldridge, Michael. (1998). A Roadmap of Agent Research and Development. *Autonomous Agents and Multi-Agent Systems, 1*(1), 7-38. doi:10.1023/a:1010090405266

Kapetanakis, Stelios, Petridis, Miltos, Knight, Brian, Ma, Jixin, & Bacon, Liz. (2010). A Case Based Reasoning Approach for the Monitoring of Business Workflows. In Isabelle Bichindaritz & Stefania Montani (Eds.), *Case-Based Reasoning. Research and Development* (Vol. 6176, pp. 390-405): Springer Berlin Heidelberg.

Keogh, Eamonn, & Mueen, Abdullah. (2017). Curse of Dimensionality. In Claude Sammut & Geoffrey I. Webb (Eds.), *Encyclopedia of Machine Learning and Data Mining* (pp. 314-315). Boston, MA: Springer US.

Khalid, S., Khalil, T., & Nasreen, S. (2014, 27-29 Aug. 2014). *A survey of feature selection and feature extraction techniques in machine learning.* Paper presented at the 2014 Science and Information Conference.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science, 220*(4598), 671-680. doi:10.1126/science.220.4598.671

Kittler, Josef, Hatef, Mohamad, Duin, Robert P. W., & Matas, Jiri. (1998). On Combining Classifiers. *IEEE Trans. Pattern Anal. Mach. Intell., 20*(3), 226-239. doi:10.1109/34.667881

Kokorakis, Vasileios Manousakis, Petridis, Miltos, & Kapetanakis, Stelios. (2017). A Blackboard Based Hybrid Multi-Agent System for Improving Classification Accuracy Using Reinforcement Learning Techniques. In Max Bramer & Miltos Petridis (Eds.), *Artificial Intelligence XXXIV: 37th SGAI International Conference on Artificial Intelligence, AI 2017, Cambridge, UK, December 12-14, 2017, Proceedings* (pp. 47-57). Cham: Springer International Publishing.

Komiyama, Junpei, Honda, Junya, & Nakagawa, Hiroshi. (2015). *Optimal regret analysis of Thompson sampling in stochastic multi-armed bandit problem with multiple plays*. Paper presented at the Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, Lille, France.

Kuncheva, Ludmila I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*: Wiley-Interscience.

Kuncheva, Ludmila I., & Whitaker, Christopher J. (2003). Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. *Machine Learning, 51*(2), 181-207. doi:10.1023/a:1022859003006

Lam, Louisa. (2000). *Classifier Combinations: Implementations and Theoretical Issues*. Paper presented at the Proceedings of the First International Workshop on Multiple Classifier Systems.

Lars Braubach, Alexander Pokahr, Kai Jander. (2005, 2013/07/19 15:21). Jadex. Retrieved from https://www.activecomponents.org/bin/view/About/Features

Lars Braubach, Alexander Pokahr, Kai Jander. (2012). Jadex Multi-Agent System. Retrieved from https://www.activecomponents.org/bin/view/About/Features

Leslie Pack Kaelbling, Michael L. Littman, Andrew W. Moore. (1996). Reinforcement Learning: A Survey. *Journal of Articial Intelligence Research, 4*, 237-285.

Lichman, M. (2013). {UCI} Machine Learning Repository. Retrieved from http://archive.ics.uci.edu/ml/

Linkens, D. A., Abbod, M. F., & Browne, A. (1999, 1999). *Blackboard architecture for intelligent control system.* Paper presented at the Emerging Technologies and Factory Automation, 1999. Proceedings. ETFA '99. 1999 7th IEEE International Conference on.

Luo, Tao, Chen, Guoliang, & Zhang, Yunquan. (2013). H-DB: Yet Another Big Data Hybrid System of Hadoop and DBMS. In Joanna Kołodziej, Beniamino Di Martino, Domenico Talia, & Kaiqi Xiong (Eds.), *Algorithms and Architectures for Parallel Processing: 13th International Conference, ICA3PP 2013, Vietri sul Mare, Italy, December 18-20, 2013, Proceedings, Part I* (pp. 324-335). Cham: Springer International Publishing.

Metropolis, Nicholas, Rosenbluth, Arianna W., Rosenbluth, Marshall N., Teller, Augusta H., & Teller, Edward. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics, 21*(6), 1087-1092. doi:10.1063/1.1699114

Milgrom∗, Jonathan Levin and Paul. (2004). *Introduction to Choice Theory*. Retrieved from https://www.semanticscholar.org

Mooi, Erik, & Sarstedt, Marko. (2011). Cluster Analysis *A Concise Guide to Market Research: The Process, Data, and Methods Using IBM SPSS Statistics* (pp. 237-284). Berlin, Heidelberg: Springer Berlin Heidelberg.

Morgan, R. Englemore and T. (1988). *Blackboard Systems*: Pub. Addison-Wesley Pub. Co.

Mui, L., Mohtashemi, M., & Halberstadt, A. (2002). *A Computational Model of Trust and Reputation for E-businesses*. Paper presented at the Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 7 - Volume 7.

Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*: The MIT Press.

Murty, M. Narasimha, & Devi, V. Susheela. (2011). Nearest Neighbour Based Classifiers *Pattern Recognition: An Algorithmic Approach* (pp. 48-85). London: Springer London.

Nikolaev, Alexander G., & Jacobson, Sheldon H. (2010). Simulated Annealing. In Michel Gendreau & Jean-Yves Potvin (Eds.), *Handbook of Metaheuristics* (pp. 1-39). Boston, MA: Springer US.

Nodarakis, Nikolaos, Rapti, Angeliki, Sioutas, Spyros, Tsakalidis, Athanasios K., Tsolis, Dimitrios, Tzimas, Giannis, & Panagis, Yannis. (2017). *(A)kNN Query Processing on the Cloud: A Survey*, Cham.

Nolle, L., Wong, K. C. P., & Hopgood, A. A. (2002). DARBS: A Distributed Blackboard System. In Max Bramer, Frans Coenen, & Alun Preece (Eds.), *Research and Development in Intelligent Systems XVIII* (pp. 161-170): Springer London.

O'Leary, Daniel E. (1994). *Models of consensus for multiple agent systems*. Paper presented at the Proceedings of the Tenth international conference on Uncertainty in artificial intelligence, Seattle, WA.

Petridis, M., & Knight, B. (1996). The integration of an intelligent knowledge-based system into engineering software using the blackboard structure. *Advances in Engineering Software, 25*(2), 141-147. doi:http://dx.doi.org/10.1016/0965-9978(95)00101-8

Petridis, M., Knight, B., & Edwards, D. (1991). A Design for Reliable CFD Software. In C. A. Brebbia & A. J. Ferrante (Eds.), *Reliability and Robustness of Engineering Software II: Proceedings of the Second International Conference held in Milan, Italy, during 22–24 April 1991* (pp. 3-17). Dordrecht: Springer Netherlands.

Polikar, Robi. (2009). Ensemble learning. http://www.scholarpedia.org/article/Ensemble_learning

Pourpanah, Farhad, Tan, Choo Jun, Lim, Chee Peng, & Mohamad-Saleh, Junita. (2017). A Q-learning-based multi-agent system for data classification. *Applied Soft Computing, 52*, 519-531. doi:https://doi.org/10.1016/j.asoc.2016.10.016

Procaccia, Ariel D. (2008). *Computational Voting Theory: Of the Agents, By the Agents, For the Agents.* (Doctor of Philosophy), Hebrew University.

Rahman, A. F. R., & Fairhurst, M. C. (1999). Serial Combination of Multiple Experts: A Unified Evaluation. *Pattern Analysis & Applications, 2*(4), 292-311. doi:10.1007/s100440050038

Ramchurn, Sarvapali D., Huynh, Dong, & Jennings, Nicholas R. (2004). Trust in multi-agent systems. *Knowl. Eng. Rev., 19*(1), 1-25. doi:10.1017/s0269888904000116

Rogova, Galina. (1994). Combining the results of several neural network classifiers. *Neural Networks, 7*(5), 777-781. doi:http://dx.doi.org/10.1016/0893-6080(94)90099-X

Sànchez-Marrè, M., Gibert, K., Sojda, R. S., Steyer, J. P., Struss, P., Rodríguez-Roda, I., . . . Roehl, E. A. (2008). Chapter Eight Intelligent Environmental Decision Support Systems. In A. A. Voinov A. E. Rizzoli A.J. Jakeman & S. H. Chen (Eds.), *Developments in Integrated Environmental Assessment* (Vol. Volume 3, pp. 119-144): Elsevier.

Schumacher, Michael. (2001). *Objective coordination in multi-agent system engineering: design and implementation*: Springer-Verlag.

Schweitzer, Frank. (2007). Collective Decisions in Multi-Agent Systems. In Shingo Takahashi, David Sallach, & Juliette Rouchier (Eds.), *Advancing Social Simulation: The First World Congress* (pp. 7-12). Tokyo: Springer Japan.

Shehory, Onn. (2001). *Software architecture attributes of multi-agent systems*. Paper presented at the First international workshop, AOSE 2000 on Agent-oriented software engineering, Limerick, Ireland.

Sorici, A., Picard, G., Boissier, O., Santi, A., Hbner, J. (2012). *Multi-Agent Oriented Reorganisation within the JaCaMo infrastructure.* Paper presented at the The Third International Workshop on Iinfraestructures and tools for multiagent systems: ITMAS. .

St, Ross, phane, Pineau, Joelle, Chaib-draa, Brahim, & Kreitmann, Pierre. (2011). A Bayesian Approach for Learning and Planning in Partially Observable Markov Decision Processes. *J. Mach. Learn. Res., 12*, 1729-1770.

Sun, S., & Huang, R. (2010, 10-12 Aug. 2010). *An adaptive k-nearest neighbor algorithm.* Paper presented at the 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery.

Tabrizi, Sahar S., & Cavus, Nadire. (2016). A Hybrid KNN-SVM Model for Iranian License Plate Recognition. *Procedia Computer Science, 102*, 588-594. doi:http://dx.doi.org/10.1016/j.procs.2016.09.447

Taylor, A.D. (2005). Social Choice and the Mathematics of Manipulation [Press release]

Teacy, W. T. Luke, Patel, Jigar, Jennings, Nicholas R., & Luck, Michael. (2006). TRAVOS: Trust and Reputation in the Context of Inaccurate Information Sources. *Autonomous Agents and Multi-Agent Systems, 12*(2), 183-198. doi:10.1007/s10458-006-5952-x

Teodorescu, Elena Irena, & Petridis, Miltos. (2013). An Agent Based Framework for Multiple, Heterogeneous Case Based Reasoning. In Sarah Jane Delany & Santiago Ontañón (Eds.), *Case-Based Reasoning Research and Development: 21st International Conference, ICCBR 2013, Saratoga Springs, NY, USA, July 8-11, 2013. Proceedings* (pp. 314-328). Berlin, Heidelberg: Springer Berlin Heidelberg.

TURING, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind, LIX*(236), 433-460. doi:10.1093/mind/LIX.236.433

Twardowski, B., & Ryzko, D. (2014, 11-14 Aug. 2014). *Multi-agent Architecture for Real-Time Big Data Processing.* Paper presented at the Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on.

Tweedale, Jeffrey, & Cutler, Philip. (2006). Trust in Multi-Agent Systems. In Bogdan Gabrys, Robert J. Howlett, & Lakhmi C. Jain (Eds.), *Knowledge-Based Intelligent Information and Engineering Systems: 10th International Conference, KES 2006, Bournemouth, UK, October 9-11, 2006. Proceedings, Part II* (pp. 479-485). Berlin, Heidelberg: Springer Berlin Heidelberg.

Van Dyke Parunak, H., Nielsen, Paul, Brueckner, Sven, & Alonso, Rafael. (2007). Hybrid Multi-agent Systems: Integrating Swarming and BDI Agents. In Sven A. Brueckner, Salima Hassas, Márk Jelasity, & Daniel Yamins (Eds.), *Engineering Self-Organising Systems: 4th International Workshop, ESOA*

2006, Hakodate, Japan, May 9, 2006, Revised and Invited Papers (pp. 1-14). Berlin, Heidelberg: Springer Berlin Heidelberg.

Wang, W. (2010, 18-23 July 2010). *Heterogeneous Bayesian ensembles for classifying spam emails.* Paper presented at the The 2010 International Joint Conference on Neural Networks (IJCNN).

Weerdt, Mathijs de, & Clement, Brad. (2009). Introduction to planning in multiagent systems. *Multiagent Grid Syst., 5*(4), 345-355.

Wibral, Michael, Lizier, Joseph T., & Priesemann, Viola. (2015). Bits from Brains for Biologically Inspired Computing. *Frontiers in Robotics and AI, 2*(5). doi:10.3389/frobt.2015.00005

Wilson, Jeffrey R., & Lorenz, Kent A. (2015). Standard Binary Logistic Regression Model *Modeling Binary Correlated Responses using SAS, SPSS and R* (pp. 25-54). Cham: Springer International Publishing.

Witkowski, M., & Pitt, J. (2000, 2000). *Objective trust-based agents: Trust and trustworthiness in a multi-agent trading society.* Paper presented at the Proceedings Fourth International Conference on MultiAgent Systems.

Wittig, T. (1992). *ARCHON: an architecture for multi-agent systems*: Ellis Horwood.

Wooldridge, Michael, & Jennings, Nicholas R. (1995). *Agent theories, architectures, and languages: a survey*. Paper presented at the Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents, Amsterdam, The Netherlands.

Woźniak, Michał, Graña, Manuel, & Corchado, Emilio. (2014). A survey of multiple classifier systems as hybrid systems. *Information Fusion, 16*, 3-17. doi:http://dx.doi.org/10.1016/j.inffus.2013.04.006

Xia, Lirong. (2011). *Computational Voting Theory: Game-Theoretic and Combinatorial Aspects.* (Doctor of Philosophy), Duke University.

Zhou, Zhi-Hua, & Tang, Wei. (2003). Selective Ensemble of Decision Trees. In Guoyin Wang, Qing Liu, Yiyu Yao, & Andrzej Skowron (Eds.), *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing: 9th International Conference, RSFDGrC 2003, Chongqing, China, May 26–29, 2003 Proceedings* (pp. 476-483). Berlin, Heidelberg: Springer Berlin Heidelberg.

Zhou, Zhi-Hua, Wu, Jianxin, & Tang, Wei. (2002). Ensembling neural networks: Many could be better than all. *Artificial Intelligence, 137*(1), 239-263. doi:https://doi.org/10.1016/S0004-3702(02)00190-X