Fuzzy Rule-Based Systems: Design, Analysis, and Applications

By

Jeremy Kerr-Wilson

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering

University of Alberta

Abstract

The extraction of knowledge from data is a relatively recent computational pursuit which has been the focus of significant research attention and has an extensive field of potential applications. With the advent of widespread data collection describing a variety of systems spanning many fields of expertise, the extraction of useful knowledge from data, for prediction or insight, has extraordinary potential and significant value. This realization has led to the development of machine learning, a collection of computational processes and algorithms through which sophisticated computer models can be constructed on the basis of data. Human centric systems, and more specifically certain forms of fuzzy systems, are a notable subset of machine learning which focus on knowledge extraction using multi-valued logic and set theory. Fuzzy systems are particularly well-suited to human-centric modelling as fuzzy sets describe real world systems, as perceived by humans, much more accurately than binary models. Fuzzy rule-based models are a form of fuzzy model which is particularly well-suited to human centric tasks due to the high degree of readability and interpretability conveyed to an expert reader. This, combined with their strong predictive ability, makes fuzzy rule-based systems an excellent candidate for those computational modelling pursuits where predictive accuracy may not be the singular requirement of a model.

The objective of this dissertation is to design, analyze, and develop novel applications, methodologies, and algorithms for use with fuzzy rule-based systems, seeking to further their utility in predictive and human-centric modelling. In this dissertation, fuzzy rule-based systems are applied to different problem types, combined with existing computational data-structures and architectures, extracted from data in novel manners and formats, and analyzed to assess certain aspects of rule quality. Acknowledging the critical role of human centricity in computational modelling, we develop a set of fuzzy rule stability criteria which aim to quantify aspects of fuzzy rule quality while capturing critical non-numerical aspects of rule quality such as repeatability, consistency, and generalizability. We examine the generation of fuzzy rule-based models using hierarchical clustering and extract granular fuzzy models from data, forming information granules in the consequent parts of the fuzzy rules. We make use of fuzzy rule-based systems as the component models (weak learners) of a boosted ensemble, exploring their predictive power and adaptability in this environment. Finally, a novel fuzzy rule architecture is proposed using a hierarchical structure, alongside a generation procedure for extracting this hierarchical structure form data, with the aim of improving predictive performance and increasing the interpretability of the system.

Each of these topics are justified with extensive experimental studies, using real-world data sets available from public repositories, demonstrating the feasibility or superiority of the proposed methods as compared to existing methodologies.

Preface

The research presented in this thesis was performed under the supervision of Dr Witold Pedrycz and was supported by funding from the National Sciences and Engineering Research Council of Canada (NSERC), specifically the Discovery Grant Program, and the Canada Research Chair Program.

Chapter 4 of this this dissertation has been published as Kerr-Wilson, J., & Pedrycz, W. (2016). Some new qualitative insights into quality of fuzzy rule-based models. Fuzzy Sets and Systems. https://doi.org/10.1016/j.fss.2016.05.002. Dr Pedrycz was instrumental in guiding me through the creation of this study, contributed significantly to idea development, and helped with my learning process regarding manuscript composition. A significant amount of the idea development, all program implementation and data collection, and result analysis was performed by me under his supervision.

Chapter 5 of this thesis has been published as Kerr-Wilson, J., & Pedrycz, W. (2016). Design of rule-based models through information granulation. Expert Systems with Applications, 46, 274–285. https://doi.org/10.1016/j.eswa.2015.10.030. I was responsible for the manuscript composition, idea development, and experimental process including programming and experimental execution and analysis. Dr Pedrycz was involved in the concept formation and was helpful in guiding me through the existing literature on the topic as well as supervising manuscript composition and experimental structure.

Chapter 6 of this thesis has been submitted to Fuzzy Sets and Systems as Kerr-Wilson, J. & Pedrycz, W., "Boosting with Fuzzy Rule Based Systems". I was responsible for the idea development, the implementation of experimental code, data collection, result analysis, and manuscript composition of this study. Dr Pedrycz worked on this study in a supervisory role, aiding in the initial formation of the research topic, and having meaningful contributions to the manuscript composition and refinement process.

Chapter 7 of this thesis has been submitted to Fuzzy Sets and Systems as Kerr-Wilson, J. & Pedrycz, W., "Generating a Hierarchical Rule-based Model". Dr Pedrycz was the supervisory author of this study and aided in initial idea formation and provided helpful advice on the experimental direction of the study. I was responsible for the composition of the manuscript, the experimental setup and data collection, and the analysis of the results.

Acknowledgments

I would like to thank my supervisor, Dr. Witold Pedrycz, from whom I have learned so much throughout my graduate experience. I profoundly appreciate your support, your patience, and your contributions both to my work and to our field. You have made my doctoral process an exciting experience through which I have grown significantly as a researcher, teacher, and person. Your never-ending passion and insight for research and academic excellence has driven my success, and it has been a privilege to be your student.

I would also like to thank my supervisory committee, Dr. Marek Reformat, Dr. Petr Musilek, Dr. Venkata Dinavahi and Dr. Vladik Kreinovich for their comments and suggestion. I greatly appreciate your contributions and time.

Additionally, I would like to thank my parents, Greg and Vicki Kerr-Wilson, for their ongoing and continuous support of my academic pursuits, and for encouraging me to pursue my passions in software, engineering, and research.

I would like to thank my spouse, Victoria Merritt, for her support, encouragement, understanding, and patience as I have worked towards the completion of my PhD.

Further, a heartfelt thank you to all of the Electrical and Computer Engineering departmental staff, who have always been friendly and helpful. Thank you specifically to Pinder Bains and Wendy Barton who both provided exceptional student support during my time here.

Finally, I would like to thank all my friends and family for their ongoing support and encouragement throughout my graduate experience. You have all made my time as a graduate student wonderful, both in and out of school.

Jeremy Kerr-Wilson

University of Alberta

January 2019

Table of Contents

List of Tables

List of Figures

Symbols

$\mathbf{x}_\mathbf{k}$ **–** The k^{th} input instance

y_k – Output value or prediction for the k^{th} instance

m – Fuzzification coefficient

n – Number of variables (features)

$\boldsymbol{D}$ – A dataset

N – number of instances

$U = [u_{ik}]$ – Partition matrix

c – Number of rules/clusters

p – Number of partitions or models

t – Designation for an iteration or hierarchical level

$|| \mathbf{.} ||$ – Distance measure

$| \mathbf{.} |$ – Length operator or absolute value

A_i –Membership function for cluster i

$\boldsymbol{v_i}$ – Cluster prototype or particle

b_i – The consequent of a fuzzy rule

ρ – Parameter for granule formation

$W = [w_k]$ – instance weights, weight of the *kth* instance

K – Number of classes

ε – Error or error threshold

$f(x)$ – A given function

$h(x)$ – A weak learner

Q – The performance index for FCM

E – Error, classification or regression

L – Number of levels in a hierarchy

$Class_k$ – the class label designation for the *kth* instance

1 INTRODUCTION

Computational modelling is a powerful framework which can be used to help humans interact with and understand complex data and systems in a meaningful and productive way. Using complex algorithms, researchers can extract knowledge from data, forming highly accurate models describing complex systems with relative ease. As computational modelling and machine learning has developed as an area of research interest, the accuracy, or ability to correctly predict system behavior, of the models has been the overwhelming focus of much of the research. While model correctness, and consequent accuracy, is obviously important, another factor of modeling – human readability and interpretability – has received significantly less attention but is also valuable. The goal of interpretable models is to present the extracted knowledge in a format which is meaningful and digestible to a human reader, a factor which is critical in many areas of expertise.

Fuzzy sets and systems are a useful tool in constructing human-centric models. Fuzzy modelling provides a critical component for human readable systems, as it pulls computational operations out of a binary state and into a more analog space. The consideration of models described by fuzzy constructs has two advantages. First, fuzzy modelling provides a strong basis for machine learning as fuzzy models can describe complex systems in compact formats. Secondly, the analog nature of fuzzy systems results in more intuitively interpretable models as the real-world systems we attempt to describe with computation models are necessarily non-binary, or indeed, fuzzy.

Fuzzy rule-based systems offer a powerful yet concise format for representing complex systems in a compact, readable, and accurate manner. These systems are often derived from data using Fuzzy C Means clustering [1], and least squares output estimation. Fuzzy rule-based systems come in a variety of formats designed to address regression [2][3][4] and classification problems [5][6][7].

The application of fuzzy rules to computational modeling and knowledge extraction from data has a long and robust history, with significant research effort having been applied in this area for some time [8][9][10][11][12][13][14][15][16]. Fuzzy rules, in varying formats to accommodate their specific tasks, have been applied to a huge range of modeling problems, and have found success as applied to many areas of ongoing research. In this study, we examine fuzzy rules from several different perspectives, performing rule quality meta-analysis in the form of fuzzy rule stability criteria, exploring the application of information granulation to fuzzy rules, applying fuzzy rule-based systems to a boosted ensemble, and constructing a hierarchical rule-based architecture.

The question of fuzzy rule quality is one which is classically addressed through the evaluation of predictive performance; however, we can easily recognize that this is not the only facet of a high-quality rule-base. In this dissertation, we propose the concept of fuzzy rule stability, defining rule stability to be the ability of a dataset and algorithmic combination to consistently produce the same or similar rules from the same or similar data. In this sense, we view fuzzy rule stability as how well the rules represent underlying patterns or knowledge in data, as reflected by how readily they are reproducible. We assert that if a rule is readily reproducible from the data then it is of high quality as it would appear to describe stable system knowledge. In this study, we seek to quantify the concept of rule stability by introducing three numerical stability metrics, each of which aims to capture a different facet of rule stability, and which can be used in combination to assess the overall stability of rules produced from a given methodology.

Traditional fuzzy rule formats express underlying fuzzy sets and fuzzy memberships functions through single numerical representations, obfuscating the complex underlying fuzzy landscape from the reader. This inability to communicate the full complexity of the model to the reader is undesirable as it limits interpretability and human readability. The study of information granules provides an avenue for improving readability by relaying more useful information to the reader, while maintaining an easy to understand format. Information granulation has been extensively studied in the literature [17][18][19][20][21][20][22][23], and its applications frequently overlap with fuzzy sets and systems due to their mutual interest in representing complex structures in a simple way, as well as their imprecise natures. In this dissertation, we apply interval-based information granules to the output parts of fuzzy rules, with the goal of providing a rule format which simultaneously transmits more crucial system knowledge to the user, while also providing a clear, easy to understand rule format which does not require extensive effort to interpret.

Ensemble learning is the concept of predicting system behavior using a set of models in combination, with the hope of producing better overall performance from the ensemble (group of models) than a single model could provide on its own. The idea behind these strategies is that diverse component models can make up for the weaknesses of other component models when making predictions as a group. Boosting is a well-known variation of the idea of ensemble learning, which strictly defines an iterative process for generating a weighted ensemble. Boosting functions by tracking data weights, where a higher weight indicates that the current ensemble does not adequately describe the behavior of that instance. These weights are updated at each iteration to reflect the new ensemble state, and weights are used in the computation of new learners with the goal of addressing the weakness of the existing ensemble. Boosting was originally proposed as AdaBoost by Freund and Shapire [24][25] who provided a well-defined framework for the generation of a weighted ensemble in a two-class environment. Since this initial study, boosting has been the focus of an enormous amount of research and it has been applied to a wide range of topics and problems. Despite this attention, the application of fuzzy models in a boosted ensemble has not been extensively studied, with only a few cursory papers being published to date [26][26][27]. In this dissertation, we provide some first steps in filling this void by proposing a methodology for boosting with fuzzy rule-based systems. The goal of this research is to attain the improved predictive power of a boosted ensemble, while maintaining the advantages of fuzzy rules, including interpretability and relative simplicity. We propose a novel weak learner architecture for use in a standard boosted ensemble and demonstrate the ability of the ensemble to successfully improve classification accuracy as compared to a single fuzzy rule-based system.

While fuzzy rule-based systems provide a powerful tool for generating predictive models, their efficacy, both with respect to predictive power and interpretability, is drastically lowered in the presence of high dimensional data. This drawback is known as the curse of dimensionality and is qualified as the exponential growth of a problems search space as the number of features increases. This causes a serious headache for rule-based modelling, as the number of possible combinations of linguistic terns becomes very large, meaning that we either accept poor performance through a small, readable number of rules, or accept rule explosion in an attempt to accurately model a system. An additional drawback to rule explosions is its effect on human readability. Research in psychology has demonstrated that the human mind does not meaningfully comprehend large sets of objects at one time, and this human limitation affects model readability when we model a large number of features. A candidate solution, which is not well explored in the literature, is the construction of a hierarchical fuzzy rule-based model. By generating and evaluating rules in a hierarchical architecture (as opposed to a flat architecture) we can represent significantly more complex constructs, while maintaining a small number of simple rules at each level of the hierarchy. In this study, we propose a novel hierarchical rule-based model architecture which seeks to not only improve the readability of the model, but also to reduce the overall model size and improve predictive ability as compared to a flat model of similar size. This is achieved through the construction of a cascading topology in which simple rules describing a limited number of features are built in a hierarchical manner. These rules are connected through their output parts, where predictions from previous layers are considered in subsequent computations, effectively refining the predictive process through the addition of further system knowledge.

In this dissertation, each of these topics are presented in detail as distinct studies, and the proposed methodologies are justified through extensive experimentation and analysis using publicly available real-world datasets.

1.1 Research objectives and Originality

The key objectives of the presented research are as follows:

- To develop a set of quantitative criteria for assessing the quality of fuzzy rules from the perspective of fuzzy rule stability, to demonstrate the application of the defined criteria through experimentation, and to provide general guidelines for the use of the stability criteria in assessing model quality.
- To define a methodology for the generation of interval based granular fuzzy rules from data using hierarchical clustering as a vehicle for rule extraction, to assess the feasibility of hierarchical clustering as a tool for rule generation, and to define and assess evaluation techniques for granular fuzzy models.
- To propose a workflow for the application of fuzzy rule-based systems as the component weak learner in a boosted ensemble, to demonstrate the successful improvement of classification accuracy compared to a single learner, and to assess the performance of the fuzzy rule ensemble as compared to standard weak learners.
- To design, construct, and evaluate a novel hierarchical fuzzy rule-based architecture with the goal of improving the predictive power of fuzzy rule-based models, to fight the curse of dimensionality, and to maintain or improve the interpretability of complex systems.

These topics exhibit several aspects of novelty which expand the existing research on fuzzy rules in several distinct categories. Additionally, the presented research raises further questions, providing opportunities for future study on the presented topics in most cases. The proposed stability criteria are of paramount novelty, with no other quantitative work having been developed in this field to our knowledge. The application of fuzzy models to a boosted ensemble has seen extremely limited attention and no other study addresses the use of a fuzzy rule-based system as a component weak learner. The hierarchical structure proposed is, to our knowledge, unique and demonstrates a clear improvement in predictive power and interpretability. Our use of hierarchical clustering in fuzzy rule extraction has not been previously examined, to our knowledge, and the application of information granules is generally somewhat limited.

Each of these topics exhibit significant novelty in their specific areas of research and prompt further opportunity for study. These topics make use of existing work and well-known methodologies, but with key aspects of originality.

These topics demonstrate originality in the following specific aspects:

- A novel evaluation of fuzzy rule quality quantifiably expressed through the evaluation of stability metrics which are novel both in their formulations and method of computation.
- A novel approach to granular fuzzy rule generation using hierarchical clustering as the vehicle for rule extraction from data.
- The novel use of fuzzy rule-based models as the component weak learner in a classic boosted ensemble, with novel modifications to the fuzzy rules to maximize adaptability.
- A novel model architecture in the form of a hierarchical rule-based model, and a novel methodology for the extraction of this structure from data.

1.2 Dissertation Organization

The chapters of this document are structured as follows:

Chapter 2: State of the Art

This chapter offers a focused literature review on those topics fully relevant to the research presented in this dissertation, including the state of fuzzy modelling with a focus on fuzzy rule-based systems and their analysis, the uses of information granulation in fuzzy rule-based systems, the applications of boosting in both fuzzy and non-fuzzy contexts, and a focused examination of hierarchical fuzzy structures.

Chapter 3: Theoretical Background

This chapter presents the details of many established algorithms and processes which are used in the construction of novel fuzzy models and other research in later sections. The topics covered include Fuzzy C-Means clustering, extraction of fuzzy rules from data, boosting, gradient descent, particle swarm optimization, hierarchical clustering, and cluster validity indices.

Chapter 4: Rule Stability Criterion

This chapter presents three novel fuzzy rule stability criteria and outlines their definitions, uses, and applications to the successful analysis of a fuzzy rule-based system. This chapter contains technical definitions of the criteria, experimentation, and discussion of the obtained results with respect to the implications on rule stability and quality.

Chapter 5: Fuzzy rules from hierarchical clustering with information granules

This chapter provides a methodology for the formation of partially granularized fuzzy rules, and their extraction from data. This topic experiments with the use of hierarchical clustering as a vehicle for rule extraction from data, and experimentally compares this methodology to the well-known Fuzzy C-Means methodology. We additionally assess the performance of the granular models through specialized evaluation measures and discuss the usage of these measures in quantifying granular performance.

Chapter 6: Boosting with Fuzzy Rules

This chapter is concerned with the application of small fuzzy rule-based systems as the component weak learners in a boosted ensemble. We present a specialized fuzzy classification rule architecture for maximizing learner flexibility in a boosted environment and present the necessary changes to established algorithms for considering boosting data weights in model generation. Further, we experimentally demonstrate the improved performance of the boosted ensemble as compared to a single learner and compare the proposed methodology to standard weak learners.

Chapter 7: Hierarchical fuzzy rule-based modelling

In this chapter we present a novel hierarchical fuzzy rule-based architecture and define a complete methodology for the extraction of this architecture from data. We present extensive experimentation showing the improved performance of the new topology versus a flat fuzzy rule-based system and discuss the interpretability implications of this type of model format.

Chapter 8: Conclusions and Future Studies

The work presented in this dissertation is summarized, and we identify both the limitations of the current work and various directions of future research on the topics presented.

2 STATE OF THE ART

A critical aspect of modern science and engineering is the topic of computational modelling and knowledge extraction from data, as applied to real world problems and systems. There is a desire to develop computational and mathematical methods for creating robust models which can provide key insight into real world systems, presenting the extracted system knowledge to expert users in an understandable way, and for these models to make high quality, accurate predictions regarding the behavior of complex systems. In many fields, it is critical that predictive models are human-readable, as we need to know both the predicted behavior and what knowledge has been applied to reach that conclusion. As many of these systems represent real-world phenomenon, the application of binary logic to these fundamentally analog problems does not always make sense. The use of fuzzy logic and fuzzy modes to describe real world systems has shown a great deal of promise, as fuzzy models are able to provide concise meaningful knowledge representation in a human-readable format.

The concept of fuzzy logic, and by extension fuzzy sets, has been around since the 1960s when it was first proposed in the famous inaugural paper *Fuzzy sets* by Zadeh [28]. This founding work laid the framework for an entire field of study by proposing a logic system in which truth values and set memberships could be defined in a non-binary fashion. Further seminal works over the years include the definition of fuzzy relation by Sanchez in the 1970s [29], the development of triangular norms by Menger in 1942 [30], and then later applied to fuzzy systems by Sklar in the 1960s [31]. These studies, along with a large host of additional research in the area of fuzzy logic and fuzzy sets, have resulted in a well-defined robust fuzzy mathematical framework. Notable works in the area include seminal papers on fuzzy logic control by Lee [32], on fuzzy rules by Dubois and Prade [33], and on linguistic quantifiers and information granules and their applications in a fuzzy framework by Zadeh [34][17].

The remainder of this section provides a focused literature review on several specific topics of fuzzy modeling, focusing on those studies most relevant to the research presented in later chapters of this dissertation.

2.1 FUZZY RULES AND THEIR GENERATION FROM DATA

The first appearance of fuzzy rules in literature was by Mamdani in his famous paper on industrial processing plant control [35]. In this paper, Mamdani proposed that a simple chain of *if-then* rules could be used to successfully control a complex system, such as an industrial plant. This type of rule-base later became known as a Mamdani style fuzzy rules, and the concept was generalized by Takagi and Sugeno

[36] allowing for arbitrary functions as the consequent part of the rules. This generalized form has since been known as Takagi-Sugeno style fuzzy rules, or TS fuzzy rules. While the initial Mamdani style rules targeted linguistic quantifiers, Takagi-Sugeno style rules formalized the ability of rules to predict in a real valued output space, making them candidate solutions to continuous function estimation or regression problems. Additionally, more recent works have derived formats for the application of fuzzy rules to classification problems [37][38]. Since these introductory studies, there have been countless developments and applications of fuzzy rules in the literature, and their study is the focus of significant ongoing research.

A critical aspect of fuzzy rules is their ability to be extracted from data. This topic has received a great deal of research attention over the years, and a wide variety of methodologies are available. These include strategies using evolutionary or genetic algorithms [39][40][41][12][42][43][44][4][45][37], particle swarm optimization [46][47] and other optimization techniques [48][3][49][50], support vector methodologies [14][51][52], and input space division strategies [53][8][54], amongst others [13][15][55][56]. This is an open topic with many recent studies and ongoing research efforts. In the work presented in this dissertation, we are primarily focused on the use of fuzzy clustering techniques for rule extraction from data, a topic which has been extensively covered in the existing literature [57][48][58][59][60][61][62][63]; however, it is critical to note that, as shown through the extensive number of available references, there are many feasible approaches to rule extraction, all of which have shown experimental success.

The inaugural studies on fuzzy rule-based systems proposed a flat rule base in which rules are only considered as a single IF-ELSE chained architecture. More recent studies have proposed the application of fuzzy rule structures in different architectures in attempts to improve performance, reduce the number and complexity of rules, or address some other specific issue. Some example formats proposed in the existing literature include hierarchical schemes [54], ensembles [64] and boosted ensembles [65][26][27][66][6], and fuzzy rule trees [67] or forests [68].

Due to their powerful predictive properties, adaptability to different situations, and interpretability, fuzzy rule-based systems have been applied to a huge number of problem spaces. Existing applications of fuzzy rule-based systems include medical diagnostics [69][70][7][71][72][73][74] and medical imaging [75], image classification and computer vision [6][76][77], learning assessment [78], financial decision making [79] or other economic or stock related problems [38][80][81], signal transmission problems [82], traffic and other transportation problems [83][84][85], weather forecasting [68], biometrics [86], intrusion detection [87], soil spectroscopy [49], hydrology [88], software reliability [89], recommender systems [90], and other engineering systems [91][92][93], to list some recent examples. Many of these applications highlight domains where model interpretability is an important factor in choosing which formats are best suited to a given problem, and the significant use of fuzzy rules in these fields further motivates a research focus on interpretability alongside predictive accuracy.

In additional to the research attention allotted to the generation and application of fuzzy rules, fuzzy rules have additionally prompted a great deal of related meta-research. This includes such topics as cluster validity indices for determining the correct number of rules for a given system [94][95][96][59][97][98][99][100] and the analysis of various aspects of rule quality and interpretability, along with studies aimed at addressing these issues [101][102][103][104][105][106][107], to highlight two relevant and well-studied areas. Choosing an appropriate number of rules is an important factor in experimental setup, and we make use of such indices in both Chapters 4 and 7. The study of rule interpretability, especially those topics discussed in [102], is of critical importance to our development of a set of rules stability criteria, as it is a similar aspect of qualitative rule quality we are attempting to quantitatively capture. Interpretability is also a major consideration in the formation of information granules in Chapter 5 as well as a motivating factor in the formation of a hierarchical architecture in Chapter 7, and we draw on many of the concepts discussed in these studies in our own work and in the analysis of the proposed methodologies.

Fuzzy rules continue to be a significant research topic with much ongoing work in the area as their application to human-centric computing continues to show positive results.

2.2 INFORMATION GRANULATION IN FUZZY MODELS

Information granulation comes naturally as a symbiotic partner to fuzzy logic, as it provides a mechanism for improved linguistic representations of knowledge. Information granules have been used extensively in fuzzy modelling, including applications involving fuzzy rules, and have been the topic of substantial standalone research. Generally, research on information granulation seeks to provide solutions to interpretability and human centricity problems arising from traditional machine learning algorithms, moving computational models towards a format better suited to describe linguistic quantifiers. This concept is outlined by Zadeh in his well-known paper “Fuzzy logic = computing with words” [34], where the fundamentals of the concept are laid out. Further studies, including [17], [108], and [109], have contributed to formalizing various aspects of information granulation over the years.

In fuzzy rule-based systems, information granulation has been applied to classification problems [110], regression problems [111][112][113], and time series problems [114]. The application of information granules to regression style fuzzy rules notably includes the research presented in Chapter 5 of this dissertation [111], which makes use of the existing work in this field. Additional applications of information granulation include their application to fuzzy radial basis networks [22], their application in forming linguistic representations from membership functions [21] or fuzzy clusters [20], their use in novel clustering algorithms [115][116], applications in hierarchical models [23], the development of higher order information granules [117], their use in fuzzy cognitive maps [118], and in knowledge transfer [119]. As demonstrated by the existing literature, the scope of potential applications is broad; however, it is interesting to note that there does not appear to be any singular application which has received a large amount of focused attention.

Shadowed sets define a three valued set representations [19], which can be used to provide a more interpretable view of a given membership function [18] and represent a specific form of information granule, which is extracted from a fuzzy set [120]. While the literature on this topic is limited, studies making use of shadowed sets have included automatic selection of threshold parameters in clustering [121] and other applications in fuzzy clustering algorithms [122][123], the selection of data subsets for neural network training [124], and the approximation of fuzzy numbers [125]. Rough and fuzzy rough sets [126][127][128] are closely related to shadowed sets and information granules, containing lower and upper approximation sets which define the *rough* designated area for a given set. Rough sets have been applied to fuzzy modelling in a handful of studies, including fuzzy-rough feature selection [129], image classification [130], image compression [131], and fuzzy rule interpolation [132].

2.3 BOOSTING IN A FUZZY ENVIRONMENT

Boosting was first developed in the 1990s by Freund and Schapire [24][25] as AdaBoost. Since this initial study, there has been significant research on the topic of boosting, and additional algorithms, improvements, and variants have been proposed. Outside of the original binary classification algorithm, studies have proposed altered algorithms for gradient boosting [133][134][135][136] and modifications to handle multi-class problems [137][138].

Due to the widespread, generally successful application of boosting in many different fields, boosted ensembles have been proposed as candidate solutions to many different computational problems. Some example applications include speech recognition [139], population dynamics [140], various image classification tasks [141][142][6][143][144] and image feature detection tasks [145][146][147][148] [149][150][151][147][152], defect prediction [153], fault detection [154], bankruptcy prediction [155][156][157][158], travel time prediction [134], insurance cost modelling [159], solar power forecasting [160], resource consumption modelling [161][162], gender recognition [163], natural event prediction and assessment [164][165], fingerprint classification [166], smoke detection [167][168], and medical diagnosis [169]. While this list of boosting applications is substantial, it is only a small fraction of the full literature on the topic, as the range of application is very extensive.

Boosted ensembles have also been studied in a fuzzy context, although the existing literature is quite limited. Fuzzy applications of boosting include a handful of works on the generation of individual fuzzy rules through boosting [27][66][26][170] and similar rule building methods [5]. Additional research in the realm of fuzzy systems has been done to compare the use of fuzzy versus non-fuzzy operators for ensemble prediction [171], boosting with the related concept of granular models [172], and fuzzy classifier ensembles [64]. As indicated by the comparatively small amount of literature on the topic of fuzzy boosting as compared to the plethora of non-fuzzy studies, this topic has not received a great deal of research attention. Both the use of fuzzy learners in a boosted ensemble, and the fuzzification of boosting techniques have not been thoroughly studied, and this gap in the existing literature provides motivation for our work on this topic in Chapter 6 of this dissertation.

2.4 Hierarchical Fuzzy Models

Hierarchical fuzzy modelling has different meanings in different contexts, and for this reason the topic may cause some confusion to the reader. We can separate hierarchical fuzzy models into two primary categories: those studies which generate fuzzy models through hierarchical methodologies and those studies which construct a fuzzy model consisting of a hierarchical topology or architecture.

Regarding hierarchical model generation techniques, several studies exist applying different approaches. Certain studies such as [5] and [173] are hierarchical only in their use of divisive “hierarchical” model generation strategies, used to compute high quality or highly specific rules. In these studies, the generation procedure is the hierarchical aspect of the study and the resulting fuzzy model remains flat. As such, this type of hierarchical model is not of particular relevance to the proposed hierarchical research in this dissertation.

Some studies propose hierarchical fuzzy architectures taking novel forms, often based on similar non-fuzzy model architectures. This includes [174], [175] and [176], which form tree-like structures where higher level layers act to direct the flow of the decision making process through the model. In these studies, the branches of the fuzzy trees do not contribute to the decision-making process but act only to direct the traversal of the tree to the appropriate leaf node where a prediction is made. These studies all form hierarchical fuzzy structures resembling a decision tree. The hierarchical aspect of these studies is in the topological design of the model; however, the relegation of output prediction to only the lowest levels of the tree marks a clear distinction between these types of models and our proposed hierarchical rule-base.

Of the remaining studies concerned with hierarchical fuzzy architectures, much of the existing literature is focused on strictly two-level systems. This includes [177] which generates a two level structure through genetic algorithms, [117] which forms hierarchical information granules, and [178] which uses a higher-level generalized rule structure with lower level sub-rules optimized through particle swarm optimization. These are examples of simple hierarchical topologies which serve a very specific purpose, often generating specialized sub rules to improve performance.

Finally, we arrive at the topic of general fuzzy hierarchical structures, focusing on those studies which concern themselves with fuzzy rules. In these studies, more generalized hierarchical topologies are proposed for classification or function estimation. In [179], the authors propose a specialized hierarchical structure in which poorly fitted data is set aside after generating each hierarchical level, forming increasingly specialized rules as the algorithm progresses. Another study, [180] proposes a much simpler hierarchical structure in the form of a feed-forward fuzzy rule-based network, using the predictions of previous layers as inputs to the next layer of rules with the goal of preventing rule explosion. Two studies by Joo and Lee, [181] and [182] define a hierarchical feed-forward structure in which the predictions from previous rules are used in the output part of subsequent rules, avoiding interpretability issues arising from a lack of physical meaning to intermediary values. A similar study [183] concerns similar topics but discusses a few different hierarchical topologies, but more as a thought experiment, as experiments are very limited. All these papers are simple introductory works in which an idea is proposed, generally with the aim of reducing the number of involved rules in a system, and in each case the proposed research fails to examine any data-driven modeling, relying fully on expert developed rules. Despite these limitations, the work proposed in [180], [181] and [182] represents the most relevant existing work to the research proposed in Chapter 7 of this dissertation, as they propose similar hierarchical architectures and these studies are primarily limited by their lack of either a well-defined rule generation procedure or extensive experimentation.

2.5 FINAL REMARKS

This chapter identifies and discusses a great deal of existing literature relevant to this thesis. The work presented in this dissertation builds upon the research established in many existing studies, including but not limited to those studies discussed in this chapter. Some of the cited literature represents foundational work in its field from which entire areas of interest have arisen, and many of these concepts are used extensively throughout this dissertation.

3 BACKGROUND

This chapter provides the complete necessary background knowledge for the research topics presented in Chapters 4 through 7. The information contained in this chapter assumes a basic familiarity with fuzzy logic and fuzzy sets, as well as a foundational understanding of machine learning. This section provides the specifics of Fuzzy C-Means clustering, Hierarchical Clustering, Particle Swarm Optimization, the design and extraction of TS-fuzzy rules from data, Boosting, and Gradient Descent.

3.1 FUZZY C MEANS

Fuzzy C-Means (FCM) [1][185] is well-known fuzzy clustering algorithm, in which a fuzzy partition is formed on an input data set. Fuzzy clustering is similar to Boolean clustering, with the key difference that fuzzy clusters are not mutually exclusive, and that cluster assignments are non-binary. These two criteria are interlaced, as it is fuzzy membership which allows for instances to "belong" to more than one cluster to varying degrees.

The goal of FCM clustering is to form a fuzzy partition from the provided data. Formally, this results in the computation of two key structures – the fuzzy partition matrix, *U*, defining data membership to clusters, and the cluster prototypes, *V,* defining the cluster centers. FCM clustering is computed for a given number of clusters, *c*, provided as an input parameter to the algorithm, and requires a fuzzification coefficient, *m,* which controls the "fuzziness" of the computed partition (this value is commonly taken as *m* = 2.0 in the literature).

FCM clustering is realized through the minimization of an objective function, *Q*:

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^{m} \, ||\boldsymbol{x_k} - \boldsymbol{v_i}||^2 \tag{3.1}$$

where *c* is the number of clusters, *N* is the size of the input dataset, $\boldsymbol{x_k}$ is the *kth* input data instance, $\boldsymbol{v_i}$ is the *ith* cluster prototype, *m* is the fuzzification coefficient, u_{ik} is the fuzzy membership of the *kth* instance to the *ith* cluster, and || . || is a distance function, commonly taken to be Euclidean distance or a scaled version thereof.

The algorithm is initialized by randomizing *U*, and then proceeds iteratively by first updating the cluster prototypes:

$$\boldsymbol{v_i} = \frac{\sum_{k=1}^{N} u_{ik}^{m} \boldsymbol{x_k}}{\sum_{k=1}^{N} u_{ik}^{m}}$$

(3.2)

followed by the re-computation of the partition matrix with the new prototype locations using the following membership function:

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{x_k} - \boldsymbol{v_i}\|}{\|\boldsymbol{x_k} - \boldsymbol{v_j}\|} \right)^{2/(m-1)}}$$

(3.3)

Where (3.2) is computed for each cluster, $i = 1, 2, \ldots c$, and (3.3) is computed for each data instance and each cluster, $i = 1, 2, \ldots c$ and $k = 1, 2, \ldots N$.

The algorithm is stopped when the given termination criteria are met; commonly either a maximum number of iterations or when the change in the partition matrix is less than some specified threshold value.

$$\|U(t) - U(t-1)\| < \varepsilon$$

(3.4)

Where t indicates a given iteration index, and ε is the specified error threshold. The exact algorithm is readily available in the literature [1][184].

3.2 HIERARCHICAL CLUSTERING

Hierarchical clustering is a well-known clustering algorithm which computes data groupings (clusters) in a hierarchical manner, as the name implies. This form of clustering is generally based on a nearest neighbor distance function, which provides the basis for hierarchical groupings [185][186]. There are two possible approaches to forming a hierarchical partition: Top down (a divisive approach) or bottom up (an agglomerative approach). Although both approaches form valid partitions, the bottom up approach is significantly more computationally efficient, and is the focus of this section.

Bottom up hierarchical clustering is realized by grouping data points closest to one another to form *sub-clusters*, and then iteratively combining these sub clusters in a hierarchical manner. Which data points and sub-clusters are combined at what time is determined through a *linkage policy*. The three most common linkage policies are as follows [186]:

Average linkage: one determines an average distance of the data to be clustered to the points located in the sub cluster.

Single linkage: the strategy is based on determining the shortest distance of the data to be clustered to the data occurring in the sub cluster.

Complete linkage: the strategy is based on the determination of the largest distance of the data to be clustered to the data present in the sub cluster.

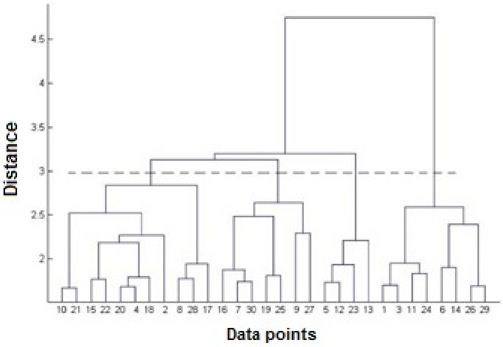


Figure 3.1: Dendrogram showing hierarchical clustering structure. Dotted line indicates a cluster cutoff strategy for c = 4

A useful byproduct of hierarchical clustering is that the algorithm naturally produces a graphical visualization of the clustering structure known as a *dendrogram,* an example of this structure is given in Figure 3.1. This is the raw result of the clustering process, showing the hierarchical linkage of the clustered data. This can be a useful tool for assessing the structure of the data, or for determining which linkage policy suites the needs of the problem best.

To obtain formalized clusters from a dendrogram we use one of two methods: 1) *cutoff*: a point in the tree is chosen as the cutoff point, and the subtrees resulting from this cut are taken as clusters. 2) *distance*: some distance between two sub clusters is given as a maximum for which clusters can merge.

The dendrogram in Figure 3.1 shows an example of the *cutoff* method, denoted by the dotted line. In this case, we chose a cutoff which forms exactly 4 clusters. Dendrograms can be visually useful in determining the natural number of clusters, as we can examine the high-level structure of the graph. For example, in Figure 3.1, the rightmost sub-cluster is clearly distinct from the rest of the graph.

The *distance* strategy is useful as it is capable of automatically selecting a natural number of clusters; however, as distances are highly data-set dependent, this method requires an intimate knowledge of the dataset or extensive experimentation to determine an appropriate value.

Hierarchical clustering is realized in an iterative manner. At each iteration, the current sub-clusters (and un-clustered data points) are examined in pairs to determine which pair should be merged, according to the specified linkage policy. The identified pair is then merged, the data structure is updated, and the process continues until a complete hierarchical structure (dendrogram) is formed. In this dissertation we make use of existing hierarchical clustering implementations found in the MATLAB machine learning toolbox.

3.3 PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) [187] is an optimization procedure which aims to mimic swarm behaviors often observed in nature [188]. The algorithm functions by defining a set of particles which reside within some error space and are capable of "moving" through the error space and evaluating their current position. At each iteration the particles update their position based on their individual historical best position, and a universally known global best position. To help avoid local minima, a degree of randomness if injected into the particle's movement patterns, helping to insure a more thorough exploration of the error space.

Each particle retains three individual attributes – its personal best position (historical), its current position, and its velocity. These values are used in combination with a universally known global best position to iteratively update the particles velocities at each iteration:

$$\boldsymbol{v}_i^{t+1} = \boldsymbol{v}_i^t + \alpha\boldsymbol{\epsilon}_1[\boldsymbol{g}^* - \boldsymbol{x}_i^t] + \beta\boldsymbol{\epsilon}_2[\boldsymbol{x}_i^{*(t)} - \boldsymbol{x}_i^t] \tag{3.5}$$

Where $\boldsymbol{v}_i$ and $\boldsymbol{x}_i$ are the velocity and position of the *ith* particle, *t* indicates a given iteration, and $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are two random vectors taking values in the unit interval, and $\boldsymbol{g}^*$ is the global best position. The parameters α and β are learning parameters (also known as acceleration constants), and they are often taken to be around 2.

Computed velocities are then used to update particle positions:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1}$$

(3.6)

There are several proposed variants to the original PSO algorithm. One notable, and largely successful, improvement is the addition of a momentum term or inertia function $\theta(t)$:

$$\boldsymbol{v}_i^{t+1} = \theta \boldsymbol{v}_i^t + \alpha \boldsymbol{\epsilon}_1[\boldsymbol{g}^* - \boldsymbol{x}_i^t] + \beta \boldsymbol{\epsilon}_2[\boldsymbol{x}_i^{*(t)} - \boldsymbol{x}_i^t] \tag{3.7}$$

Where θ evaluates to a value between 0 and 1. In the simplest case, we take the inertia function as a constant value. This addition introduces a virtual mass to the particle, with the intent of making the algorithm converge more quickly [189]. Other improvements include accelerated PSO which does away with each particles individual best, relying solely on the global best [190]. While this algorithm seems overly simplistic, studies have shown that this algorithm is still capable of fast global convergence[191].

3.4 THE DESIGN OF FUZZY RULES

Fuzzy rules are a powerful machine learning tool which offer many modelling advantages. Their combination of predictive power and readability makes them well-suited for many areas of research where the need to understand the reasoning behind model behavior is crucial. In their most general form, a fuzzy rule takes the following format:

$$IF\ \boldsymbol{x}\ is\ A_i\ THEN\ y\ is\ f_i(\boldsymbol{x}, p_i) \tag{3.8}$$

Where $\boldsymbol{x}$ is the input to the fuzzy rule, A_i is a fuzzy set, y is the predicted output for the instance given as a function, $f(\boldsymbol{x}, p_i)$ where p_i is a functional parameterization. In this format, the functional consequent of the rule can be any function, although polynomials are most common. This results in a rule which defines a fuzzy functional mapping between the input and output space of a problem. This rule format is known as Takagi-Sugeno (TS) [36] fuzzy rules, and their use in machine learning and fuzzy modelling has been extensively studied.

By computing a collection of fuzzy rules and using them as a set of IF-ELSE conditions, we form a fuzzy rule-based system (FRBS). As we are concerned with *fuzzy* rules, to whom membership is defined to a degree, the rule base may involve multiple rules, to varying degrees, in the decision-making process. Specifically, the output of a FRBS is given by:

$$\hat{y}_k = \sum_{i=1}^{c} A_i(\boldsymbol{x_k}) f_i(x_k, p_i)$$

(3.9)

Where $\hat{y}_k$ is the predicted output for the *kth* data point.

If rule consequents are constant values, we refer to the rule format as Mamdani style fuzzy rules [35]:

$$IF\ \boldsymbol{x}\ is\ A_i\ THEN\ y\ is\ b_i$$

(3.10)

Where b_i is a constant valued output to the rule, representing the center of an underlying fuzzy set. This can be interpreted as a 0^{th} order polynomial, making them a special case of TS-fuzzy rules. As discussed in Chapter 2, this was the original proposition of FRBSs, and the generalized format was developed later.

In a similar fashion, the output of this style of rule-base system is given by:

$$\hat{y}_k = \sum_{i=1}^{c} A_i(\boldsymbol{x_k}) b_i$$

(3.11)

Where b_i is the constant valued output of the *ith* fuzzy rule. This rule format has the advantage of being significantly more readable and interpretable to a human agent wishing to analyze the response of a model, and in many scenarios constant outputs can be interpreted as linguistic terms.

3.4.1 The generation of fuzzy rules from data

The goal of most machine learning research is to extract useful system knowledge from simple data. This allows collected data to be used in a useful manner, often with the goal of predicting future system behavior. There are a variety of ways through which fuzzy rules can be extracted from data, one of which is using fuzzy clustering and output estimation.

The computation of fuzzy rules is considered in two parts – the condition and consequent parts of the rule. A common approach is to use FCM clustering to generate rule conditions. Using this procedure, clustering is performed only in the input space of the considered dataset, and the cluster prototypes are used to form the conditional fuzzy sets for the rule-base. Considering the format given in (3.8), we utilize the *n-dimensional* fuzzy sets defined by the combination of the memberships in U_i and the prototype $\boldsymbol{v_i}$ to denote the condition part of a rule A_i. The prototype location denotes the center of A_i, and the shape of the fuzzy set is described by the fuzzy partition in U_i.

The output parts of fuzzy rules are formed using least squares estimation. For the simplest case, Mamdani style rules, the format is given in (3.10). In this format, rule consequents are constants and the problem can be formulated as follows:

$$\boldsymbol{y} = U\boldsymbol{b}$$

(3.12)

Where $\boldsymbol{y}$ is the vector of N target output values, U is the partition matrix from fuzzy clustering, and $\boldsymbol{b}$ is a vector of constant values (rule consequents). In this case, $\boldsymbol{b}$ is the unknown to be solve for, and the problem takes the form of a well-known matrix equation with several known solutions. In many cases, least squares estimation is the most practical solution, as matrix inversion is computationally time consuming.

In the case of higher order TS-style fuzzy rules, the format for which is given in (3.8), the output estimation problem becomes more complex. For example, the linear case is described by:

$$y = \boldsymbol{b_i}^{\boldsymbol{T}}[\boldsymbol{x_k}, 1]$$

(3.13)

Where $\boldsymbol{b_i}$ is now a vector of functional parameters of length n+1 for n-dimensional data. The additional of the constant value, 1, to the input vector serves as the intercept in the computed linear equations. This calculation results in linear outputs:

$$y \ = \ b_1\, x_1 \ + \ b_2\, x_2 \ + \ \dots + \ b_n\, x_n \ + \ b_{n+1}$$

(3.14)

Where x_j indicates the *jth* feature of the instance under consideration. The solution matrix also becomes more complex:

$$\boldsymbol{z_{ik}} = u_{ik}[\boldsymbol{x}_k\,, 1]$$

(3.15)

$$Z = \begin{bmatrix} \boldsymbol{z_{11}} & \cdots & \boldsymbol{z_{c1}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{z_{1N}} & \cdots & \boldsymbol{z_{cN}} \end{bmatrix}, b = \begin{bmatrix} \boldsymbol{b_1} \\ \dots \\ \boldsymbol{b_c} \end{bmatrix}, \text{and } y = \begin{bmatrix} y_1 \\ \dots \\ y_N \end{bmatrix}$$

(3.16)

Here *Z* is a (*c x (n+1))* by *N* matrix containing each $\boldsymbol{z_{ik}}$ for i = 1, 2, … *c*, and *k* = 1, 2, … *N*, *b* is a *c x* (*n +1)* length vector representing all the parameters of a linear functional output, and *y* is a *N* length vector of target values from data. We now have a parameter estimation problem in the form:

$$y = Z\boldsymbol{b}$$

(3.17)

This is a very similar problem to the one presented in (3.12), although *Z* is much larger than *U*, and a solution can be computed in the same manner. Once functional parameters have been calculated, predictions can be computed by solving the following linear equation:

$$\hat{y}_k = [\boldsymbol{z_{1k}}^{\boldsymbol{T}}\ \boldsymbol{z_{2k}}^{\boldsymbol{T}} \dots \boldsymbol{z_{ck}}^{\boldsymbol{T}}] \begin{bmatrix} \boldsymbol{b_1} \\ \dots \\ \boldsymbol{b_c} \end{bmatrix}$$

(3.18)

The result of (3.18) is a fuzzy membership weighted averaging of each rule's predicted output according to the linear functions defined by $\boldsymbol{b}$.

This process of output estimation can be extended to higher order polynomials if desired, although the computational complexity of the estimation grows with the order of the polynomial [192].

3.4.2 Evaluation of fuzzy rule-based models

With any machine learning task, the ability to evaluate and assess the quality of a model is of the utmost importance. There are many ways in which models can be assessed, and those criteria pertinent to the research in this dissertation are outlined in this section.

3.4.2.1 Reconstruction Error

Without considering the predictive power of a fuzzy model, the simplest way to evaluate model quality is to assess how well the model describes the modelled data. One way this can be accomplished is through the evaluation of reconstruction error. Reconstruction error (RE) is, as the name implies, an assessment of how accurately a model can reconstruct the input data. As this is an error measurement, RE measures the discrepancy between the model's representation of the training data and the actual training data. We consider reconstruction error to be a normalized version of FCM's objective function, consider:

$$RE = \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}^{m}\|\boldsymbol{x_k} - \boldsymbol{v_i}\|^2$$

(3.19)

Which closely resembles (3.1).

This is a simple and intuitive measure for representing the accumulated error of a model with respect to its ability to represent the data; however, this evaluation metric has obvious drawbacks, most notably the fact that it does not consider a model's predictive accuracy, and that RE tends to decrease as model complexity increases.

3.4.2.2 Error for continuous outputs

The most common and intuitive form of model assessment is to consider a model's predictive accuracy. These measures provide an indication of how well a model predicts system behavior, considered over the scope of the evaluated dataset.

When dealing with continuous output domains, accuracy is measured as the difference between the predicted output and the actual output for a given data instance, aggregated across the dataset. A common aggregator is root mean squared error (RMSE):

$$RMSE = \sqrt{\sum_{k=1}^{N}\frac{(y_k - target_k)^2}{N}}$$

(3.20)

Where y_k represents the predicted output for a given instance, and $target_k$ is the actual output value from the dataset.

Another error evaluation metric is mean absolute error (MAE), which assess the predictive error of a model in a somewhat simpler manner:

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|y_k - target_k|$$

(3.21)

This measures the average difference between the predicted and actual outputs across the dataset. The key difference between these two measures is in their handling of extreme values. While MAE provides a very fair and even assessment of the error, treating all points equally, RMSE has the advantage of highlighting large errors, and minimizing small errors. While this is not always a desirable characteristic, it is often beneficial in computational modelling, as it provides an easier avenue for identifying certain types of modelling issues such as outliers or unbalanced datasets.

3.4.2.3 Classification Accuracy

When considering a classification problem, the calculation of accuracy is more straightforward. When classifying, the quality of a model is simply assessed through the percentage of correctly classified instances in the dataset:

$$Accuracy = \frac{1}{N}\sum_{k=1}^{N} I(c_k, predicted_k) \tag{3.22}$$

Where c_k is the class label of the *kth* instance, $predicted_k$ is the predicted class for the *kth* instance, and I is an identity function:

$$I(a,b) = \begin{cases} if\ a = b\ then\ 1 \\ \quad else\ 0 \end{cases} \tag{3.23}$$

Hence, the calculated accuracy indicates the percentage of correctly classified instances. For the most part, higher accuracy is always better.

3.4.2.4 Training, Testing and Overfitting

The previous sections outline equations for evaluating the predictive performance of computation models. In these sections, we state that higher accuracy/lower error are generally better; however, we must be wary of overarching generality, as it ignores the subtleties of modelling and the inevitability of incomplete or imperfect data.

One of the most important aspects of modelling which is not described by accuracy alone is the issue of overfitting. If we accept that the available data will never be a perfect representation of the underlying system, it stands to reason that a model which perfectly models the provided data does not necessarily perfectly model generalized system behavior – it may have only memorized the training data. This issue is known as *memorization* or *overfitting* and reflects the concern that a complex model runs the risk of learning the patterns of specific training inputs instead of learning useful generalizable knowledge about a system.

Memorization is often detectable when comparing training and testing performance, where testing performance is defined by the evaluation of a withheld dataset which has not been used to construct the model. In many cases, we observe that, as the model becomes more complex, there comes a point where training and testing performance begin to diverge. This is a sign of overfitting, as the algorithm may continue to improve training performance, but those improvements are not observed when evaluating the withheld testing dataset.

In Figure 3.2, we show the performance index (*Q*) for fuzzy clustering on a sample dataset, plotting *Q* for the training and testing partitions as the number of computed rules increases. We observe a growing gap between training and testing performance as the model becomes more complex (additional rules), indicating that memorization is being observed.

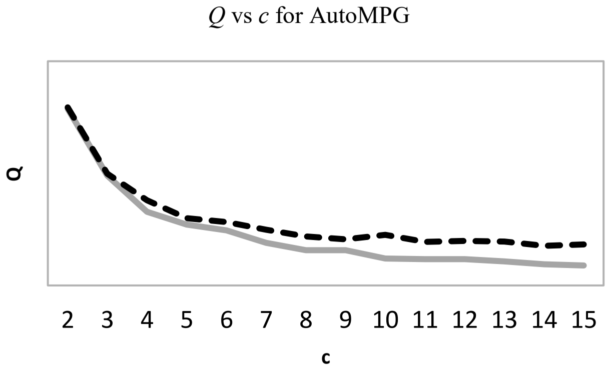


Figure 3.2: Performance index, Q vs the number of clusters, c for AutoMPG [194] (solid line is training data, dotted line is testing data)

There are a few techniques which assist in avoiding overfitting. The first is a simple split of the available data into training and testing partitions. This strategy constructs the model using only the data instances contained in the training partition, and exclusively uses the testing partition to evaluate the predictive power of a model after training is complete. This type of validation testing is crucial to high quality modelling, as testing accuracy provides a much better indication of actual predictive accuracy. Testing datasets help eliminate training biases, as we expect the model to respond well to those instances used to train it, and the way in which it handles instances it has not seen before is more pertinent. Using the knowledge gained from evaluating a testing dataset, researchers can halt training at a point where performance is maximized while memorization is minimized.

While the principle of a training/testing data split is sensible, there remain certain issues with this simple approach. First, as some data is never used in the computation of the model, we are forced to accept a lower quality model as it is likely possible to obtain better performance using the full dataset. Second, the specific split of the testing and training sets may skew performance in one direction or the other, depending on which instances end up where, or on the class/output designations in the testing data.

One way to avoid some of these issues is known as cross-validation. Cross validation is performed over a series of *k* folds, and a distinct model is trained from a distinct training/testing data split at each fold. Typically, the dataset is split into *k* equal partitions, and at each fold, the model is trained with the amalgamation of *k–1* partitions and tested with the remaining *kth* partition. Over the course of cross-validation, each partition is used as the testing partition once. Upon completion, an overall training and testing accuracy can be calculated as the average of the training and testing accuracies over all *k*-folds. The advantage of this method is that, at some fold, each instance is used as part of the test set, so the average performance of the model across all *k*-folds provides a better overall indication of model quality than a single split. The drawback to this method is that, for each experimental parameterization, *k* separate models need to be generated and evaluated – this can be computationally prohibitive in complex modelling cases.

3.4.2.5 Cluster validity indices

Many clustering tasks, including FCM clustering, require the number of clusters to be provided as an input parameter to the algorithm. This results in a pertinent research question regarding *correct* or *best* number of clusters for a given problem. The need to answer this question has resulted in the topic of fuzzy cluster validity indices. This field is well-studied, and the proposed indices are defined with the goal selecting the best number of clusters (rules) for a problem in an objective numerical fashion.

There has been significant research in this area, with different studies applying different priorities and ideas towards defining what constitutes the best choice. Existing indices typically use some combination of the input data, the fuzzy partition matrix, and the cluster prototype locations.

One of the first forays into fuzzy cluster validity indices was proposed by Bezdek in [39], and is known as the partition coefficient:

$$V_{PC} = \frac{1}{N}\sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{2} \tag{3.24}$$

This coefficient measures the degree to which clusters in a system share data instances, and the key weakness of this coefficient is that it shows a clear monotonically decreasing tendency as *c* increases, strongly favoring small *c* values. Since this early proposition, many additional indices of increasing complexity have been proposed, including the following well known options:

The Xie-Beni Index [97]:

$$V_{XB} = \frac{\sum_{i=1}^{c}\sum_{k=1}^{N} {u_{ik}}^{m}\|\boldsymbol{x_k} - \boldsymbol{v_i}\|^2}{N\, min_{i,j}\left\|\boldsymbol{v_i} - \boldsymbol{v_j}\right\|}$$

(3.25)

The Fukuyama and Sugeno Index [193]:

$$V_{FS} = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m}\left\|\boldsymbol{x_j} - \boldsymbol{v_i}\right\|^2 - \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m}\|\boldsymbol{v_i} - \overline{\boldsymbol{v}}\|^2$$

(3.26)

The Fuzzy Hyper volume Index [194]:

$$V_{FHV} = \sum_{i=1}^{c}[\det(F_i)]^{1/2}$$

(3.27)

$$F_i = \frac{\sum_{k=1}^{N} u_{ik}^{M}(\boldsymbol{x_k} - \boldsymbol{v_i})(\boldsymbol{x_k} - \boldsymbol{v_i})^T}{\sum_{k=1}^{N} u_{ik}^{m}}$$

(3.28)

Where all symbols are consistent with those used in Section 3.1. Each of these indices incorporates the input data into the calculation in addition to the partition matrix and cluster prototypes. More recent developments are increasingly complex, and examine factors such as cluster separation, overlap, and compactness in attempts to improve the overall applicability of the indices. Examples of more recent indices can be found in [195] and [94].

3.4.2.6 Non-numeric rule quality assessment

While accuracy provides a good indication of a model's predictive ability, there are other less quantifiable aspects of model quality which are also important. Consider the field of human-centric computing, which targets computational models whose structure and predictions are presented in a way which can be reasonably assessed and understood by a human agent. In this context, the absolute accuracy of a model is not the epitome of model quality, as very complex black-box models completely fail to provide useful decision-making information to an outside observer, decreasing the value of any extracted knowledge.

In the interest of more formally defining what we mean by human-centric systems, we define additional quality evaluation criteria such as *coverage, completeness, distinguishability and complementarity*. In [102] these are identified as qualities associated with *low-level interpretability*, that is, aspects of a model which aim to capture specific aspects of quality associated with data representation. These terms aim to capture those aspects of data representation which adhere to human information categorization, maintaining model sanity from a reader's perspective. Consider the following definitions:

> *Distinguishability:* Fuzzy sets should be distinct within the input space such that each membership function is able to represent a clear linguistic term and have clear semantic meaning.
>
> *Coverage/Completeness:* The fuzzy sets should cover the entirety of the universe of discourse such that every input instance belongs to at least one rule.
>
> *Complementarity:* For a given input instance, the sum of its memberships to all rules should always be one.

Readability is another crucial aspect of human-centric modelling which focuses on the intrinsic need to understand how a model arrives at a given prediction. These aspects include *simplicity, readability, consistency, completeness, and transparency*. Again, referring to [102], these terms are defined as *high-level interpretability* criteria, synonymous with human readability:

> *Simplicity:* The best model will be the model which fits the data well, with the least amount of complexity. This can take the form of limiting the number of rules, lowering the order of a TS fuzzy model etc.
>
> *Readability:* A given rule should be understandable to a human, meaning that we need to limit the number of conditions to around a maximum of seven.
>
> *Consistency:* Rules in a system should not contradict one another.
>
> *Completeness:* Same as for low level interpretability, a rule should exist for every possible input instance.
>
> *Transparency:* The structure of fuzzy rules should embody human knowledge about a system's behavior. The consequents and conclusions of the rules should be meaningful to a human reader, and their meaning should be clear.

These criteria serve as the founding principles of model interpretability and, in this dissertation, we make use of them in our quantification of rule stability.

3.5 Boosting

Boosting is the general principle of constructing a group of models (ensemble) which, as a collective, outperform any single member of the group. The ensemble is constructed iteratively, with later models trained to address the weaknesses of previous models. The resulting collection makes predictions as a unit in a weighted manner, with each member's weight dependent on its individual performance.

The most well-known boosting implementation is AdaBoost, which was originally proposed in the mid 1990's by Freund and Schapire [24][25]. This algorithm is applicable to two-class classification problems only.

3.5.1 Bagging

Boosting algorithms seek to generate a set of diverse weak learners which, in combination, provide a single strong classifier. As many learners are being trained to address the same problem (as posed by data), all learners are generated from a limited dataset. Generally, each learner in an ensemble is trained using the same procedure – that is, excepting data weights, each learner is computed in the same way. This has the potential to be highly detrimental, as using the same data for all learners may decrease their diversity. A helpful addition, which aids in the avoidance of this problem, is data selection through bagging, and boosting commonly makes use of bagging for this reason.

Bagging is the process of selecting a subset of some size randomly from the training dataset, with the important caveat that repeat selections are permitted. This allows for training datasets of adequate size to be generated many times while still being distinct from one another. This promotes stronger overall ensemble training with a simple method.

As potentially relevant in a boosted environment where data is weighted, bagging can also be performed in a weighted manner, such that more heavily weighted instances are more likely to be selected. In the case of a weak learner generation procedure which poorly considers data weights, weighted bagging can be used to help achieve successful boosting without modification to the original algorithm.

3.5.2 AdaBoost

AdaBoost [24] is the most recognizable variant of boosting and it has been successfully applied to a wide range of classification problems. The AdaBoost algorithm is defined for a generalized weak learner and assumes a two-class problem with class labels taking the values of 1 or –1.

The goal of the algorithm is to compute an ensemble, denoted *F*, starting with an empty ensemble, denoted *F(0)*. We also initialize a set of data weights, one per training instances, stored in a vector *W*. Initially, each weight is set to the same value, $w_i = 1/N$, for $j = 1, 2, \ldots N$.

The algorithm proceeds iteratively, and at each iteration a new weak learner is trained from bagged data and the current data weights. The learner computed at iteration *t* is denoted as $h_t(\mathbf{x})$. The weak learner generation procedure is generic; that is, any classifier can be used as a component weak learner of a boosted ensemble, so long as the learner generation process is able to consider changing data weights in its calculations, or weighted bagging is employed.

Once a weak learner has been generated, its performance is evaluated according to the following formulation:

$$\varepsilon_t = \sum_{i=1}^{N} w_i E(h_t(x_i), y_i)$$

(3.29)

Where ε_t is the weighted error for the learner $h_t(x)$ and *E* is the given error function, computed in terms of the weak learner and the known output for instance *i*, y_i.

Consider the error function produced by the original AdaBoost algorithm:

$$E(h(x), y) = e^{-yh(x)}$$

(3.30)

AdaBoost is valid for a binary classification problem; hence, the class predictions take either a value of 1 or –1 indicating belongingness of the data point to one of the problems two classes.

The weighted error is then used in the calculation of the new learner's ensemble weight, $\propto_t$:

$$\propto_t = \frac{1}{2} ln\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right)$$

(3.31)

The existing ensemble from the previous iteration, *F(t–1),* is then updated to include the new learner obtained at iteration *t*:

$$F(t) = F(t-1) + \propto_t h_t$$

(3.32)

And finally, the data weights are updated with respect to the new learner and its ensemble weight:

$$w_{i,t+1} = w_{i,t} f\left(y_i, \propto_t, h_{t,i}\right)$$

$$f\left(y_i, \propto_t, h_{t,i}\right) = e^{-y_i \propto_t h_t(x_i)}$$

(3.33)

Where $w_{i,t}$ is the weight of instance *i* at iteration *t*, and *f* is an evaluation function for the error of a given input instance, for a given weak learner, with its given ensemble weight.

These steps are taken iteratively until an ensemble of the desired size is constructed, or some other termination criteria is met. It is imperative that the weak learner generation procedure takes the data weighting into consideration, as this is the mechanism through which overall ensemble performance is achieved.

When the time comes to use the ensemble to make classification predictions, the whole ensemble can be used to obtain a single prediction in the following manner:

$$F(x) = \sum_{t=1}^{m} \propto_t h_t(\boldsymbol{x})$$

(3.34)

Where $\boldsymbol{x}$ is a given instance, *F* is the completed ensemble and *m* is the size of the ensemble. The manner in which the weak learner is evaluated is specific to the model used; however, the expectation is that a single class prediction is provided.

3.5.2.1 AdaBoost M1 Variant

The AdaBoost M1 variant is a modification to the AdaBoost algorithm proposed by Freund and Schapire to extend their original algorithm to the multi-class space [25]. The M1 variation of AdaBoost makes the following simple changes.

First, the error rate of a weak learner is changed to be compatible with a non-binary classification:

$$p^t = \frac{w^t}{\sum_{i=1}^{N} w_i^t}$$

$$\varepsilon_t = \sum_{i=1}^{N} {p_i}^t [\![h_t(\boldsymbol{x_i}) \neq y_i]\!]$$

(3.35)

Where ⟦ . ⟧ describes an indicator function which returns 1 if the expression is true and 0 otherwise, and w^t is an instance weight at iteration t.

In the M1 variant, if the newest learner is weak, determined by $\varepsilon_t > ½$, then the learner is rejected, and the algorithm is stopped. Otherwise, the procedure continues with a few further modifications. In a similar manner to the update to the error calculation, the weight updates are also modified:

$$\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}$$

$$w_i^{t+1} = w_i^t \beta_i^{1 - [\![h_t(x_i) \neq y_i]\!]}$$

(3.36)

Finally, we consider the updated equation for obtaining a hypothesis from the boosted ensemble:

$$F(x) = argmax_{c \in C} \sum_{t=1}^{m} (log \frac{1}{\beta_t}) [\![h_t(\boldsymbol{x_i}) \neq y_i]\!]$$

(3.37)

Where C is the set of class labels in the problem space, and F is the completed ensemble comprised of m weak learners. This variation of AdaBoost for the multi class case is very simple, and many of the modified equations are intuitive extensions of the binary case with the updated computation of a multi-class error rate.

3.5.3 SAMME

As the original AdaBoost algorithm is only applicable to two-class problems, many studies have proposed variants for the extension of the boosting mechanism to a multi-class scenario (including the M1 variant previously described). Another of these variants is the SAMME algorithm (Stage-wise Additive Modeling using a Multi-class Exponential loss function) [137]. This algorithm operates on the same general principles as AdaBoost, but with specific changes to certain formulations to accommodate a multi-class environment.

The first aspect which needs modification is the error function, previously given in (3.29). The AdaBoost variant functions with the assumption of binary output (–1, or 1); hence, for a multi-class scenario where class assignments are simply distinct labels, modifications must be made. Consider the formula:

$$\varepsilon_t = \frac{\sum_{i=1}^{N} w_i \neg I(c_i, h_t(\boldsymbol{x_i}))}{\sum_{i=1}^{N} w_i}$$

(3.38)

Where *I* is an indicator function, namely:

$$I(a,b) = \begin{cases} 1 & if\ a = b \\ 0 & otherwise \end{cases}$$

(3.39)

And its negation is described as:

$$\neg I(a,b) = \begin{cases} 0 & if\ a = b \\ 1 & otherwise \end{cases}$$

(3.40)

The result of (3.38) is a weighted error with respect to training data weights. This is not strictly equivalent to the AdaBoost formulation but serves a similar purpose.

Secondly, the ensemble weight calculation needs modification. In AdaBoost, the computed weights become negative if the accuracy of the weak learner is worse than ½. This is reasonable as we should reject learners whose performance is worse than a random guess; however, this criterion changes for a multi-class scenario, where the threshold for a better than random guess becomes 1/*K* where *K* is the number of classes. Consider the multi-class modification:

$$\propto_t = log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) + log(K-1)$$

(3.41)

Finally, the equation for updating data weights is modified. In a similar manner to the error equation, the formulation is updated to consider *correct* or *incorrect* class predictions.

Consider:

$$w_i = w_i exp(\propto_t \neg I(c_i, h_t(\boldsymbol{x_i})))$$

(3.42)

Which modifies the weight update calculation to similarly match the error calculation in (3.38).

Other than these changes, the SAMME algorithm functions identically to AdaBoost – the algorithm proceeds by iteratively calculating a generalized weak learner, assessing said learner, and updating algorithm data on this basis.

3.6 Gradient Descent

Gradient descent is a well-known iterative optimization tool which uses knowledge of a problems derivative function (or an approximation thereof) to search for a local minimum by following the direction of the negative gradient.

Gradient descent functions by iteratively evaluating the gradient (derivative) of a function at the current search location and aiming the search path in the direction of the negative gradient. We begin the search at an arbitrary point in the search space and update the current search position according to:

$$\boldsymbol{x_{n+1}} = \boldsymbol{x_n} - \gamma \nabla f(\boldsymbol{x_n})$$

(3.43)

Where ∇f is the first derivative of the function we wish to optimize, and γ is a predefined step size. We iteratively update the position of $\boldsymbol{x}$, moving in the direction of the negative gradient until either a local minimum is found ($\nabla f(\boldsymbol{x_n}) = 0$), or some other termination criterion is met, for example a maximum number of iterations.

In its simplest form, gradient descent is highly susceptible to local minima, so we may wish to make modifications to improve the likelihood of finding a global minimum. A common modification is the addition of a momentum term, which helps accelerate the search vectors in the right directions, speeding up optimizations, and helping avoid shallow local minima. In the simplest case, the addition of a momentum term takes the following form:

$$\Delta \boldsymbol{x_n} = \beta \Delta \boldsymbol{x_{n-1}} - \gamma \nabla f(\boldsymbol{x_n})$$

$$\boldsymbol{x_{n+1}} = \boldsymbol{x_n} - \Delta \boldsymbol{x_n}$$

(3.44)

Where $\Delta\boldsymbol{x_n}$ conceptually represents the velocity of the search, which is now updated with an added term, weighted by the parameter β, which remembers the velocity of the search at the previous iteration.

Gradient decent is an optimization tool which is appropriate for scenarios with known error functions. When configured correctly, it is a fast and reliable optimization technique which is used extensively in many fields.

4 CRITERIA FOR FUZZY RULE STABILITY

Fuzzy rule-based models are a pillar of fuzzy modelling which provide a concise, human-readable, and computationally powerful way to express complex system knowledge. The most established and well-studied form of fuzzy rules are Takagi-Sugeno style rules, and their specifics are described in detail in Section 3.4. Typically, the focus of studies concerned with fuzzy rules tends towards the improvement of model accuracy, while other aspects, such as interpretability, readability, and comprehensibility, fall by the wayside.

This is not to imply that these softer aspects of model quality have been completely ignored. Section 3.4.2.6 provides examples of literature focusing on definitions concerned with aspects of interpretability; however, the properties discussed in these studies lack concrete quantifiable meaning and measurement. In other words, while these definitions are useful for academic discussion of rule quality, we lack any calculable metrics to quantify these concepts, limiting their real-world usefulness.

In this chapter, we explore the novel concept of providing a quantitative consistent measure targeting certain aspects of rule quality; specifically, we focus on the quantification of rule stability. For our purposes, we define rule stability as the ability of a rule generation procedure to consistently generate *similar* rules for *similar* training data. This includes the extraction stable knowledge, even when there are small fluctuations in the training data. This is a desirable characteristic, as we expect rules to convey essential knowledge about the system, and fluctuations in initialization conditions and training data should not affect what we hope is concrete knowledge. Put another way, if the knowledge extracted from data highlights meaningful system behavior, we would expect this knowledge to be reproducible regardless of the exact training data used.

Formally, consider the following three quantifiable aspects rule stability:

> *Multiplicity of rules* – Rules are produced with identical conditions and conclusions. Identical rules appearing consistently in experimentation is an indicator of stable knowledge.
>
> *Conflicting rules* – Rules are identified with identical condition parts, but different conclusions. Such rules are brittle, as they do not provide a coherent prediction of system behavior, indicating lower quality rules.
>
> *Generalizable rules* – Rules whose condition parts are slightly different; however, their conclusions are the same. These can be combined into a single rule whose condition part becomes more general; hence, the quality of the rule itself becomes higher.

These metrics outline what we consider to be the crucial aspects of fuzzy rule stability, and, in this chapter, we propose a methodology for the computation of numerical metrics quantifying each aspect independently.

In this chapter, we define model stability as the production of similar rules given similar data and input conditions. That is, for a given problem, when using consistent parameters and algorithms, we expect consistent models to be generated. Furthermore, if the quality of the rules produced by these models is high (they describe the data well), we would expect that similar rules should be produced from a random subset of the data, given that the subset is of sufficient size.

The essential issue in this study is to provide a comprehensive characterization of certain qualitative (non-numeric) properties of FRBSs. When dealing with a fuzzy model constructed from data, how do we define the multiplicity, generalizability and conflicting properties of the model? The objective of this research is to define these properties in a numeric manner and to analyze their use for assessing the quality of rules extracted from data.

Our stability criteria seek to deliver an essential qualitative view of rule quality. To achieve this, we abstract away from the numeric details of individual rules (such as those captured by membership functions), and focus instead on rules as pieces of knowledge, providing insight into the brittleness and volatility of rules. To this end, we pose the following pertinent research questions:

- To what degree are the rules stable, and to what extent do they offer consistent knowledge about the system? Are they resilient to small changes in data?
- Can rules be generalized, and in what manner?

To assess the stability of fuzzy rules, we devise a strategy for generating a set of fuzzy rule-based models from subsets of the training data to enable rule comparisons. Consider a family of fuzzy models, each with the same number of fuzzy rules. We divide the training data into equally sized subsets and generate a fuzzy rule-based model from each subset using identical parameterization. The fuzzy rules generated in this manner are then analyzed *en mass* to assess the consistency with which certain information is extracted from the data given varying training data and initial conditions (but identical parameterizations). Our method of analysis is to construct rules using identical design parameters using subsets of the data $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, …, $\boldsymbol{D}_p$ (which are not identical to $\boldsymbol{D}$ but share the same or very similar statistical properties as those of $\boldsymbol{D}$). We then assess the properties outlined above.

Our approach realizes an additional higher-level layer of knowledge representation. This higher-level is used to abstract away from numerical details without modifying the original model, providing a linguistic space in which stability analysis can be more easily performed, visualized in Figure 4.1.

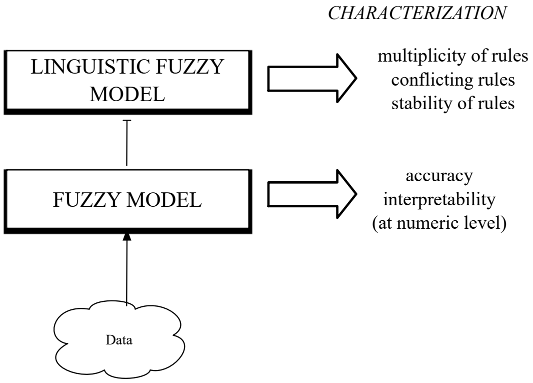


Figure 4.1: Fuzzy models and an augmentation of its characterization at the linguistic level of information granules

4.1 INTERPRETATION OF RULES - A SYMBOLIC VIEW

In this study, we consider Takagi-Sugeno style fuzzy rules. To assess rule quality, we need a way to compare rules from independently generated models (data sub-sets) in a simple manner. Fuzzy rules are meant to provide generalized knowledge about a system and, as such, we consider them to be a high-level representations of system knowledge. When comparing fuzzy rules across multiple rule-based models, it is this high-level structure that we would like to compare. Expanding on this concept, when comparing rules, we want to abstract the comparison away from the numerical details of the model and focus on the high-level structure of each rule. This implies a linguistic rule format, where rule conditions and conclusions are represented as linguistic quantifiers, granularizing each feature for the sake of interpretability.

To accomplish this, we propose the following “unfolding” procedure which functions by projecting the rule prototypes onto each input variable to form pseudo-linguistic labelling. An example of such a projection and subsequent labelling is shown visually in Figure 4.2.

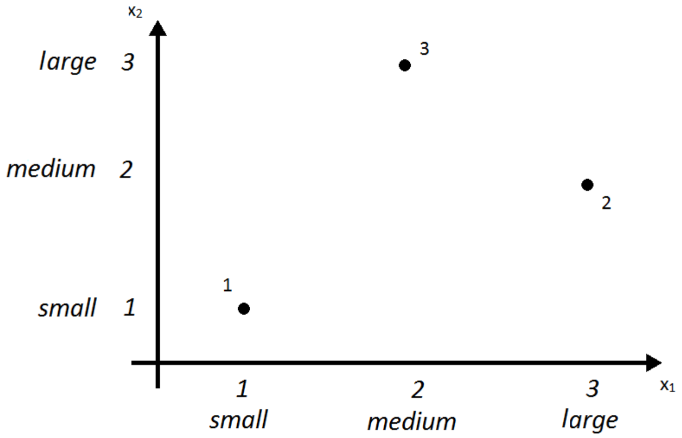


Figure 4.2: Unfolding rules with the use of Cartesian products by projecting prototypes on individual input variables.

By performing this granulation on each FRBS, we are left with a rule format which is more conducive to high level rule comparisons. This format will be essential to our analysis of rule stability, where the ability to abstract away from minor rule variability and concentrate on overarching rule structure is critical. By assigning a linear integer ordering over each input dimension of the rule, we create a granular rule format composed of pseudo-linguistic variables. In the resulting rules, we intuitively understand that linguistic variables are implicitly assigned in increasing order for each input dimension. We represent the result of this process in an integer format, which generalizes fuzzy rules to a symbolic rule space.

For instance, referring to Figure 4.2, the three rules expressed in the defined symbolic format would read as follows:

$$(1,1) \rightarrow 1; (2,3) \rightarrow 3, (3,2) \rightarrow 2$$

$$(small, small) \rightarrow small; (medium, large) \rightarrow large, (large, medium) \rightarrow medium \tag{4.1}$$

Considering both the integer and implicit linguistic assignments for each rule.

This format will be used as the basis for rule comparison in upcoming sections, where the quantitative aspects of rule quality and stability are more formally defined.

In this simplified format, the distance between two rules can easily be computed as a rectilinear distance:

$$\|a - b\| = \sum_{j=1}^{n} |a_j - b_j| \tag{4.2}$$

Where *a* and *b* are two integer format rules, each defined for *n* input features. Generally, we consider the distance between two rules to consider only the input dimensions, addressing the consequent parts of the rules separately in future sections.

4.2 MEASURES OF RULE STABILITY

As outlined previously, we consider rules extracted from data to provide a high-level representation of the patterns expressed by the data – it's contained knowledge. We intuitively expect that, to some degree, the rules produced in a high-quality model will show a degree of stability if the same procedure is executed again. Put another way, we expect the variance of rules generated in the same way using the same (or similar) data to be small. Further, if the rules represent stable system knowledge, then small fluctuations or variations in the training data should not significantly impact the computed rules.

These metrics aim to quantify the degree of stability by using the granulation strategy described in Section 4.1. The granulation procedure maps the rules into a symbolic space, where we can easily compare rules based on their high-level linguistic interpretations. This is useful both because it makes the comparison of fuzzy rules simpler, and because variations in training data, which may cause rules to shift, are effectively removed through this process, revealing the relevant structure.

The goal of this research is to assess the rule stability of a system. Consider an environment in which we have some dataset ***D***, which is split into a number of equally sized subsets, $\boldsymbol{D_1}$ ... $\boldsymbol{D_p}$. Each subset is used as the training data for a FRBS using identical input parameters. This results in a set of *p* FRBS's. This collection, or family, of fuzzy models can then be granularized using the procedure in Section 4.1 and analyzed *en mass*.

Each of the following three sub sections describes one of three stability metrics: *multiplicity, generalization,* and *conflict*.

4.2.1 Multiplicity

The goal of the multiplicity metric is to evaluate the degree to which rules repeat within a family of fuzzy models. If a given rule is consistently generated, it is likely that it expresses meaningful system knowledge and is hence of high quality. Given a collection of *p* FRBS's, each containing *c* rules, an analyzed family of fuzzy models contains a total of p*c fuzzy rules. By searching through this collection for the number of times a specific rule appears, we determine a multiplicity score for that rule. The possible range of a multiplicity score is therefore 1 (the rule occurs a single time) to p (the rule occurs in every model). A higher multiplicity score indicates a more stable (readily producible) rule. Formally, we define multiplicity as the following; given the set of all rules, $\boldsymbol{S}$**,** the multiplicity score for some rule i is given by:

$$M_i = |\boldsymbol{B_i}|$$

$$\boldsymbol{B_i} = \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{i}\| = 0, \forall \boldsymbol{x} \in \boldsymbol{S}, i \in \boldsymbol{S}\}$$

(4.3)

For *i = 1, 2, … r* where *r* is the cardinality of $\boldsymbol{S}$. In the computation of (4.3), | · | denotes the length of a set, and || · || denotes the distance between two rules, given in (4.2). The result of this computation, $\boldsymbol{B_i}$, is the set of all matching rules, and the multiplicity of the *ith* rule is given by the length of this set.

Defining multiplicity on a per rule basis is a useful; however, to assess the quality of modelling more generally, we need to define a more generalized measure. To achieve this, we define a parameter independent variation, *Θ***.** This metric is defined by dividing the system's total multiplicity score (*M*) by the maximum possible multiplicity score for that system.

First, we calculate the total multiplicity score for a system as:

$$M = \sum_{i=1}^{r} (M_i - 1)$$

(4.4)

Where M_i is the multiplicity for the *ith* rule, computed over all *r* unique rules. For the purposes of these calculations we consider multiplicity to be calculated once for each unique granularized rule in the family of FRBSs. To assess the desired parameter independent variant, we then normalize *M* by the maximum possible score:

$$\Theta = \frac{M}{c * (p - 1)}$$

(4.5)

We subtract one from each rule's multiplicity such that rules which only occur once do not contribute to the generalized multiplicity score. This has the added benefit of providing scoring priority to singles rules occurring many times over many rules occurring a few times (e.g. A rule with some score q should be weighed higher than two rules with scores *q/2* in combination). In the normalization component (denominator) of this equation, the summed score is divided by the maximum possible multiplicity score given the previous stipulations such that a value of $\Theta = 0$ results from no repeating rules, and a value of $\Theta = 1$ results from identical rule sets in every model.

4.2.2 Generalization

Generalization is a complementary metric to multiplicity which aims to quantify the degree of similar but not identical rules in a family of FRBSs. In this instance, we are interested in similarities in the condition part of the rules, and we require matching consequents for two rules to generalize. Two rules are defined as generalizable if their rectilinear distance is smaller than some threshold *e*:

$$\rho(\boldsymbol{I},\boldsymbol{I}') = |i-i'| + |j-j'| + |k-k'| + \ldots + |l-l'| \tag{4.6}$$

For two rules ***I*** and ***I’*** defined as:

$$\boldsymbol{I}: (i,j,k,\ldots l) \rightarrow z \; and \; \boldsymbol{I}': (i',j',k',\ldots l') \rightarrow z \tag{4.7}$$

Note that the distance evaluated in these calculations excludes rule consequents, which *must* be equal for a potential generalization. From these definitions, we state that two rules generalize each other if $\rho(\boldsymbol{I}, \boldsymbol{I}') \leq e$, where *e* is the generalization threshold. In the simplest case, we take a threshold of *e = 1*, meaning that we allow a distance of one granular input value between the two rules. Considering the granular format, if integer granulations are not consecutive, then the distance between two rules is greater than 1. For *e = 1*, a generalized rule takes the following form:

$$(i \; or \; i',j,k,\ldots l) \longrightarrow z \tag{4.8}$$

Where *i* and *i’* differ at most by *e*.

Generalized rules are of higher quality, as they convey a greater amount of system knowledge – the single rule applies to a larger amount of the input domain without change in the predicted value.

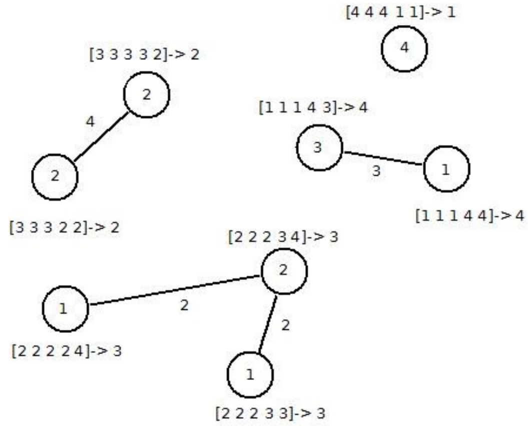


Figure 4.3: Generalization visualization as connected rules. Nodes represent individual rules, with the contained values being multiplicity. Edge values indicate generalization

To compute a generalization score for a family of FRBS's, we consider the total number of generalized rule pairs. The previously defined multiplicity measure needs to be considered in this calculation, as repeating rules form generalized pairs in combination. Consider the visualization shown in Figure 4.3. In this graph, each rule is represented by a circular node, with multiplicity values given as the internal node label. Rules connected by an edge generalize one another (e = 1), with the generalization score indicated by the edge value. Visually, the generalization score of this family of fuzzy rules is given by the sum of all edge weights. This graphical format is a useful tool for visualizing the degree of multiplicity and generalization, as well as visualizing how generalized pairs relate to one another in the granularized input space.

As with multiplicity, we define a parameter independent score for generalization, K. Recall from (4.3) the set $\boldsymbol{S}$:

$$K = \frac{/\boldsymbol{G}/}{c*p}$$

$$\mathbf{G} = \left\{(\mathbf{x}, \mathbf{y}) \mid \rho(\mathbf{x}, \mathbf{y}) = 1, \left\| z_x - z_y \right\| = 0, \forall \mathbf{x} \in \mathbf{S}, \forall \mathbf{y} \in \mathbf{S}\right\}$$

(4.9)

Where z_x and z_y are the consequents of rules $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively, | · | is a length operator, || · || is a distance function, and $\boldsymbol{G}$ is a set of rule pairs, notated as *(**x**, **y**)*, denoting two rules, $\boldsymbol{x}$ and $\boldsymbol{y}$**.** The function $\rho(\boldsymbol{x}, \boldsymbol{y})$ is the distance measure from (4.6).

4.2.3 Conflict

The last of our stability metrics measures the degree of conflict or disagreement within a family of fuzzy rules. We define two rules to conflict with one another if they have identical conditions, but different consequents. Unlike multiplicity and generalization, conflict is a generally undesirable feature of fuzzy rules, as it indicates inconsistency and ambiguity regarding the behavior of the system. Using a similar format to (4.7), we defined two conflicting rules, ***I*** and ***I'*** as:

$$\boldsymbol{I} = (a,b,c\ldots) \rightarrow z \text{ and } \boldsymbol{I}' = (a,b,c\ldots) \rightarrow z' \tag{4.10}$$

Where ***I*** and ***I'*** are in conflict if $z \neq z'$. Conflict leads to lower quality rules, as the response of the system is either less specific or contradictory. From the definition given in (4.10), we recognize that conflict occurs to some degree, depending on the absolute difference between *z* and *z'*. The granular values of *z* and *z'* may be similar, only differing by a small amount (e.g. 5 and 6 differing by 1), or the difference may be large (e.g. 2 and 10, differing by 8). While any degree of conflict indicates that the rules are of lower quality, it is important to recognize that the *degree* of conflict has important implications. A small degree of conflict is less concerning than larger discrepancies. With these considerations, we can formalize a degree of conflict between two rules:

$$\delta(\boldsymbol{I}, \boldsymbol{I}') = |z - z'| \tag{4.11}$$

Where we consider conflict only between rules with identical conditions.

Again, we establish an overall measure of conflict, as well as a parameter independent measure for use in comparative experiments. We define *Z* as the sum of conflict in a system, normalized by the parameters *c* and *p*:

$$Z = \frac{\sum_{i=1}^{r} \delta(\boldsymbol{F}_i)}{p * c}$$

$$\mathbf{F} = \left\{(\boldsymbol{x}, \boldsymbol{y}) \mid \rho(\boldsymbol{x}, \boldsymbol{y}) = 0, \left| z_x - z_y \right| > 0, \forall \boldsymbol{x} \in \boldsymbol{S}, \forall \boldsymbol{y} \in \boldsymbol{S} \right\} \tag{4.12}$$

Where ***F*** forms set of rule pairs of length *r*, notated as ***(x, y)***, denoting two rules, ***x*** and ***y,*** which are in conflict.

4.3 Case Studies

This section provides a set of experimental studies demonstrating how the proposed stability metrics can be applied to assess the quality of extracted fuzzy rules. In addition to experiments on conflict, generalization, and multiplicity, we also provide further analysis of the selected datasets to aid in experimental parameterizations.

4.3.1 Evaluation of the number of desired clusters

In this section, we perform experiments to obtain appropriate *c* values for later use. This is done through a combination of graphical analysis and cluster validity indices. In the full study [101], these experiments are performed using a larger number of datasets, but for the sake of brevity, we only include a sample in this dissertation.

First, we graph the performance index (*Q*) and output error (*E*) versus *c*, the formulations for which are given in (3.1) and (3.20) respectively. In this dissertation, we examine the AutoMPG and Abalone datasets, both available from the UCI machine learning repository [196].

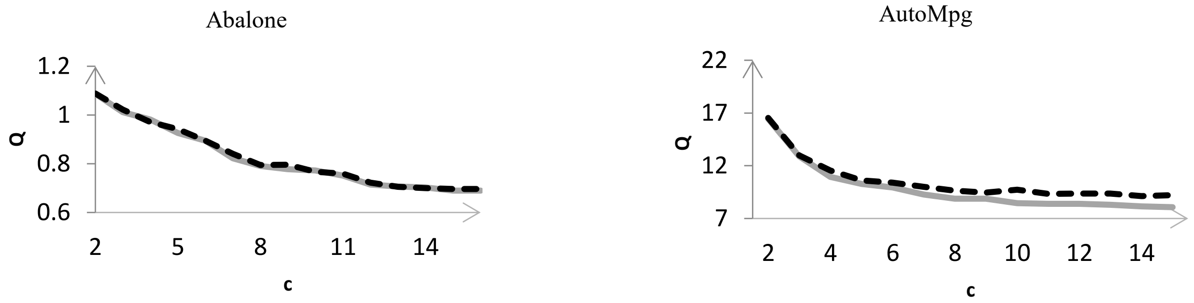


Figure 4.4: Performance index Q versus the number of rules c. Solid line refers to training data, dotted line concern test data

Figure 4.4 plots the performance index vs *c* for the two data sets under consideration. What we are looking for in these graphs is a clear pivot point, where the performance index ceases to decrease rapidly and starts to level out. In the AutoMPG plot, this point is quite visible at around 3-5 rules. The abalone graph is less obvious. We may postulate that a pivot exists at around 8 rules, or possibly that the leveling off occurs at 2 rules and the whole of the visible graph is the flatter part.

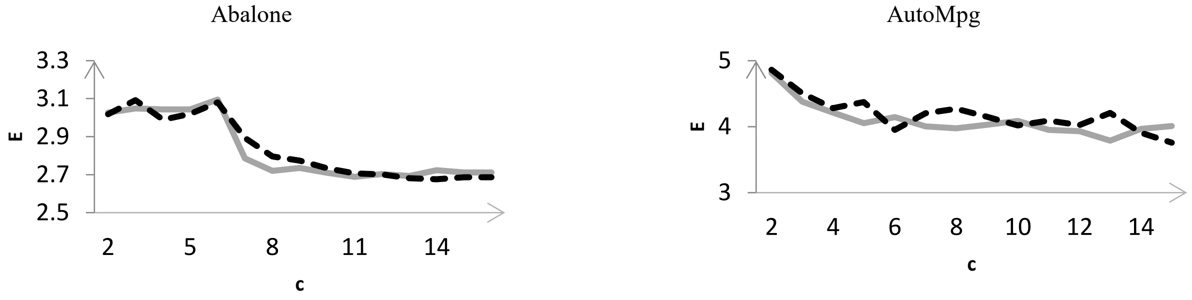


Figure 4.5: Output error E versus the number of rules c. Solid line is train data, dotted line is test data

The equivalent curves for output errors are shown in Figure 4.5. In these cases, the relationships between performance and the number of rules is less monotonically linear, so the choice is less obvious for both datasets. Abalone appears to drop somewhat dramatically at around 7 rules before clearly leveling off, and AutoMPG is perhaps best suited to 5 or 6 rules, although this data is unconvincing.

Another strategy for selecting *c* values are fuzzy validity indices, discussed in Section 3.4.2.5. For these experiments, we compute the partition coefficient (V_{pc}), partition entropy (V_{pe}), and the Xie-Beni index (V_{xb}).

Using the cluster validity indices and graphic approaches in combination we select a set of feasible *c* values for each dataset for use in upcoming experiments, given in Table 4.1.

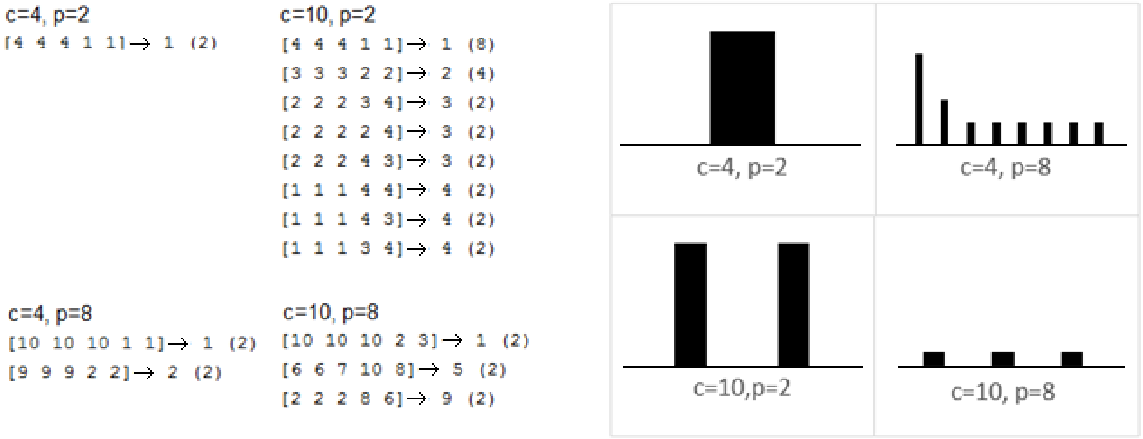
Table 4.1 Chosen c values from the analysis of performance index, error, and fuzzy cluster validity indices

Data set *c* choices						
Method	AutoMPG	Abalone	California	House	Ailerons	Pole
***Q* Graph**	3, 5	2, 8, 12	4, 7, 15	4, 6	3, 4	n/a
***E* Graph**	6	8	7, 10	2	n/a	3, 11
Vpc	2	2	2	2	2	2
Vpe	2	2	2	2	2	2
Vxb	3, 5	13, 15	5, 11	2	3	2
Other	10, 15	n/a	n/a	10	10	15
Final Choices	2, 3, 5, 6, 10, 15	2, 8, 12, 15	2, 4, 7, 10, 15	2, 4, 6, 10	2, 3, 4, 10	3, 11, 15

4.3.2 Case study for Stability Metrics

In this section, we discuss in detail the assessment and analysis of the proposed stability metrics. This section provides a comprehensive outline of the experimental processes used in later studies, and also serves to clearly demonstrate what is being quantified by each metric.

For this case study, we consider the AutoMPG dataset once again. The presented results use the following parameterizations in combination: *c = 4, 10* and *p = 2, 8*. These values are selected such that some interesting cases are exposed (small and large values for each parameter).



```
c=4, p=2
[4 4 4 1 1]→ 1 (2)

c=10, p=2
[4 4 4 1 1]→ 1 (8)
[3 3 3 2 2]→ 2 (4)
[2 2 2 3 4]→ 3 (2)
[2 2 2 2 4]→ 3 (2)
[2 2 2 4 3]→ 3 (2)
[1 1 1 4 4]→ 4 (2)
[1 1 1 4 3]→ 4 (2)
[1 1 1 3 4]→ 4 (2)

c=4, p=8
[10 10 10 1 1]→ 1 (2)
[9 9 9 2 2]→ 2 (2)

c=10, p=8
[10 10 10 2 3]→ 1 (2)
[6 6 7 10 8]→ 5 (2)
[2 2 2 8 6]→ 9 (2)
```

Figure 4.6: AutoMPG multiplicity, recurring rules and histograms for 4 parameterizations

Experiments assessing multiplicity are presented in Figure 4.6. On the left, repeating rules are given, with the value in brackets indicating their multiplicity (number of times repeated). This data translates directly into the histograms shown on the right. In histogram format, axes are omitted for compactness, and because exact values are not what concern us most. Note that only rules with multiplicity > 1 are reported, so in the case of c = 10 and p = 8, few cases are shown as there are only a small number of repeating rules. The purpose of this graphic is to visually demonstrate the multiplicity in a system. In all cases where this histogram format is given, the y-axis is scaled to the maximum possible value for each rule to have repeated, p. We use these histograms to quickly gain an overview of the multiplicity in a system, looking for repeating rules and their relative frequencies.

In Section 4.2.1, a system wide and parameter independent variation of multiplicity is defined, which aims to provide a quantitative assessment of the concept. These values, M and θ, are presented for this case study in Table 4.2.

Table 4.2: Multiplicity and Θ values for 4 parameterizations

	c = 4, p = 2	**c = 4, p = 8**
M	1	16
Θ	0.25	0.57
	c = 10, p = 2	**c = 10, p = 8**
M	2	3
Θ	0.20	0.04

Recall that higher values of multiplicity indicate a more stable rule base, as we can assert that the rules formed are more readily reproducible. In the example given, we identify the case of $c = 4$ and $p = 8$ to have produced the most stable rules, as indicated by the highest value of theta.

In Section 4.2.2, the definition of a generalization is given as the distance between two rules being less than some threshold. While this is intuitive to understand as a relationship between two rules, the way in which multiple rules may interact through generalizations, and the way in which generalization and multiplicity are related, may be less intuitively clear. To aid in the understanding of this concept, we present generalization graphs, which show how rules are connected within a family of FRBSs.

Figure 4.7 provides the generalization graphs for the four case study parameterizations, once again with respect to the AutoMPG dataset. In these graphs, a node represents a rule in the family of FRBSs, and edges indicate generalization between two rules. The value given inside a node indicates the multiplicity of that rule, and the value of the edge indicates the generalization value between connected nodes.

This graphical format provides additional information beyond scalar values, showing how the rules in a system relate to one another. We can see that, for certain parameterizations, the linkage amongst the rules is substantial, showing that although rules may not always be identical (multiplicity), very similar rules are regularly produced. Recognize that in the case of (a) and (c), the number of partitions is two, so the largest multiplicity (and likely generalization) is also two, and similarly for (b) and (d), but with $p = 8$. We identify that in many cases (a, b, c), rules form interconnected (generalized) clusters of p rules (visible for two cases in (a), two cases in (c) and two cases in (b) with one cluster having 7 connected rules). This is an indicator of relatively stable system knowledge as, even though the rules differ, they are similar enough to form generalized clusters. This type of insight is enabled through this graphical format and would not be possible given only numerical data.

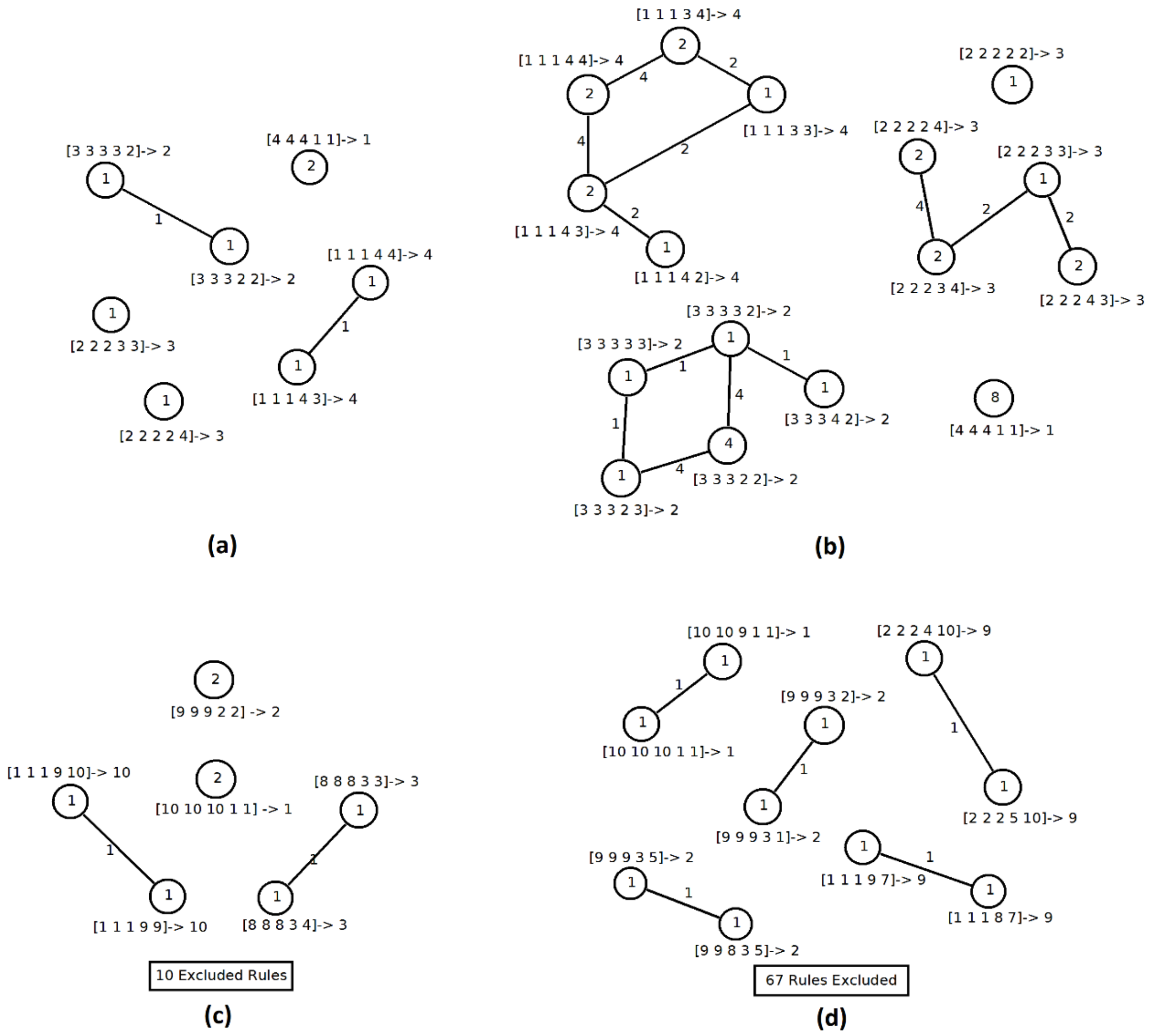


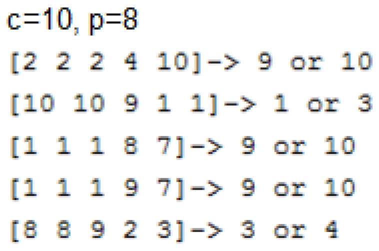
Figure 4.7: Generalization graphs: (a) c = 4, p = 2 (b) c = 4, p = 8 (c) c = 10, p = 2 (d) c = 10, p = 8. For clarity of visualization, rules which are not generalized have been excluded

Table 4.3: Generalization and K scores for 4 parameterizations

	***c* = 4, *p* = 2**	***c* = 4, *p* = 8**
|G|	2	20
K	0.25	0.63
	***c* = 10, *p* = 2**	***c* = 10, *p* = 8**
|G|	2	5
K	0.10	0.06

In addition to the generalization graph, we also compute the generalization scores for the systems; specifically, the overall score *G* (graphically in Figure 4.7 as the sum of all edge weights) and the parameter independent variant *K*, given in Table 4.3. Generalization scores are tricky to analyze on their own, as they are so strongly linked with multiplicity. Generally, we would say that a higher degree is better; however, more generalized rules is not desirable at the cost of repeating rules.

The final measure of fuzzy rule stability is conflict, defined in Section 4.2.3.



```
c=10, p=8
[2 2 2 4 10]-> 9 or 10
[10 10 9 1 1]-> 1 or 3
[1 1 1 8 7]-> 9 or 10
[1 1 1 9 7]-> 9 or 10
[8 8 9 2 3]-> 3 or 4
```

Figure 4.8: Conflicting rules from 4 parameterizations

Figure 4.8 shows the single parameterization from the four case studies in which conflict occurred (in the other three cases there were no conflicting rules). In this figure, conflict is shown through ambiguity in the consequent parts of the fuzzy rules. Recalling that conflict is computed to a degree, we identify that most conflicts in this case only occur to a degree of 1, with a single case where the degree is 2. As discussed in Section 4.2.3, small degrees of conflict can be understood as a generalization in the consequent of the fuzzy rules. While we observe some rule instability, these small degrees of conflict indicate slightly less specific system knowledge, as opposed to horrifically contradictory rules.

The total system conflict is then calculated, similarly to multiplicity and generalization, with a sum of conflict and the parameter independent variant given in Table 4.4.

Table 4.4: Conflict and Z scores for 4 parameterizations

	***c* = 4, *p* = 2**	***c* = 4, *p* = 8**
Σ(δ(F))	0	0
Z	0.00	0.00
	***c* = 10, *p* = 2**	***c* = 10, *p* = 8**
Σ(δ(F))	0	6
Z	0.00	0.075

As already mentioned, the conflict for 3 of the 4 cases was reported as zero, and we can see that in the case with non-zero conflict the resulting *Z* is very small.

4.3.3 Quantitative comparison of granular assignments

Our stability metrics are reliant on the integer-based granulation format defined in Section 4.1; however, to this point, we have not assessed the validity of this abstraction. The following set of experiments provides justification for the use of this methodology as a legitimate rule comparison methodology by examining the actual similarity of fuzzy sets which are considered equivalent in the granular format.

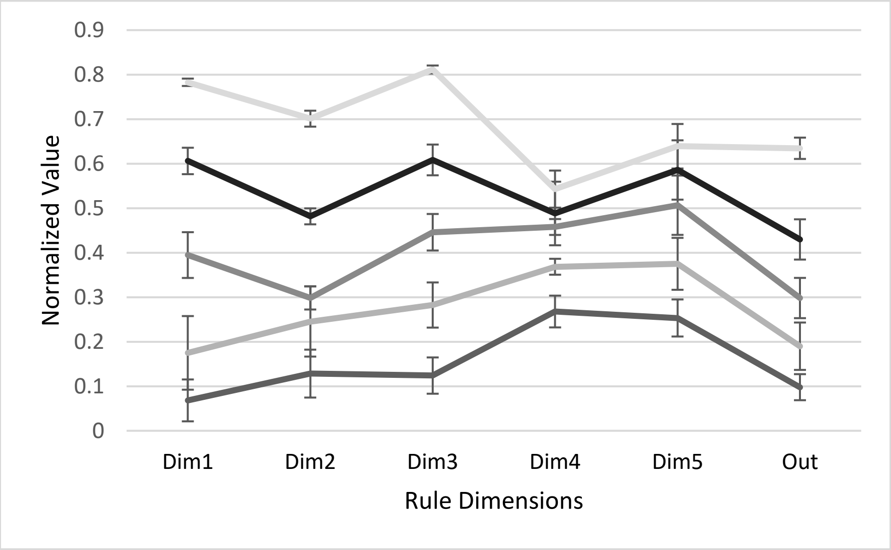


Figure 4.9: Granule assignment similarity for the AutoMPG dataset

The presented data is calculated by identifying the set of numeric values (cluster prototypes, rule outputs) associated with a given integer label for each feature and calculating the mean and standard deviation of each of these sets. The mean and spread are normalized to the unit interval to eliminate questions of scale within the dataset and to simplify visualization.

Figure 4.9 visualizes the results of this analysis for the AutoMPG dataset using $c = 5$ and $p = 8$. The x-axis categorically lists each feature in the problem space, and each series represents the mean values (with spread shown as error bars) for integer labels one through five, in increasing order bottom to top. This plot demonstrates that, in the vast majority of cases, we are justified in rule comparison using our integer format. This is demonstrated by the clear distinction between the mean values of each label and the minimal overlap between their deviations. A more thorough study, and further analysis and discussion, is provided in the full article [101], which applies this analysis to additional datasets. This type of experiment allows us to confidently proceed with our granular abstraction, having experimentally justified rule comparisons in this format.

4.4 STABILITY EXPERIMENTS

In this section, we provide a set of multiplicity histograms and a tabulated set of results for the parameter independent variations of each stability metric, given for a handful of real-world datasets. The full text of this study presents experiments for a more extensive number of datasets; however, for the purposes of this document we provide a reduced set considering the following datasets:

- AutoMPG
- California
- Pole

These datasets are all available from the UCI machine learning repository and provide a representative sample of the types of results presented in the full study.

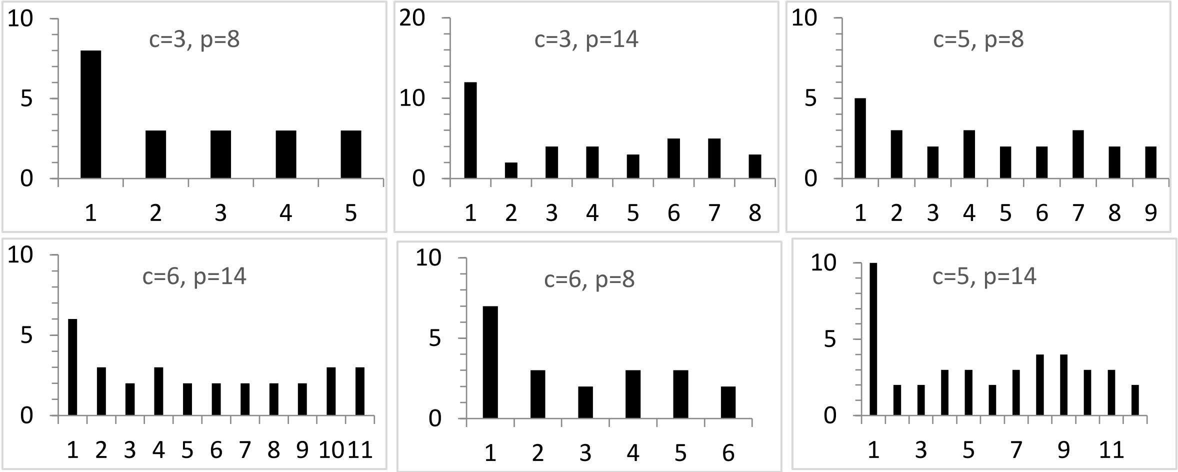


Figure 4.10: Multiplicity histogram for AutoMPG dataset

Table 4.5: Θ, Z, and K metrics for AutoMPG dataset

c	***p***	***Θ***	***K***	***Z***
2	**2**	1.00	0.00	0.00
2	**8**	0.86	0.88	0.00
2	**14**	0.92	0.93	0.00
3	**2**	0.33	0.00	0.00
3	**8**	0.71	1.00	0.00
3	**14**	0.77	2.48	0.00
5	**2**	0.40	0.20	0.00
5	**8**	0.43	0.83	0.05

5	**14**	0.45	1.21	0.06
6	**2**	0.33	0.00	0.00
6	**8**	0.33	0.56	0.04
6	**14**	0.24	0.95	0.07
10	**2**	0.00	0.20	0.00
10	**8**	0.06	0.09	0.00
10	**14**	0.02	0.11	0.05
15	**2**	0.00	0.00	0.00
15	**8**	0.01	0.00	0.03
15	**14**	0.01	0.03	0.00

Figure 4.10 and Table 4.5 present the first set of experimental results using the AutoMPG dataset. These experiments, as with upcoming experiments, are performed using the choices of *c* determined experimentally in Table 4.1 and $p = 2, 8, 14$ (representing a small, moderate, and large choice of *p*). In all experiments, only an interesting subset of multiplicity histograms are presented for brevity. In this case, Figure 4.10 shows that, in almost all experiments, the AutoMPG dataset produces one rule which is very common amongst partitions, indicating a very stable piece of system knowledge. This is the primary takeaway from the presented histograms, as otherwise we simply identify a significant number of rules which repeat with moderate frequency.

Moving our attention to Table 4.5, we can make clearer sense of the multiplicity histograms through combined knowledge of the overall system multiplicity as well as the degree of generalization and conflict. An initial point of interest is that when $\Theta = 1$, $K = 0$. This is by definition, as a perfect multiplicity score results in no generalizations; however, it is of interest to our analysis as this is the only case where high multiplicity in combination with low generalization should be considered a stable result. A further observation is concerned with cases where the number of partitions is low, e.g. $p = 2$. In these cases, the values of certain metrics are inconsistent with those reported using higher *p* values. This is likely a classic example of trying to generalize from a small sample size, and this is a limitation we must be cognizant of in our analysis.

Concentrating on individual scores, we note that, as *c* increases, the reported values of Θ and *K* decrease rapidly. This is generally an indication of decreasing rule stability, which is further supported by some cases of conflict beginning to arise. Overall, for values of *c* in the range of 2-6 the metrics indicate that rule formation is a stable process, which implies that the knowledge extracted from the system is representative.

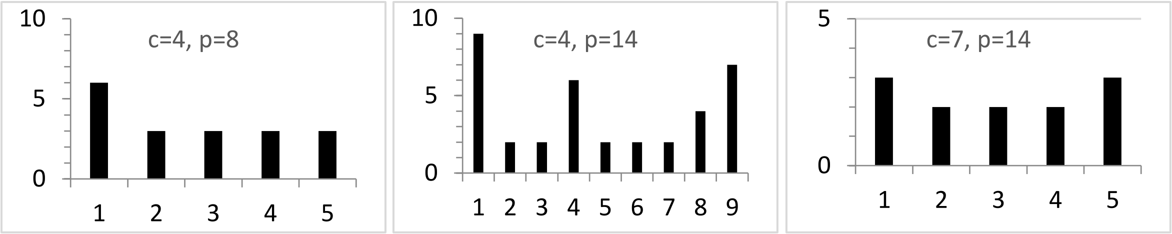


Figure 4.11: Multiplicity histogram for California dataset

Table 4.6: Θ, Z, and K metrics for California dataset

c	**p**	**Θ**	**K**	**Z**
2	2	1.00	0.00	0.00
2	8	1.00	0.00	0.00
2	14	0.85	0.93	0.00
4	2	0.50	0.25	0.00
4	8	0.46	0.59	0.25
4	14	0.52	0.79	0.29
7	2	0.29	0.07	0.00
7	8	0.12	0.09	0.41
7	14	0.08	0.13	0.59
10	2	0.20	0.00	0.00
10	8	0.03	0.03	0.28
10	14	0.03	0.06	0.14
15	2	0.00	0.03	0.00
15	8	0.01	0.01	0.08
15	14	0.01	0.02	0.00

The California data set, results given in Figure 4.11 and Table 4.6, provides an additional example of stable rule formation as identified by our analysis. Many of the same patterns are visible for this dataset as for the AutoMPG dataset; however, we note that higher conflict is visible much more quickly in this instance. Once again, the apparent rule stability decreases as the number of rules per partition increases, at a similar rate to those seen using AutoMPG. This dataset provides a good example to highlight why we must assess multiplicity and generalization as intertwined metrics, demonstrated for *c* = 4. While the experiments record multiplicity scores of only around 0.5, the generalization scores are high, and we should interpret the stability of these rules more positively as a result of this combination.

Table 4.6 also provides an opportunity to discuss the nature of conflict in rule stability. In the case study in Section 4.3.2, we presented some example conflict for a system, and we noted that conflict occurs to a degree. This means that, in cases where the degree of conflict is small, we may be seeing conflict manifested as rule generalization in the output space of the problem. This type of conflict is still undesirable, as it implies weaker predictive power of the resulting model; however, it is important to understand that small degrees of conflict should not cause undue alarm in stability analysis.

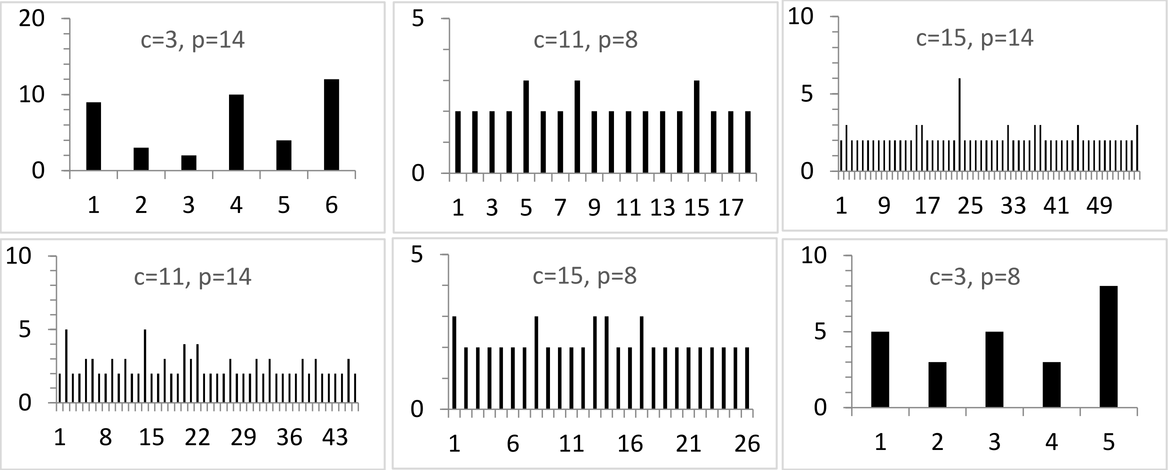


Figure 4.12: Multiplicity histogram for Pole dataset

Table 4.7: Θ, Z, and K metrics for Pole dataset

c	p	Θ	K	Z
3	2	0.33	0.00	0.33
3	8	0.90	0.00	0.08
3	14	0.87	0.00	0.24
11	2	0.09	0.00	1.82
11	8	0.27	0.00	7.23
11	14	0.48	0.00	7.71
15	2	0.00	0.00	2.07
15	8	0.30	0.00	8.57
15	14	0.35	0.02	14.39

Our final in-depth analysis considers the Pole dataset, results given in Table 4.7 and Figure 4.12. This result is of an entirely different nature to the two previously analyzed experiments and demonstrates the behavior of the proposed stability metrics when the modelling process does not go well. To begin, notice that the multiplicity histograms in Figure 4.12 contain cases where there are many rules that repeat more than once, but with very few rules occur a large number of times, with the phenomenon being more pronounced as *c* increases. Overall, these histograms appear chaotic and do not show trends towards stable rules. Next, we examine Table 4.7. Notice that generalizations scores are very small across the board. This, when coupled with higher multiplicity, is a red flag, indicating that even those rules generated repeatedly are likely not describing meaningful system knowledge. Furthermore, a simple glance at the conflict column tells us that a huge number of conflicting rules are being generated and that, at least in some cases, the degree of conflict is large. This is an additional indication of unstable rule formation, and the consistency of this type of result across all parameterizations speaks strongly to the instability of the system.

4.5 Further Analysis

In this section, we provide focused analysis on each of the proposed stability metrics and the parameter *p*, discussing in general terms how these metrics can be used, what we have observed experimentally, and what further steps for their application and improvement may be available.

4.5.1 The parameter *p*

The effect of the parameter *p* was not explicitly examined in this study. In our experimentation, we deliberately picked a small, moderate, and large value of *p* in our experiments, to cover our options without making the experimental results too long. From our observations, we have identified two key factors in the choice of *p*. 1) If *p* is too small then it is difficult to draw meaningful conclusions from our stability metrics, as the sample size is simply too small, and the results may not be representative. 2) If *p* is too large we may segment the dataset into partitions containing too few instances to form high quality models, invalidating any stability analysis performed. These observations are at odds with one another, meaning that the choice of *p* will always be dataset dependent, as a smaller dataset will require more care with respect to point 2, but a larger data set may not have any such concerns. As such, our only conclusion can be that the choice of *p* should be made intelligently by the researcher, finding a reasonable balance between the size of the dataset and the desire for more robust stability analysis.

4.5.2 Multiplicity

For evaluating the stability of a FRBS, Θ indicates how consistently similar, or nearly identical, rules are produced. Multiplicity provides a straightforward quantification of how often the same rules are produced from data, with consistently extracted knowledge implying higher quality rules. As shown in our experiments, the parameter p can have a significant impact on the ability to meaningfully assess multiplicity, as if it is chosen to be small the total number of rules under analysis is also small. Generally, we consider a higher value of Θ to indicate more stable rules, and this metric is the most straightforward in its analysis.

4.5.3 Generalization

The value of K is closely linked to that of Θ. This is intuitive and desirable, as K is designed to capture a degree of similarly amongst non-identical rules. The relationship between multiplicity and generalization is clearly illustrated in the generalization graphs, and these two metrics in combination provide the overall indicator of stability. For the most part, ignoring edge cases, we expect Θ and K to vary together, with higher Θ scores being accompanied by higher K scores. The absence of one of these scores in the presence of the other is a red flag indicating that something is amiss, as demonstrated experimentally by the Pole dataset. On its own, generalization does not provide definitive stability knowledge, and its analytical power comes in combination with multiplicity.

4.5.4 Conflict

A system exhibiting a significant degree of conflict is intuitively understood to be one of low stability, as conflict indicates uncertainty is system behavior, manifested through inconsistent predictions for certain inputs. This makes conflict a useful measure of instability, and it can be used independently to show the likelihood of poor predictive quality (implied by inconsistent predictions), or of inconsistencies in the modelling process resulting in vastly different models from similar data. Conflict is also useful in combination with the other two metrics, as it can help confirm conclusions drawn from multiplicity and generalization. We note that small amounts of conflict are acceptable, especially when considering large systems. A small degree of conflict is a form of generalization in the output parts of the rules, and while this decreases rule specificity, these slightly lower quality rules are not intrinsically unstable, and some (small) degree of conflict should be expected when analyzing large sets of fuzzy rules.

4.6 CONCLUDING REMARKS

This chapter proposed three metrics for assessing the stability of fuzzy rules extracted from data. The goal of these metrics is to provide an indication of rule quality through the lens of rule stability, assessed via their reproducibility from a subset of data. Our stability metrics can be applied in combination to achieve a comprehensive view of model quality, with each metric assessing a different aspect of rule stability. Multiplicity and generalization quantify the degree of repeating rules; multiplicity through exact rule matching, and generalizations through the identification of similar rules. These metrics are closely linked, and we wish to maximize their assessed values. Providing a contrasting view, conflict quantifies the undesirable aspect of disagreeing or inconsistent rules, and we seek to minimize the degree of conflict in order to maximize rule quality.

The stability metrics in this study are computed through the division of data into some number of subsets, such that the rules produced from each subset can by analyzed *en mass*. To simplify the task of rule comparison we have defined a simple granulation method through which rules are transformed from their original form into a higher-level granular format. This allows us to clearly define the formulations of our stability metrics and abstract away from any slight rule variances caused by fluctuating data. We propose this rule granulation as a form of linguistic interpretation, moving rules away from their numeric details and into a higher-level knowledge space. In many ways, rule comparison at this level of abstraction improves the quality of our analysis, as we compare rules in a human-centric linguistic space, which is more compatible with the concept of stable system knowledge.

We have applied the proposed metrics to assess rule stability for a selection of publicly available data sets and have shown experimentally how our metrics can be used in combination to gain a complete understanding of rule stability for a given algorithm and dataset.

5 GRANULAR FUZZY RULE-BASED MODELS

Information granulation is a form of knowledge representation which attempts to the improve human readability and interpretability of complex data structures. In this chapter, we examine the use of information granules in the consequent parts of fuzzy rules, replacing the functional outputs of a standard TS-fuzzy model with information granules. From a human readability perspective, a polynomial rule consequent is a poor representation of the system knowledge, as it is difficult for a casual reader to grasp the overall behavior of the system at a glance. While we can appreciate that, in the context of a fuzzy rule, a functional consequent represents a complexly shaped fuzzy set, the limited information regarding the shape and size of this structure makes interpretation challenging. Information granulation can help resolve such issues by providing more information to the reader, while still maintaining a concise and accurate format.

In this chapter, we employ hierarchical clustering as an alternative to FCM clustering for the purposes of rule extraction. This methodology is not studied in the existing literature and provides an additional aspect of novelty to our study. The goals of the research presented in this chapter are therefore twofold. First, to explore the use of hierarchical clustering as a vehicle for fuzzy rule extraction from data, and secondly, to improve model interpretability by applying interval-base information granules to the consequent parts of fuzzy rules. As an additional related point of interest, we also assess the performance of hierarchical clustering as a rule generation procedure in comparison to FCM cluster and discuss strategies for the evaluation of granular fuzzy models.

This chapter contains significant novelty with respect to two primary topics. First, as mentioned, the use of hierarchical clustering for fuzzy rule extraction has not been studied in the existing literature, and our use of hierarchical clustering for this purpose is therefore novel. Further, the methodology for extracting fuzzy rules from data using hierarchical clustering contains certain additional aspects of novel work which arise from the use of this algorithm. Second, the application of interval-based information granules to the consequents of fuzzy rules has not been well-studied, and the extraction of these structures from data, as well as the successful evaluation of this form of fuzzy rule-based model, provides substantial further novelty.

We recognize the combination of information granules and FRBSs to be a logical progression in human-centric computing, given that a major advantage of fuzzy rule-based systems is their interpretability.

5.1 Fuzzy Rule Generation with Hierarchical Clustering

This section outlines a procedure for generating a FRBS from data using hierarchical clustering. This method is based on the computation of cluster prototypes, similar to those obtained from FCM, from a crisp data partition. The initial crisp data partition is formed using a cutoff clustering strategy from the hierarchical clustering dendrogram. For details on hierarchical clustering and cluster formation through a cutoff strategy refer to Section 3.2. The crisp partition is subsequently fuzzified using a standard fuzzy membership function and cluster prototypes are calculated from the partitions. Prototypes are computed as the mean position of each crisp partition:

$$\boldsymbol{v_i} = \frac{1}{N_i}\sum_{j=1}^{N_i}\boldsymbol{x_j} \tag{5.1}$$

Where the sum is taken over certain data points *j = 1, 2, ..., N*$_i$, where the considered data points belong to the crisp partition *i, i = 1, 2, ... c*, for a *c* cluster partition. Once we have computed the prototype locations, we fuzzify the partition using a standard fuzzy membership function:

$$A_i(\boldsymbol{x}) = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|\boldsymbol{x}\text{-}\boldsymbol{v_i}\|}{\|\boldsymbol{x}\text{-}\boldsymbol{v_j}\|}\right)^{1/(2\text{-}m)}} \tag{5.2}$$

Where $A_i(\boldsymbol{x})$ is the membership of point $\boldsymbol{x}$ to cluster *i*, *m* is a fuzzification coefficient, *m* > 1, and || . || is a distance measure.

At this juncture, we have computed identical data structures to those resulting from FCM clustering; notably, a set of cluster prototypes and a fuzzy partition matrix. From here, we follow a standard fuzzy rule generation procedure, assigning one rule condition per cluster prototype, resulting in Mamdani style fuzzy rules.

The prototypes produced by FCM and hierarchical clustering are different. As they are the essential structural components of the rule-based models, it is beneficial to quantify their similarity (proximity), to assess how the use of hierarchical clustering impacts rule locations. Consider $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, … $\boldsymbol{v}_c$ to be the prototypes generated by FCM. The prototypes obtained using some hierarchical linkage policy are denoted $\boldsymbol{f}_1, \boldsymbol{f}_2, .., \boldsymbol{f}_c$. We compute weighted Euclidean distances (the same as used in clustering) between $\boldsymbol{v}_i$ and $\boldsymbol{f}_j$, say r_{ij}, and organize the results as a matrix, $R = [r_{ij}]$. To assess the similarity among the prototypes, we determine the minimum entry in each row of R, $r_i = \min_{j=1,2,\ldots,c} r_{ij}$ and sum up these partial results, attaining an overall cluster similarity measure, denoted r:

$$r = \sum_{i=1}^{c} r_i \tag{5.3}$$

This index is an indicator of cluster similarity between any two clustering results. In this chapter, we use this measure to compare rules generated using FCM and hierarchical clustering.

5.2 Granularization of the Fuzzy Model

The use of hierarchical clustering in this chapter is only the secondary focus of the proposed work. The primary concern of this study is the topic of information granulation; specifically, the granulation of the consequent part of the fuzzy rules. In the following section, we define a strategy for the extension of Mamdani style fuzzy rules into a partially granular format. The motivation for the granularization of fuzzy rules is rooted in a desire to make fuzzy models more *realistic* in how they portray structures representing real-world data. In traditional Mamdani rules, rule consequents are represented by single values (constant functions). While the researcher is aware that there are underlying fuzzy sets beneath these simplistic representations, it is difficult, if not impossible, to accurately interpret many aspects of the underlying fuzzy structure, e.g. Support, core, etc. By forming an information granule in the consequent part of a rule, we provide an additional dimension to the knowledge presented by the model through a simple and easy to understand modification.

Consider a granular model whose output is given by:

$$Y = \sum_{i=1}^{c} A_i(\boldsymbol{x}) \otimes B_i \tag{5.4}$$

Where $B_i = [b_i^-, b_i^+]$ is an interval built around the traditional numeric consequent b_i, Y is an interval–valued output prediction, and ⊗ is an interval multiplication operator. The calculation of a single predicted output is given by a predicted interval:

$$[y^-, y^+] = \sum_{i=1}^{c} A_i(\boldsymbol{x})[b_i^-, b_i^+]$$

(5.5)

Taken over each rule, $i = 1, 2, \ldots c$.

The system described by these equations, (5.4) and (5.5), represents the desired granular format. Next, we define a methodology for the computation of these intervals. When forming information granules, we are concerned with the satisfaction of the two conflicting criteria of justifiable granularity; that is, it is desirable for the granule both to encapsulate as much of the data it represents as possible, while remaining as specific as possible.

Consider the total sum of fuzzy memberships belonging to a single fuzzy rule:

$$u_{i,total} = \sum_{k=1}^{N} u_{ik}$$

(5.6)

Due the formulation for fuzzy memberships, many of the memberships to a given rule are likely to be very small, as a side effect of membership summation to one. Consequently, we do not want to capture every point with non-zero membership in the information granule. Rather, we only want to capture those data points which meaningfully belong to a given rule. To accomplish this, we apply some threshold, $\rho \in [0,1]$, and we form an information granule defined as the smallest possible interval containing a total membership less than or equal to ρ*u_{i_total}. This granule (interval) simultaneously targets both aspects of justifiable granularity. By making the interval as small as possible, we target the maximization of granule specificity, and by including the bulk of the total membership (dependent on the choice of ρ), we justify the granule with as much relevant data as possible. Each interval is formed iteratively, by selecting in turn those data points with the highest membership to the relevant rule until the desired membership threshold is reached. Once the collection of relevant data is established, the interval is formed from the maximum and minimum output values in set, defined as $[b_i^-, b_i^+]$. This process is repeated for each fuzzy rule, resulting in a FRBS with interval valued consequents.

5.3 Evaluation of Granular Fuzzy Models

With a fully defined methodology for the extraction of a granular fuzzy rule-based model from data, we shift our attention to the evaluation of format. Traditional evaluation methods, such as RMSE, are incompatible with a granularized output; hence, new evaluation metrics are needed. In this section, we define two evaluation metrics encapsulating the two aspects of justifiable granularity, coverage and specificity. These criteria will be used in combination to assess the quality of the computed granular fuzzy model.

5.3.1 Coverage

When considering what constitutes an accurate response from a granular fuzzy system, it is intuitively desirable that the target value should be contained within the predicted interval. To quantify this concept, we define a Boolean test for interval accuracy. For any given instance, we test if the predicted interval contains the target value, and compute *coverage* as the percentage of predicted intervals passing this test:

$$coverage = \frac{1}{N}\sum_{k=1}^{N} \varphi_{Y_k(y)} \tag{5.7}$$

Where $\varphi_{Y_k(y)}$ is a characteristic function indicating containment within the interval Y_k. As this is a percentage calculation, coverage scores take values in the unit interval, [0,1], where 1 indicates that all predicted intervals contained their target values. At this juncture, we highlight that the coverage scores for a system varies with the parameter ρ. As ρ increases, so too do the size of the resulting rule intervals, and consequently the predicted intervals. Obviously, the larger the predicted interval, the more likely that interval is to contain the predicted value.

5.3.2 Specificity

Arbitrarily high coverage is easily attainable by predicting arbitrarily large intervals; however, the usefulness of such a model is highly questionable as the resulting intervals (granules) are too vague to provide meaningful insight. For this reason, coverage alone is not a sufficient judge of model quality. To counteract this shortcoming, we propose a conflicting evaluation metric which encapsulates the concept of specificity. Specificity describes how *specific* a prediction is, which in the context of a granular model incorporates the degree of predictive certainty. Specificity is linked to the size of the predicted intervals; the smaller the interval, the higher it's specificity. Specific intervals are desirable, as they describe system behavior with a greater degree of precision and help maintain the linguistic meaning of predicted granules.

Consider the following definition for the specificity of an interval, Y:

$$spec(Y) = 1 - \exp\left(-\frac{length(Y)}{|max - min|}\right)$$

(5.8)

In this equation, the length of a predicted interval is normalized by the cardinality of the input space, with *min* and *max* being the minimum and maximum values of the domain over which Y is defined. Applying this definition to the evaluation of a granular FRBS, we compute the average specificity of all predicted intervals:

$$spec = \frac{1}{N}\sum_{k=1}^{N}\left[1 - \exp\left(-\frac{length(Y_k)}{|max - min|}\right)\right]$$

(5.9)

This formulation is proposed such that more specific intervals result in a higher specificity score.

We have now defined two conflicting metrics for the evaluation of a granular fuzzy model, coverage and specificity. By using these conflicting metrics in combination, we have taken steps to avoid certain pitfalls of granular modelling, while providing a quantitative assessment of granular model quality. When assessing the quality of a granular model, it is desirable to maximize both metrics and, given their conflicting nature, to find the ideal balance between the two.

5.4 Experimental Studies

This section contains presents experimental studies using the granular fuzzy rule extraction methodology established previously in this chapter. We first examine the application of hierarchical clustering in detail and run some simple experiments to determine the best parameterization for this algorithm. Next, we assess the quality of fuzzy models generated by the proposed methodology using of the granular evaluation criteria defined in Section 5.3.

5.4.1 Linkage Assessment

In these initial experiments, we consider the AutoMPG dataset, available from the UCI machine learning repository, as a useful case study. The complete study [111] provides more comprehensive experiments, using a larger number of datasets to more convincingly assert our findings.

Models are constructed and evaluated using training and testing data partitions, with the training part containing 70% of the full dataset and the testing part containing the remaining 30%. Each experiment is repeated 10 times, using unique data partitions, to enable statistical analysis. We consider the fuzzification coefficient, *m,* and the number of fuzzy rules in a model, *c,* to be essential parameters whose values need to be assessed experimentally. When assessing the quality of a non-granular model, we consider RMSE to be the critical indicator of performance.

First, let us examine in detail the process of hierarchical clustering. Consider the following set of dendrograms produced by the three linkage policies defined in Section 3.2, *single linkage, average linkage,* and *complete linkage*, shown in Figure 5.1.

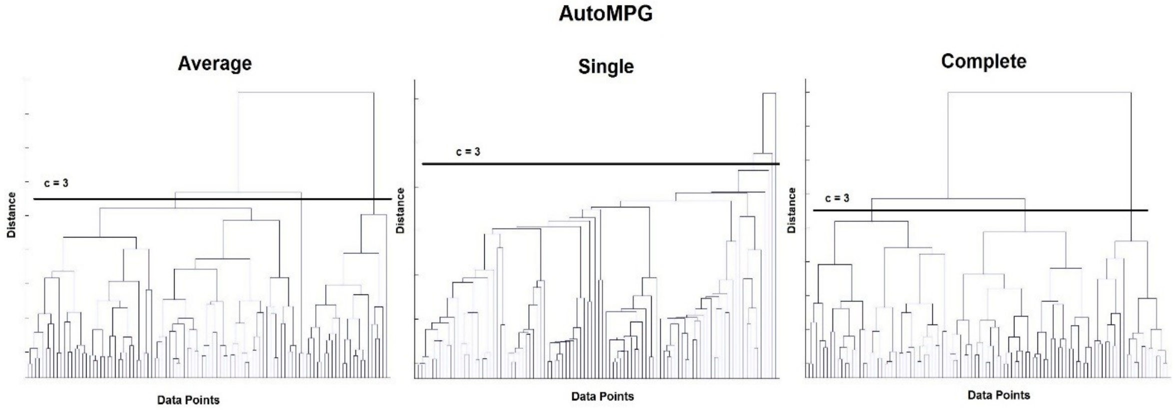


Figure 5.1: Dendrograms for average, single, and complete linkage, horizontal bar indicates level of clustering for c=3

The dendrograms presented in Figure 5.1 demonstrate the significant impact the choice of linkage policy has on the resulting clusters. Notice specifically that single linkage is very susceptible to the effects of outlier data and is likely to form unbalanced clusters. Conversely, complete and average linkage form a more balanced cluster structure, more similar those formed by FCM.

We quantify these differences by computing their *closeness, r,* defined in (5.3), to assess the relative similarity between hierarchical linkage policies and FCM. The *r* values for a set of sample parameters are given in Table 5.1 and shown graphically in Figure 5.2.

Table 5.1: Closeness measure (r) for AutoMPG compared with FCM

AutoMPG			
c	**m**	**link**	**r**
3	2	average	6.97
3	2	single	18.51
3	2	complete	5.60
5	2	average	10.50

5	2	single	17.08
5	2	complete	10.68
8	2	average	6.75
8	2	single	13.71
8	2	complete	3.22

These experiments provide insight into the structures formed by different linkage policies. As previously identified through dendrograms, the computation of *r* confirms our initial observations that average and complete linkage policies generate clusters more similar to FCM, and that single linkage produces significantly different structures.

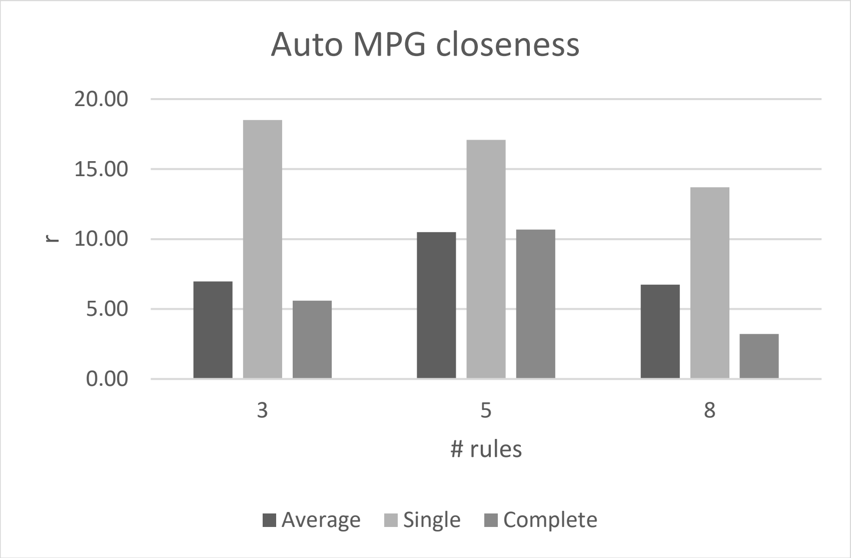


Figure 5.2: Bar plots showing closeness results for all data sets. Each grouping contains results for linkage types, average, single and complete, respectively

Different linkage policies resulting in different data structures is interesting, and we wish to explore how these differences impact the performance of the FRBS. It is not intrinsically detrimental that hierarchical clustering and FCM return different rules, and if the clustering results were identical there would be no value to the exploration of this topic. To assess the impact of hierarchical clustering on rule formation, we evaluate the predictive performance of FRBSs formed through different hierarchical linkage policies and compare their performance.

Table 5.2: RMSE results for Auto MPG, various parameterization

Auto MPG - RMSE ± std_dev					
c	**m**	**Average linkage**	**Single linkage**	**Complete linkage**	**FCM-based model**
3	***1.1***	*6.30±1.41*	*6.99±1.36*	*4.72±0.47*	*4.55±0.36*
3	***1.5***	*5.76±1.06*	*7.37±1.04*	*4.60±0.45*	*4.37±0.36*
3	**2**	5.64±0.91	7.55±0.85	4.71±0.43	4.30±0.34
3	**2.5**	5.74±0.83	7.64±0.75	4.99±0.41	4.31±0.32
3	**3**	5.89±0.77	7.69±0.72	5.30±0.40	4.35±0.31
5	**1.1**	5.68±0.86	6.67±1.67	4.79±0.73	4.25±0.33
5	**1.5**	5.50±0.74	6.59±1.57	4.70±0.66	4.14±0.31
5	**2**	5.48±0.71	6.45±1.38	4.79±0.61	4.08±0.30
5	**2.5**	5.59±0.71	6.41±1.25	5.02±0.57	4.07±0.28
5	**3**	5.76±0.71	6.48±1.21	5.29±0.54	4.11±0.29
10	**1.1**	5.30±0.83	6.61±1.45	5.00±0.48	4.11±0.27
10	**1.5**	5.14±0.74	6.13±1.32	4.80±0.46	4.05±0.27
10	**2**	5.01±0.66	5.71±0.93	4.69±0.45	4.06±0.29
10	**2.5**	5.05±0.58	5.55±0.60	4.80±0.41	4.12±0.29
10	**3**	5.21±0.53	5.59±0.49	5.03±0.37	4.16±0.29

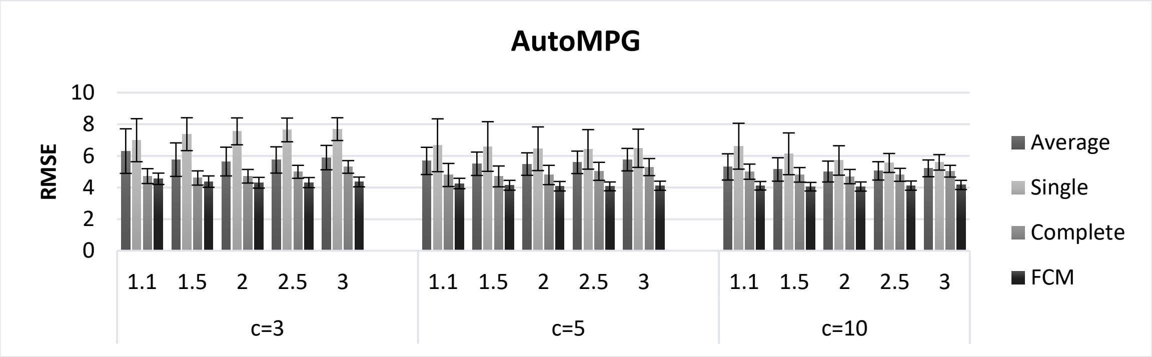


Figure 5.3: Bar graphs for RMSE, stddev as error bars, x-label is m. Each grouping contains, from left to right, RMSE for three linkage types, average, single and complete as well as FCM

Analyzing the experimental results in Table 5.2 (graphically shown in Figure 5.3), we recognize that, in general, hierarchical clustering is unable to produce rule-based models that are of higher quality than those constructed using FCM. The differences are statistically significant (*t*-test completed for a confidence level *p* = 0.05) with the exception of the first two experiments reported in Table 5.2, namely *c*

= 3 and ρ = 1.1 and 1.5, shown in italics. In these two specific cases, complete linkage demonstrates similar performance to FCM; however, average and single linkage are still easily outperformed.

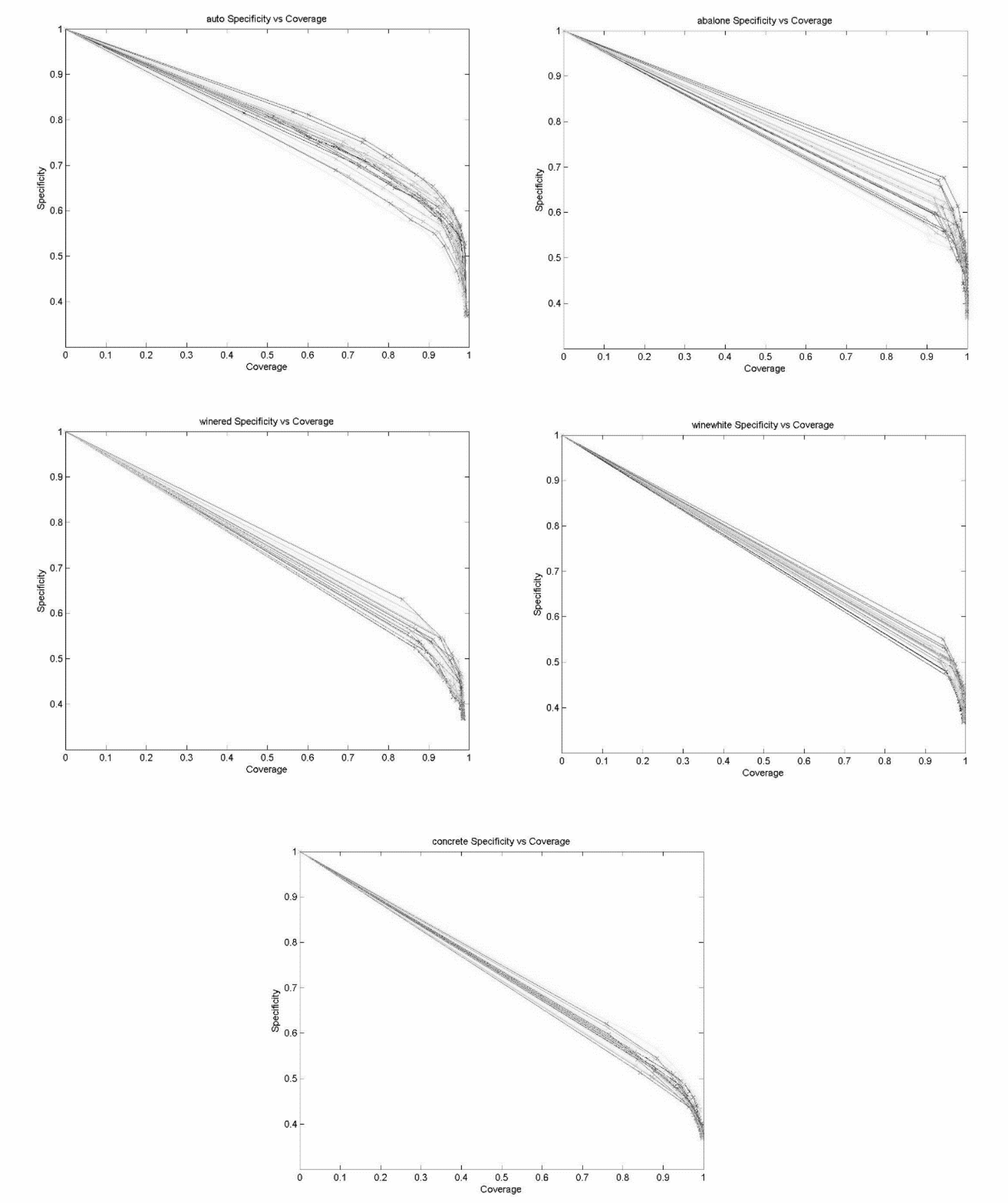


Figure 5.4: Specificity (y-axis) vs Coverage (x-axis), c = [3;5;8], m = [1.1;2.0;3.0], linkage = [single, average, complete], ρ = 0, 0.1, 0.2; … 1.0

As the use of hierarchical clustering for fuzzy rule extraction is not a well-studied methodology, we should not blindly trust traditionally accepted values of the fuzzification coefficient without experimental justification. In [111], this aspect is thoroughly explored through a set of exploratory experiments, and in this dissertation, Figure 5.3 and Table 5.2 can be used to assess this parameter. These results reinforce our previous observations that complete linkage seems to perform best in the formation of fuzzy rules. With respect to the choice of *m,* these results, as well as more experimentation in [111], indicate that *m* does not have a clear impact on the performance of the model. As such, we do not commit to any single value and continue to experiment with a range of potential fuzzification coefficients.

5.4.2 Coverage vs Specificity

The experiments in Section 5.4.1 are performed using a non-granular FRBS, with the intent of exploring the effect of the fuzzification coefficient and linkage policy on the formation of fuzzy rules via hierarchical clustering. In this study, we are primarily concerned with the development of partially granular fuzzy rules, and in this section, we present experimental results focused on granular fuzzy models and their evaluation through coverage and specificity.

In the proposed methodology, the size of the information granules (intervals) are controlled by an input parameter ρ. This parameter, by growing and shrinking the granules, has the apparent effect of controlling the tradeoff of justifiable granularity: coverage at the cost of specificity. As a result, the choice of ρ is a critical parameter which deserves specific attention. We analyze this parameter by varying its value in the range [0,1], and comparing the resulting coverage and specificity scores.

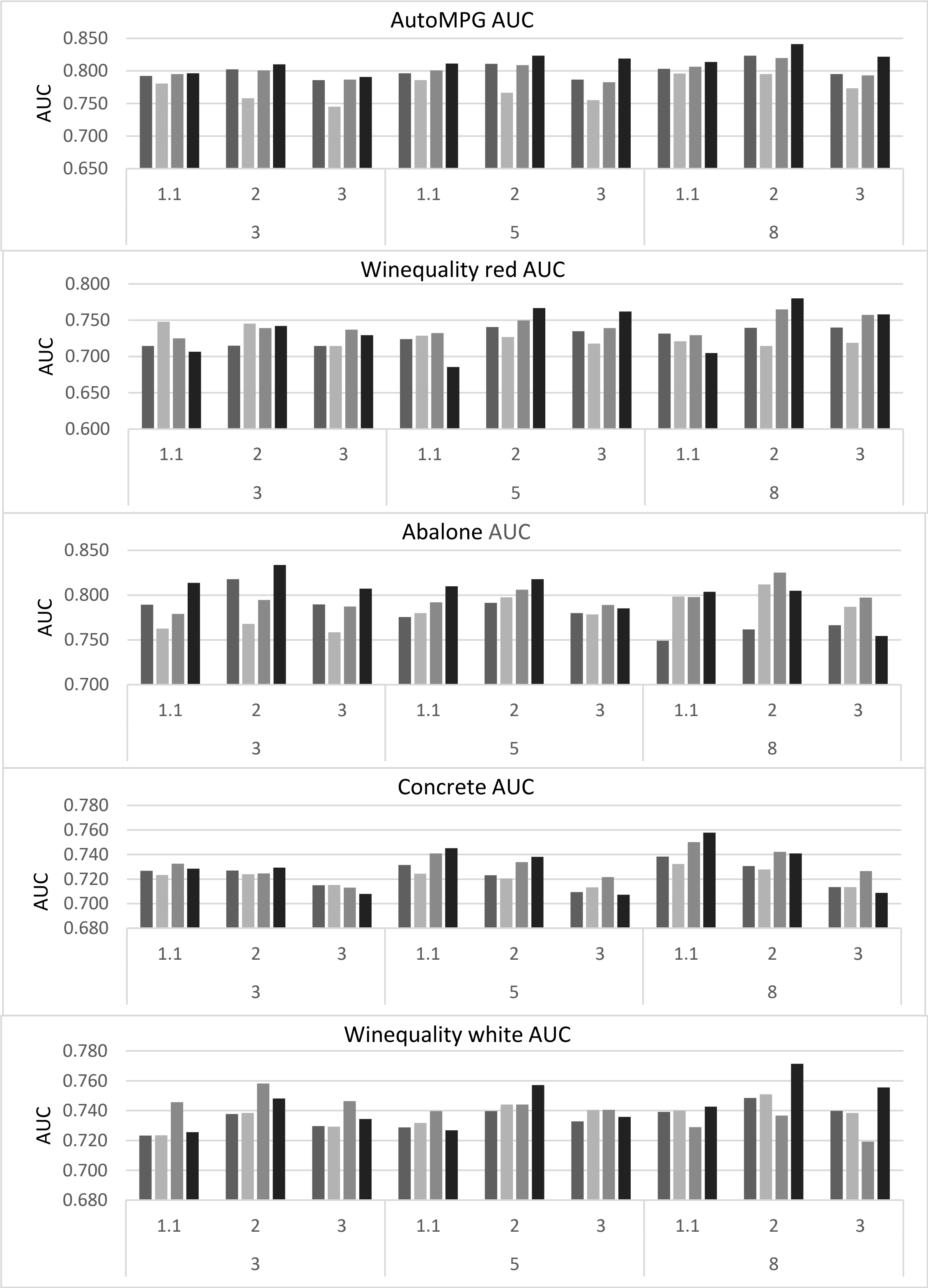
Figure 5.4 contains the curves resulting from these experiments, performed using 5 real world datasets from the UCI machine learning repository [196]. As expected, we observe that when $\rho = 0$ specificity is maximized but coverage is zero; this is an obvious side effect of zero length intervals. At the other extreme, when $\rho = 1$, we attain 100% coverage; however, specificity is extremely poor as the predicted intervals are arbitrarily large in order to fully contain the data. Examining the shape of each curve, we look to find those values of ρ where the tradeoff between coverage and specificity is maximized. A pattern in these graphs is that only small values of ρ are required to attain relatively high coverage, and that, as the size of the intervals grows, the payoff of increased coverage dwindles quickly. This is an encouraging observation, as it demonstrates that our models are able to achieve a high degree of coverage without requiring large (unspecific) granules.

In Table 5.3, we compute the Area Under the Curve (AUC) for the above plots. AUC values are useful in determining which parameterizations perform best, as we want to maximize both coverage and specificity. Certain results are highlight in bold, indicating promising parameterizations. When analyzing

AUC, a higher value indicates better performance, as it is our goal to maximize both evaluation metrics, and a larger AUC quantifies this goal across the full range of ρ values. Once again, as in previous experiments, FCM generally outperforms hierarchical clustering, and of the linkage policies tested complete linkage performs best on average.

Table 5.3: AUC values for coverage vs specificity. Higher values in bold, lower underlined.

Parameters			**AUC**				
c	**m**	**Link Type**	**AutoMPG**	**Abalone**	**Wine-Red**	**Wine-White**	**Concrete**
3	1.1	Average	0.792	0.766	0.715	0.723	0.727
		Single	0.781	0.763	0.719	0.723	0.723
		Complete	0.795	0.779	0.725	0.719	0.732
		FCM	0.796	0.754	0.707	0.726	0.729
	2	Average	0.803	0.762	0.715	0.738	0.727
		Single	0.758	0.768	0.715	0.738	0.724
		Complete	0.801	0.795	0.739	0.737	0.725
		FCM	0.810	0.805	0.742	0.748	0.729
	3	Average	0.786	0.749	0.715	0.730	0.715
		Single	0.745	0.758	0.721	0.729	0.715
		Complete	0.787	0.787	0.737	0.729	0.713
		FCM	0.791	0.804	0.729	0.734	0.708
5	1.1	Average	0.797	0.780	0.724	0.729	0.732
		Single	0.786	0.780	0.718	0.732	0.724
		Complete	0.801	0.792	0.732	0.740	**0.741**
		FCM	0.812	0.785	0.686	0.727	0.745
	2	Average	**0.811**	0.791	0.741	0.740	0.723
		Single	0.767	0.797	0.727	0.744	0.721
		Complete	0.809	**0.806**	**0.750**	0.744	0.734
		FCM	0.824	0.818	0.767	0.757	0.738
	3	Average	0.787	0.775	0.735	0.733	0.709
		Single	0.755	0.779	0.729	0.740	0.713
		Complete	0.783	0.789	0.739	0.740	0.722
		FCM	0.819	0.810	0.762	0.736	0.707
8	1.1	Average	0.803	0.789	0.731	0.739	**0.738**
		Single	0.796	0.798	0.715	0.740	0.732
		Complete	0.806	0.798	0.729	**0.746**	**0.750**
		FCM	0.814	0.807	0.705	0.743	0.758
	2	Average	**0.824**	**0.818**	0.740	**0.748**	0.731
		Single	0.795	**0.812**	0.745	**0.751**	0.728
		Complete	**0.820**	**0.825**	**0.765**	**0.758**	**0.742**
		FCM	0.841	0.834	0.780	0.771	0.741
	3	Average	0.795	0.789	0.740	0.740	0.713
		Single	0.774	0.787	**0.748**	0.738	0.713
		Complete	0.793	0.797	**0.757**	**0.746**	0.727
		FCM	0.822	0.814	0.758	0.755	0.709


AutoMPG AUC

Winequality red AUC

Abalone AUC

Concrete AUC

Winequality white AUC

Figure 5.5: AUC values for AutoMPG. Bar series represent the average, single, and complete linkage policies followed by FCM in left to right order for each parameterization.

Visualized graphically in Figure 5.5, the general trends of each linkage policy are more easily assessed.

First, we notice that the single linkage policy is almost never the best performer. Indeed, out of all the presented experiments, on only two occasions does single linkage result in the highest AUC, both for the red wine quality dataset, and both when $m = 1.1$. Second, we recognize the continued pattern of FCM outperforming all linkage policies on average. This is visible to different extents for all tested datasets, with complete linkage being the second-best methodology overall.

5.4.3 Selected Experimental Results

In previous experiments, we have explored and analyzed the potential parameterizations of our methodology. We use this data to perform a focused set experiments using the strongest parameterizations based on previous findings. The choice of ρ in the following experiments is determined for each dataset from inspection of the curves given in Section 5.4.1 and are chosen to be a "good" choice, rather than a necessarily optimal one.

Table 5.4: Selected specificity and coverage results

c	m	link	ρ	Coverage	Specificity
AutoMPG					
5	2	average	0.50	0.92 ± 0.04	0.60 ± 0.05
5	2	complete	0.50	0.92 ± 0.04	0.61 ± 0.05
8	2	average	0.50	0.90 ± 0.04	0.63 ± 0.05
8	2	complete	0.50	0.89 ± 0.04	0.64 ± 0.06
Abalone					
5	2	complete	0.20	0.96 ± 0.02	0.53 ± 0.03
8	2	average	0.20	0.97 ± 0.02	0.54 ± 0.03
8	2	single	0.20	0.98 ± 0.02	0.53 ± 0.03
8	2	complete	0.20	0.99 ± 0.01	0.54 ± 0.03
Wine-red quality					
5	2	complete	0.20	0.95 ± 0.04	0.49 ± 0.03
8	2	complete	0.20	0.94 ± 0.03	0.54 ± 0.01
8	3	single	0.20	0.98 ± 0.00	0.46 ± 0.02
8	3	complete	0.20	0.97 ± 0.01	0.50 ± 0.02
Wine-white quality					
8	2	average	0.10	0.95 ± 0.01	0.53 ± 0.02
8	2	single	0.10	0.95 ± 0.01	0.54 ± 0.02

8	2	complete	0.10	0.95 ± 0.01	0.55 ± 0.02
8	3	complete	0.10	0.96 ± 0.02	0.52 ± 0.02
Concrete					
5	1.1	complete	0.50	0.95 ± 0.01	0.47 ± 0.02
8	1.1	average	0.50	0.94 ± 0.02	0.48 ± 0.02
8	1.1	complete	0.50	0.92 ± 0.03	0.50 ± 0.02
8	2	complete	0.50	0.98 ± 0.01	0.41 ± 0.01

The results of these focused experiments are presented in Table 5.4. First, let us focus on which parameterizations resulted in the best performing models. The choice *c* is often skewer towards higher values, which is not surprising as error decreases monotonically with the # rules; however, the best performing fuzzification coefficients are evenly spread amongst the tested values but are often consistent within a given datasets. This is indicative of data set dependent parametrization, which suggests that the methodology itself is not biased towards a specific choice of *m*. The choice of ρ is also widely varied, again indicating that our methodology is able to generate high quality models given the correct parameterizations, but that the specific choices of each parameter are dataset dependent. Finally, all three linkage policies make an appearance, though we acknowledge that, consistent with previously experiments, complete linkage is clearly dominant.

As a final point of discussion, we take a more thorough look at the choice of the interval parameter ρ. Particularly, we observe that high coverage is achieved with very small values of ρ in the case of the white wine dataset, and a slightly larger value in the case of the red wine and abalone datasets. In the case of the other two datasets a more conservative value of 0.5 is also successful in achieving high degrees of coverage. This spread suggests that a more thorough study focusing the choice of this parameter may be warranted, or that optimization procedures targeting this parameter could be used to obtain a more carefully tuned model. Expanding on this idea, a more robust methodology may consider different ρ values for different information granules within a single model, potentially improving overall performance.

5.4.4 Effectiveness of the proposed methodology

The previous sections provide a significant amount of experimental data demonstrating the behavior of the proposed granular FRBSs constructed through the use of hierarchical clustering, and at this juncture we assess the overall effectiveness of our methodology as a modelling tool. We are interested in answering three main questions:

- Is hierarchical clustering an effective tool for fuzzy rule generation?
- How well does the proposed granular rule format realistically improve model interpretability?
- Are the evaluation methods proposed for this type of model sufficient?

Our experiments have demonstrated that hierarchical clustering is not a convincing alternative to FCM for the extraction of fuzzy rules from data. Throughout our experiments, and regardless of linkage policy, hierarchical clustering was at best only able to match the performance of FCM and never demonstrated a consistent ability to provide a *better* result. This, in combination with the fact that hierarchical clustering is more computationally intensive, leaves no good argument for the application of hierarchical clustering to fuzzy rule extraction.

While our experimentation is focused on assessing the quantitative performance of the granular models, a key facet of information granulation is the desire to improve human readability. While we have stated that information granules (intervals) improve model interpretability, so far, we have failed to concretely demonstrate this assertion. To address this, we provide a sample granular rule-base at this juncture for consideration, shown in Figure 5.6.

ρ = 0.2 [6.0 160.6 122.6 2913.3 12.6 80.6 3.0] –> [18.0 36.4]

[4.5 138.9 84.8 2557.6 16.4 76.5 1.7] –> [20.0 37.0]

[8.0 346.3 160.0 4137.6 12.7 73.8 1.00] –> [13.0 17.5]

ρ = 0.8 [6.0 160.6 122.6 2913.3 12.6 80.6 3.0] –> [11.0 39.1]

[4.5 138.9 84.8 2557.6 16.4 76.5 1.7] –> [18.0 46.6]

[8.0 346.3 160.0 4137.6 12.7 73.8 1.0] –> [9.0 19.4]

Figure 5.6: Sample granular rule-base, Link: average; Clusters: 3; m: 2.00

If contrasted directly with Mamdani style fuzzy rules, the advantage is, we believe, clear. A single value representing a fuzzy set is very difficult to interpret, as the degree of information communicated is very limited. The simple process of forming an interval (information granule) provides significantly more information to the reader regarding the domain of each fuzzy rule, and consequently the predicted outputs. By using information granules, readers are made more aware of a prediction's context; specifically, a predicted interval communicates a value and a degree of uncertainty, as implied by the size of the interval. Further, if a reader studies the rules themselves, the knowledge extracted from the system benefits from these same advantages, and information such as the degree of rule overlap and uncertainty are effectively communicated.

Finally, let us discuss the evaluation metrics proposed for this type of granular model. In this study, we use coverage and specificity in combination to evaluate the performance of a granular model. Coverage provides an intuitive granular equivalent to accuracy, as predictions are (to some degree of uncertainty) correct when the target value is contained within the predicted interval. This is counteracted by the measure of specificity, which promotes narrow intervals in the interest of keeping the predicted values as precise as possible. In combination, these two metrics are a satisfactory indicator of performance; however, an open question would be their relative degrees of importance. Can these criteria can be effectively combined into a single evaluation metric, and if so, through what operator or equation? The tradeoff between coverage and specificity is realized through the parameter ρ, and further study of this parameter has already been mentioned as a promising direction of future research.

5.5 Concluding Remarks

This chapter has detailed a methodology for the generation and evaluation of a partially granular fuzzy rule-based model, constructed through hierarchical clustering and justifiable granularity. We have defined two contrasting evaluation metrics, coverage and specificity, for assessing this model format, and have provided extensive experimentation demonstrating the effectiveness of the proposed rule generation, granulation, and granular evaluation methodologies. On the topic of using hierarchical clustering as a vehicle for rule extraction, we have contrasted the use of hierarchical clustering with the standard choice FCM clustering to assess its feasibility.

The use of hierarchical clustering as a rule generation strategy is not convincing as compared to FCM, as demonstrated through our experimentation. Complete linkage provided the most promising results of the tested policies; however, even the best hierarchical results were only on par with FCM. It is possible that by using different, more complex, or specifically tailored rule generation strategies, improved performance could be attained; however, using our straightforward methodology, no visible advantage has been observed, and we would not endorse the use of hierarchical clustering in this manner for fuzzy rule extraction.

The granularization of the FRBS provided a significantly more positive result. Our application of information granules successfully improved the interpretability of the fuzzy rule-based model, and the proposed evaluation metrics satisfy both requirements of justifiable granularity, providing conflicting measures of quality through which an appropriate balance can be attained. The presented example granular rules demonstrate the improved interpretability of this model format, and coverage and specificity provide an adequate indication of model performance. There remains ample room for further study in this area, both on the topic of granular evaluation strategies, and on the optimization of modelling parameters. Our experiments have demonstrated the successful evaluation of granular rule-based model performance, and we have established that the proposed methodology is easily able to attain very high degrees of coverage for most datasets, while keeping intervals adequately specific.

Granular fuzzy models are an area of research which holds a great deal of potential for future study. In this chapter, we have taken one small step into the realm on information granules in fuzzy modeling and identified many avenues for further improvement and advancement. Although the use of hierarchical clustering did not provide any improvement versus FCM, it is important for researchers to continue to examine such unexplored areas so that informed decisions can be made in future studies.

6 BOOSTING WITH FUZZY RULES

Boosting has been a pivotal development in machine learning, and the topic has been thoroughly studied, with applications in many areas of expertise. Boosting defines a methodology through which a collection of relatively poor (weak) learners can be combined into an ensemble which functions as a single strong learner. This performance improvement is achieved by identifying those training instances which are poorly modelled by the existing ensemble and targeting them at the next iteration of learner generation.

Due to the overwhelming success boosting has enjoyed as a generalized ensemble building methodology, the application of boosting has been widespread, covering diverse weak learner architectures, and many areas of expertise. Despite this significant research attention, boosting with fuzzy systems has not been well-studied. This gap in the literature motivates us to explore the topic of weak fuzzy learners in a boosted ensemble. The general topic of boosting with fuzzy learners is of significant interest; however, the scope of potential weak fuzzy learners is quite large, and the existing literature quite limited. For this reason, we limit this initial research to a specific case study, applying one type of fuzzy learner and assessing how it performs in a boosted ensemble.

In this chapter, we are interested in the application of FRBSs as the weak learners in a boosted ensemble, a topic which, to our knowledge, has not been addressed in the literature. We pose and answer such crucial questions as:

- To what degree can accuracy be improved with an ensemble of FRBS?
- How well do fuzzy classification rules function as weak learners in a boosted ensemble?

And address necessary subsequent technical queries such as:

- How well does the rule generation process adapt to weighted data?
- Are the weak learners diverse enough to promote successful boosting?
- How well are fuzzy classification rules able to perform?
- Compared to a single fuzzy rule-base?
- Compared to other boosted weak learners?

To the extent to which they are necessary in a boosting context.

We are also motivated to explore the effects of certain parameterizations, such as the number of rules in each weak learner. As the number of fuzzy rules generally implies the accuracy of a FRBS, we explore the effect of this relationship in an ensemble: are fewer rules able to match more complex rule bases in terms of accuracy through boosting? To our knowledge, the use of FRBS in a boosted ensemble has never been studied, and the style of classification rules used in this research have not been widely applied.

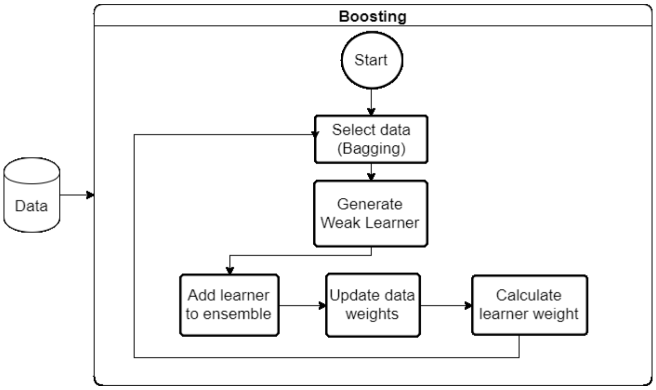


Figure 6.1: Generalized Boosting Workflow

In this work, we make use of some well-known constructs for both the generation of the weak learners, and the construction of a boosted ensemble. The component weak learner takes the form of a tailored fuzzy classification rule-based system. For the construction of the boosted ensemble we use a multi-class variant of AdaBoost, SAMME, the implementation details of which are defined in Section 3.5.3. The general workflow of our methodology is given in Figure 6.1, and defines a multi-class boosting approach in which FRBSs are used as the component weak learners of the ensemble.

6.1 Design of the Weak Learner

This section outlines a detailed methodology for the construction of a fuzzy classification rule-based system, to be used as the component weak learner in a boosted ensemble. While many aspects of this model architecture are similar to a standard TS-style FRBS, we make key changes to address the requirements of the boosting process and a classification setting.

With respect to the goal and function of the boosting mechanism, the critical component of weak learner generation is the response of the procedure to weighted data. Boosting improves ensemble performance by identifying those instances which are poorly modelling by the current ensemble and increasing the relative weight of those instances. Consequently, the new weak learner must consider these weights during its construction.

In this chapter, we target classification problems and are therefore concerned with the construction of fuzzy classification rules. This is a step away from classical TS-fuzzy rules, so a new rule format and generation procedure must be defined. While defining the classification architecture, we additionally look for opportunities to promote successful boosting. In our case, this takes the form of ensuring ample opportunity for the generation procedure to incorporate data weights into the rule extraction process.

6.1.1 Weighted Fuzzy C-Means Clustering

The computation of a fuzzy partition, used to form the condition parts of fuzzy rules, is unchanged from standard procedure outlined in Section 3.1 and 3.4, with one key difference: data weights. The consideration of data weights in the rule generation procedure is key to successful boosting, and as a result, we need to define a variation of FCM which takes them into account.

The key component to weighted FCM is the modification of the objective function, given for the original algorithm in (3.1), to:

$$Q = \sum_{j=1}^{N} \sum_{i=1}^{c} u_{ij}^{m} w_j \left\| \boldsymbol{x_j} - \boldsymbol{v_i} \right\|^2$$

(6.1)

Where we have made the addition of w_j, indicating the weight of the *jth* instance. Recalculating the necessary equations from this modified objective function, we find that the membership function is unchanged, and the computation of cluster prototypes, originally given in (3.2), becomes:

$$\boldsymbol{v_i} = \frac{\sum_{k=1}^{N} u_{ik}^{m} w_k \boldsymbol{x_k}}{\sum_{k=1}^{N} u_{ik}^{m} w_k}$$

(6.2)

Where once again we have added data weights to the equation in two places. The modified FCM algorithm can now be used in a weighted environment for the computation of rule conditions.

6.1.2 Design of Fuzzy Classification Rules

The standard fuzzy rule format (TS-style fuzzy rules) is only compatible with real valued consequents. In this chapter, we are interested in tackling classification problems, which possess nominal class labels and not real valued outputs. For this reason, we need to design a new format of fuzzy rule which is compatible with a classification setting.

In Mamdani style fuzzy rules, described in (3.10), the output part of the rule, b_i, is a real value. To update this format for handling classification problems, we consider the following modification to the consequent of the rule:

$$\boldsymbol{b_i} = [p_{class1}, p_{class2}, \dots, p_{classK}]$$

(6.3)

Where p_{class} indicates the *probability* of an instance matching that rule belonging to a given class, for some *K* class classification problem. In this format, the class label of a given instance is predicted by the following equation:

$$\boldsymbol{class_k} = \sum_{i=1}^{c} u_{ik}\boldsymbol{b_i}$$

(6.4)

Where $\boldsymbol{class_k}$ is a vector of class probabilities, *c* is the number of rules, u_{ik} is the degree of membership of instance *k* to rule *i* from the partition matrix *U*, and $\boldsymbol{b_i}$ is, as before, the consequent of rule *i.* To distill this vector into a predicted class, we find the highest predicted probability, and predict the class label associated with that value:

$$cl \ = \ arg\ max_{j=1,2,\dots K}\boldsymbol{class_j}$$

(6.5)

Where *cl* is the predicted class label, determined by the index of the maximum value in the computed array of class probabilities, and *argmax* returns the index of the maximum value of a vector.

With a classification rule format defined, we are left with the task of rule consequent extraction from data. The consequents in the proposed format represent the probabilities of matching data belonging to a given class. Probabilities can be computed by considering the number of instances in a cluster with a given class label and calculating the consequent probabilities based on this knowledge. To maintain compatibility with boosting, we consider data weights in this formulation. Formally, the consequent probability for each class is calculated at each cluster according to:

$$p_{class\,j} = \frac{\sum_{k=1}^{M_i} w_k I\left(class_j, class(\boldsymbol{x_k})\right)}{M_i}$$

(6.6)

Where p_{class_j} is the consequent probability of an instance with class label *j* belonging to the rule under consideration, here denoted *i,* for *j = 1, 2, ... K*. This computation is performed for each rule, *i = 1, 2, ...c*, and M_i denotes the number of training instances belonging to cluster *i*. Importantly, the summation in the numerator is computed only for the set of training instances belonging to the rule under consideration. Finally, *I* is an indicator function:

$$I(a,b) = \begin{cases} 1 & if\ a = b \\ 0 & otherwise \end{cases} \tag{6.7}$$

And its negation is described as:

$$\neg I(a,b) = \begin{cases} 0 & if\ a = b \\ 1 & otherwise \end{cases} \tag{6.8}$$

To determine which training instances belong to which rules, we consider the maximum fuzzy membership in the partition matrix *U* to indicate the rule to which it belongs:

$$s_k = arg\, max_{i=1,2,\ldots c} u_k \tag{6.9}$$

Where s_k indicates the cluster to which instance $\boldsymbol{x_k}$ belongs, and u_k is the *kth* column of *U,* containing the degrees of membership of $\boldsymbol{x_k}$ to each of the *c* fuzzy rules. This represents a defuzzification of the fuzzy partition, allowing us to assign each instance to a single cluster when performing consequent calculations. The resulting crisp partition is then used in (6.6), where we use *c* distinct sets of training instances, with the rule assignment of each instance determined by (6.9).

The final component of our weak learner design is a modifier function on the output of the system; specifically, a sigmoidal modifier whose parameters are tuned to maximize predictive ability. The motivation for this addition is twofold. First, when configured with a high steepness, a sigmoidal modification layer results in near binary outputs, and consequently much clearer classification. Second, by optimizing the parameters of each sigmoidal modifier, we introduce an additional degree of flexibility in the training of the weak learners. This serves both to promote better individual learner performance and to provide another vector for learner adaption to changing data weights.

Each learner exhibits a single sigmoidal function per class. The sigmoidal functions take the following form:

$$\varphi_{jk} = \frac{1}{1 + exp(-a(class_{jk} - z))}$$

(6.10)

Where φ_{jk} is the final class certainty for the jth class with respect to the kth input instance, a is a steepness parameter, $class_{jk}$ is the input certainty for the jth class of the kth input from the FRBS computed using (6.4), and z is a configurable parameter. These z parameters are optimized through gradient decent with the goal of maximizing classification accuracy.

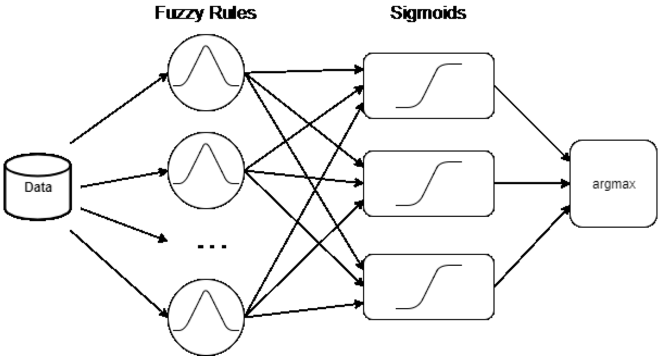


Figure 6.2: Architecture of the weak learner

Our completed learner architecture, visualized in Figure 6.2, is composed of *c* fuzzy classification rules, each of which predict class labels through a probability based consequent scheme, and whose predictions are combined through (6.4). These class probabilities are then modified by the optimized sigmoidal functions, defined in (6.10), before a final class prediction is made by determining the maximum resulting class probability, as formulated in (6.9).

6.2 Fuzzy Rule Boosting

To this point, we have concentrated on the generation of the weak learners for use in a boosted ensemble. What remains are the specifics of the boosting implementation, in combination with the proposed learner. In this study, we use the SAMME boosting algorithm to tackle multi-class classification problems, the specifics for which are given in Section 3.5.2, and the fuzzy classification rule-based model defined in Section 6.1 is used as the component weak learner.

Boosting is performed in the standard manner, by iteratively generating weak learners using data selected through bagging and the current data weights. The new learner is evaluated, its ensemble weight is computed, and data weights are updated in accordance with the new learner's performance. As is common in boosting studies, a safety condition is added at the end of each iteration that checks if the new learner improves the overall ensemble classification accuracy. If it does not, the new learner is rejected and not added to the ensemble. This safety measure guards against difficult to mitigate situations such as poorly initialized models, which would otherwise negatively contribute to the ensemble.

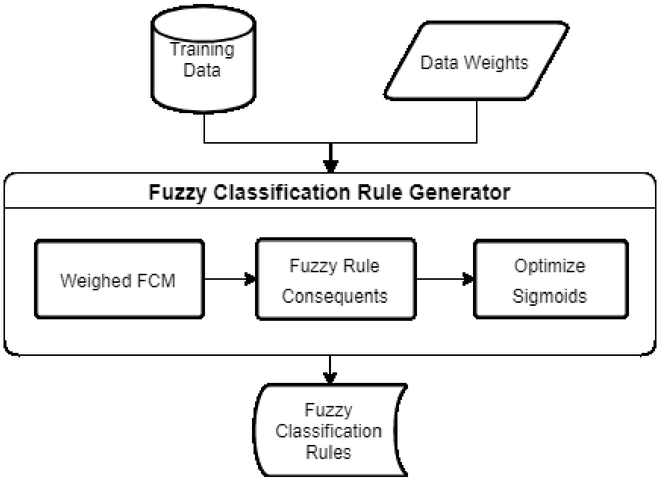


Figure 6.3: Weak Learner Generation procedure

The prediction of the completed ensemble for a given input instance is a standard boosting approach:

$$F(\boldsymbol{x}) = \sum_{t=1}^{m} \propto_t h_t(\boldsymbol{x})$$

(6.11)

Where $\boldsymbol{x}$ is a given instance, *F* is the completed ensemble and *m* is the size of the ensemble. Each member fuzzy model, h_t, is evaluated according to (6.4) and (6.9) considering the addition of the sigmoidal modifier (6.10):

$$cl \ = \ arg\ max_{j=1,2,\ldots K} \varphi_j \left(class_j\right)$$

(6.12)

Where *cl* indicates the class label prediction, and (6.12) is a modified version of (6.9), which passes the initial class certainties from the fuzzy classification rules through the sigmoidal modification functions before making a final prediction. Ensembles are computed to a certain size, as specified by an input parameter.

The generalized workflow for weak learner generation (Fuzzy Classification Rules) is given in Figure 6.3. This sub-process is used in combination with the generalized boosting workflow given in Figure 6.1 to fully define the proposed methodology.

6.3 Experimental Studies

This section provides a comprehensive set of experimental studies, examining in detail the performance of our fuzzy rule boosting methodology. First, we provide some detailed case studies, demonstrating the behavior of the proposed weak learners outside of an ensemble. Next, we examine a sample boosted ensemble in detail, demonstrating accuracy improvement as the ensemble grows. Finally, we provide two sets of comprehensive boosting results. The first set highlights the degree of accuracy improvement exhibited by the boosted ensembles as compared to a single FRBS. The second set compares the proposed methodology to equivalent ensembles composed of standard weak learners. All of these experiments are performed with publicly available real-world data sets.

The proposed methodology makes use of several algorithms which require configuration via input parameters. We keep these parameters consistent throughout our experiments wherever possible for consistency. For FCM clustering, the fuzzification coefficient $m = 2.0$ is used and the termination criteria is set as a change in objective function of less than 10^{-5} or a maximum of 500 iterations. This configuration aims to achieve reasonable runtimes without sacrificing accuracy, and the fuzzification coefficient used is a standard choice which performs well in most situations. The sigmoidal steepness parameter α is chosen to be adequately high such that the model's classification is near binary; we have chosen a value of $\alpha = 20$. For gradient descent we use a descent rate of 0.1 and a momentum term of 0.5. These values were finalized experimentally by finding a balance between the number of iterations required to reach equilibrium and the precision of the optimized parameters. The same termination criterion as those defined for FCM are used here, with a similar goal and justification.

6.3.1 Case Study Experiments

This section details two simple case studies, providing insight into the performance of both the proposed learner as a standalone model and as a component weak learner in a boosted ensemble.

6.3.1.1 Sigmoidal Modifier Case Study

First, we examine a simple case study demonstrating the effect of the sigmoidal modifier on the predictive accuracy of a single learner. This demonstrates the ability of the sigmoidal modifier to effect classification, justifying its inclusion in the architecture. We use the well-known Iris dataset from the UCI machine learning repository [196] as a simple example for these initial experiments.

Using the learner generation methodology defined in Section 6.1, we construct a single FRBS and optimize the sigmoidal modifier using gradient decent. At this juncture, we assert that the purpose of the sigmoidal modifier is not strictly to improve classification accuracy, but rather to add additional degrees of flexibility to the learner, with the goal of enhancing learner responsiveness in boosting.

In the following examples, five models (weak learners) are generated independently from one another using the Iris dataset, and a 75%/25% training/testing data split. Each model is constructed with two rules, *c = 2*, and the state of each model is recorded after training is complete. For each model, classification accuracies are calculated with and without the addition of the sigmoidal modifier, listed as *without* sigmoid, and *with* sigmoid. Table 6.1 reports the training and testing accuracies for each of the five learners, listed as M1 through M5 denoting the first through fifth model.

Table 6.1: Accuracies with and without sigmoidal modifier

Model	**With Sigmoid**		**Without Sigmoid**	
	Train	**Test**	**Train**	**Test**
M1	0.678	0.632	0.678	0.632
M2	0.705	0.737	0.67	0.658
M3	0.678	0.658	0.67	0.658
M4	0.67	0.658	0.67	0.658
M5	0.714	0.553	0.705	0.553

The accuracies presented in Table 6.1 provide insight into the effect of the sigmoidal modifier on classification performance. The critical observation to be make from these experiments is that in no experiments does the performance of the model decrease from this addition, and in certain cases (M2, M5), the modifier improves the classification rate. This demonstrates that the modifier is having its intended effect, and we may expect some increase in classification accuracy from some weak learners from its inclusion.

6.3.1.2 Boosted Ensemble Case Study

Next, we examine the boosting process is greater detail; specifically, how boosting improves classification accuracy through the iterative generation of specialized weak learners. Once again, we use the Iris dataset as the sample problem for this case study.

In all boosting experiments in this chapter, the number of rules in each weak learner is static within the ensemble, as are all other algorithmic parameters. Consider, as a first example, a boosted ensemble where each model is constructed with two fuzzy rules and an ensemble size of four.

Table 6.2: Boosting progression, Iris, rule-base side of 2s, ensemble size of 4

Iteration	**α**	**Accuracy**
1	2.72	0.88
2	1.97	0.89
3	1.39	0.92
4	1.8	0.94

Table 6.2 presents two key values: the weight each weak learner in the ensemble, α, and the ensemble accuracy at each iteration. This data serves two purposes. First, and most important, the increase in accuracy as boosting proceeds indicates successful ensemble construction. Second, learner weights taking reasonable values suggests that each learner is contributing positively to the decision-making process, and that the ensemble is functioning as an ensemble, not as a single dominant learner bearing the brunt of the decision-making load. This simple example demonstrates the feasibility of the proposed methodology and the soundness of the implementation through the improvement in accuracy as the ensemble grows.

6.3.2 Accuracy Improvement through Boosting

This section presents experimental results demonstrating the improved classification performance of the proposed methodology, as compared to a single fuzzy classification rule-based system. This serves as a litmus test for the successful application of boosting with the proposed weak learner and indicates how powerful of a tool boosting can be in this context.

All experiments in this section are performed with 10-fold cross-validation, and all individual FRBSs used for comparison contain the same number of fuzzy rules as a single component weak learner in the ensemble. For each dataset, boosting is performed using a range of model sizes and ensemble sizes which are considered in combination. Boosting is designed to construct a single strong learner from a set of weak learners, and consequently, the number of fuzzy rules making up each learner must be kept small to preserve their weak nature; hence, we experiment with *c* values in the range of 2-4.

We consider a number of datasets from the UCI machine learning repository in addition to the Iris dataset:

- *Wisconsin* – Predict breast cancer diagnosis from tumor metrics.
- *Wine Quality* – Classify wine into one of three regions from chemical measurements
- *Glass Identification* – Classify the type of glass from physical properties
- *Magic* – Detect signal from a telescope from observations
- *Banknote Authentication* -Determine if a bank note is authentic or forged

And certain additional datasets coming from the KEEL dataset repository [197]:

- *Heart Disease* – Predict heart disease diagnosis from medical metrics
- *Appendicitis* – Diagnose appendicitis from patient symptoms
- *Bupa* – Diagnose liver disorder from lifestyle metrics

The following results provide comparative accuracies for the listed datasets using a range of experimental parameters. Each dataset is modelled using *c = 2, 3, 4* and ensemble sizes of *m = 2, 4, 6*.

Table 6.3: Boosting results for Iris

Parameters		**Boosting**		**Single Model**	
Size	**Rules**	**Training**	**Testing**	**Training**	**Testing**
2	**2**	0.733	0.720	0.659	0.633
4	**2**	0.872	0.880		
6	**2**	0.890	0.860		
2	**3**	0.912	0.860	0.848	0.840
4	**3**	0.953	0.927		
6	**3**	0.961	0.933		
2	**4**	0.947	0.907	0.874	0.860
4	**4**	0.961	0.960		
6	**4**	0.977	0.967		

The first set of comparative results is given in Table 6.3 and considers the Iris dataset. This table presents training and testing accuracies for two sets of experiments – boosted ensembles and single learners, with single learner accuracies presented only once for each *c*.

Analyzing the results, we identify two keys points of discussion. First, we compare the performance of the boosted ensemble with a single model. In this case, the boosting improvement is obvious and can be observed in all experiments. In fact, there are several cases where the improvement is substantial, including when *c = 2,* where accuracy is improved from 66% to 89% with an ensemble of size 6. In this case, the results are quite clear: boosting has a significant and positive impact on the classification rate, achieving much higher accuracies than a single rule-based model. Our second point of discussion is the performance improvement resulting from increased ensemble size, and improvement is clearly visible in Table 6.3. The corollary of this is that there is certainly an upper limit to the achievable improvement through boosting using FRBSs, and we see evidence of this in the results. With larger ensembles, there are indications that the degree of improvement levels off. In the case of two rules, the jump in accuracy from two learners to four results in a 14% improvement, while the addition of two more learners only adds an additional 2%. This pattern is visible in later results as well.

Table 6.4: Boosting results for Wine Quality

Parameters		**Boosting**		**Single Model**	
Size	**Rules**	**Training**	**Testing**	**Training**	**Testing**
2	2	0.689	0.695	0.663	0.635
4	2	0.714	0.697		
6	2	0.754	0.715		
2	3	0.725	0.709	0.672	0.650
4	3	0.741	0.733		
6	3	0.778	0.733		
2	4	0.765	0.754	0.723	0.700
4	4	0.789	0.768		
6	4	0.810	0.819		

Table 6.4 presents similar results modelling the Wine Quality dataset. Generally, the results are similar in nature to Iris experiments, with boosted ensembles easily outperforming single learners and improvement increasing with ensemble size. In this case, the improvement is less marked, and the increases in performance are incremental, likely indicating that the problem is more difficult for the proposed weak

learner. A notable difference to the previous case is that in this instance we do not necessarily see signs of tapering accuracy improvement, so it may be possible to attain additional improvement with even larger ensemble sizes.

Table 6.5: Boosting results for Wisconsin

Parameters		**Boosting**		**Single Model**	
Size	**Rules**	**Training**	**Testing**	**Training**	**Testing**
2	2	0.969	0.969	0.969	0.969
4	2	0.970	0.968		
6	2	0.970	0.966		
2	3	0.971	0.972	0.970	0.968
4	3	0.972	0.968		
6	3	0.976	0.971		
2	4	0.973	0.968	0.971	0.968
4	4	0.977	0.971		
6	4	0.978	0.971		

As a final detailed example, we examine the Wisconsin dataset, results given in Table 6.5. These experiments represent a slightly different scenario, as the accuracies attained by a single learner are very high to begin with (an indication of a simple classification problem). This provides an opportunity to analyze how boosting performs in the case where there are very few remaining misclassified instances, and consequently little room for improvement. Examining Table 6.5, we note that boosting does not seem to provide improvement in this scenario, regardless of the number of rules or the size of the ensemble. In fact, all experiments perform at about the same level. This may simply indicate that the final few instances which have not been correctly classified are outliers or very difficult instances to classify. More cynically, it may indicate that the architecture of the weak learner is simply unable to handle these instances for whatever reason.

These focused experiments on boosting improvement have demonstrated the ability of the proposed methodology to compute boosted ensembles, successfully improving classification accuracy. As additional experimental evidence, we present a condensed set of comparative experiments modelling the remaining datasets.

Table 6.6: Performance improvement with a boosted ensemble

Dataset	**Dataset attributes**		**Rules**	**Accuracy**		**% Improvement**	
	# Input Variables	**# Classes**		**Training**	**Testing**	**Training**	**Testing**
Iris	4	3	2	0.890	0.860	23.19%	22.67%
			3	0.961	0.933	11.33%	9.33%
			4	0.977	0.967	10.30%	10.67%
Wine Quality	13	3	2	0.754	0.715	9.16%	8.02%
			3	0.778	0.733	10.61%	8.26%
			4	0.810	0.819	8.72%	11.93%
Wisconsin	9	2	2	0.970	0.966	0.11%	–0.29%
			3	0.976	0.971	0.55%	0.29%
			4	0.978	0.971	0.70%	0.29%
Heart Disease	13	2	2	0.658	0.637	7.86%	9.63%
			3	0.712	0.681	9.96%	4.81%
			4	0.718	0.700	8.89%	10.37%
Appendicitis	7	2	2	0.877	0.871	5.25%	4.25%
			3	0.897	0.870	3.28%	1.25%
			4	0.900	0.835	2.66%	–1.00%
Glass	9	7	2	0.518	0.464	15.36%	16.91%
			3	0.599	0.530	17.17%	13.98%
			4	0.625	0.591	11.53%	12.93%
Bupa	6	2	2	0.584	0.574	1.74%	3.02%
			3	0.593	0.566	1.19%	–1.10%
			4	0.614	0.608	3.13%	1.91%
Magic	10	2	2	0.648	0.648	0.00%	0.00%
			3	0.665	0.663	2.15%	1.79%
			4	0.698	0.697	5.33%	5.21%
Banknote Auth.	4	2	2	0.645	0.632	6.36%	4.26%
			3	0.764	0.769	15.83%	15.44%
			4	0.968	0.962	8.99%	8.52%

Table 6.6 presents the additional comparative experiments for all datasets in a reduced format. The training and testing accuracies reported for the boosted ensembles are for an ensemble size of 6 and the % improvement is reported in comparison to a single model of the same parameterization.

As visible in the % increase columns, boosting using small sets of fuzzy classification rules as weak learners is largely successful. The Iris dataset demonstrates the most successful case – with only 2 rules per learner, the use of a boosted ensemble increases classification accuracy by nearly 30%. The Wine, Heart Disease, Glass, Appendicitis, and Banknote Authentication datasets each show varying degrees of success, with consistent accuracy improvements. The Wisconsin, Bupa, and Magic datasets show less successful cases, where little or no improvement is achieved. The reason for these poor performing experiments could be attributed to a handful of factors. First, higher dimensional problems (Wine, Heart) are less suitable to FRBSs, and datasets with many classes (Glass) may be more difficult for our probability-based classification rules to model. Given this, we speculate that the weaknesses of the component learners likely exhibit themselves in a boosted ensemble. Second, because rules are always formed via clustering (weighted), and class distributions may not be structured in a clustered manner, it is intuitively obvious that the proposed weak learner may not be capable of modelling certain problems. Finally, when the number of classes is large, weak learners containing a small number of rules will have difficulty partitioning data into class labels when there are more classes than rules – it is impossible to accurately classify five classes using only two rules.

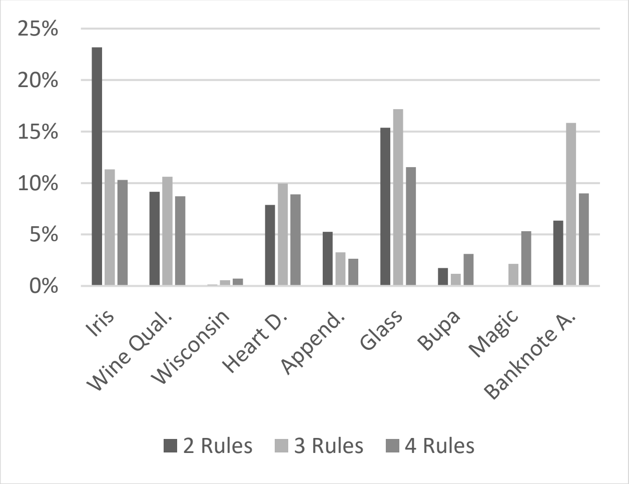


Figure 6.5: Training performance improvement by #

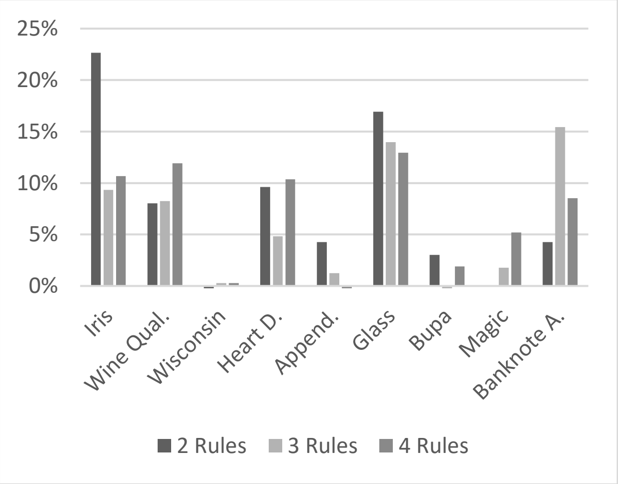


Figure 6.5: Testing performance improvement by # rules

The results in Table 6.6 allow us to answer an unaddressed question posed earlier in this study:

- How well does the rule generation process adapt to weighted data?

To answer this question, we focus on the % increases in accuracy, shown graphically in Figures 6.4 and 6.5. The degree of improvement serves as the primary indicator for learner adaptability, as each new instance which is correctly classified indicates a successful adaptation of the new learner to changing weights. The degree of success is varied by dataset, but overall the results are positive, and show marked improvement. This provides confidence that the rule generation procedure adapts well to changing data weights and affirms that this aspect of our study is a success.

We can also address queries specific to the experimental proceedure:

- How does the number of rules per learner effect performance?
- Does the number of classes have a visible effect?

Figures 6.4 and 6.5 provide key insights, allowing us to answer these questions through visual analysis. By comparing the degree improvement with the number of rules for each dataset, we can make some useful observations. While we would expect intuitively that weaker learners (those with fewer fuzzy rules) would have more room for improvement via the boosting mechanism (and hence have higher degrees of improvement) the results do not support this intuition. Visually, we observe that those ensembles with the most improvement are not obviously correlated with the number of rules, preventing us from making conclusions on this matter. This pattern (or lack thereof) is likely related to how well a certain number of rules fits a given dataset's internal structure, among other factors, and suggests that the interactions of weak learners in a boosted ensemble are complex.

Returning to Table 6.6, we identify Wine Quality, Heart Disease, and Magic to be three datasets with higher dimensionality, and Glass to be the dataset with many classes. We see no clear pattern with respect to improvement and dimensionality, as two results (Heart Disease, Wine Quality) show moderate improvement, and one (Magic) shows minimal improvement. This indicates that the specifics of a problem are a larger factor than the dimensionality of the dataset. With only one dataset having a large number of classes, and with the experiments showing substantial improvement (as shown in Figure 6.4 and 6.5), we can only conclude that the number classes are not hugely detrimental to our methodology. Generally, we conclude that while dataset characteristics play a role in the quality of individual component learners, the weaknesses of the component learners seem to be readily addressed through boosting.

6.3.3 Comparative Studies

In this final set of experimental studies, we compare the performance of the proposed methodology to equivalent boosted ensembles composed of simple established weak learners. These results provide meaningful insight into the relative performance of our methodology, contrasted against what are considered standard learners for use in a boosted ensemble.

These comparative experiments are performed with existing machine learning tools, namely MATLAB and WEKA. Five different weak learners are utilized in the generation of boosted ensembles – a decision tree, a decision stub, the OneR rule procedure, the ZeroR rule algorithm, and the NaiveBayes classification method. Decision tree experiments are executed using the MATLAB machine learning toolbox, and all other experiments make use of WEKA's built in functionality. The learner generation procedures are all built in and readily available components of the software. To maintain consistency with our previous experiments, all comparative studies are evaluated with 10-fold cross-validation, and each comparative experiment considers an ensemble size of 10. The learning rate is set to 0.1, as with the fuzzy rule boosting experiments. When working with MATLAB the AdaBoost.M1 and AdaBoost.M2 algorithms are used for two-class and multi-class problems respectively. For those experiments performed using WEKA, AdaBoost.M1 is used for binary classification and LogitBoost is used for multi-class problems.

To further expand the scope of our experiments, we consider some additional datasets:

- *Bands* – A classification problem from rotogravure printing, the task is to determine a given piece is a cylinder band
- *Cleveland* – A specific difficult subset of the heart disease dataset
- *Coil 2000* – An insurance company benchmark problem
- *Phoneme* – Predict if a sound is nasal or oral from vowel sounds
- *Pima* – Data concerning diabetes diagnosis predicted based on certain medical measurements
- *Yeast* – Categorizing yeast species from test results

All of which are available from the KEEL repository. These additions target the inclusion of larger problems (more instances) as well as more multi-class problems.

Table 6.7: Comparative Boosting Results

Dataset	**Weak Learner**					
	FR-Boosting	**Decision Tree**	**Decision Stub**	**Naïve Bayes**	**OneR**	**ZeroR**
Wdbc	0.931	0.946	0.951	0.956	0.916	0.627
Bands	0.594	0.588	0.714	0.627	0.679	0.579
Cleveland	**0.565**	0.555	**0.576**	-	0.512	0.539
Coil2000	0.940	0.940	0.940	0.907	0.939	0.940

Phoneme	**0.776**	**0.799**	0.777	0.760	0.753	0.707
Pima	0.703	0.755	0.757	0.757	0.697	0.651
Iris	**0.977**	0.887	0.953	0.933	0.927	0.333
Wine-Quality	0.810	0.854	0.983	-	0.882	0.399
Wisconsin	0.978	0.949	1.000	0.994	1.000	0.650
Heart Disease	0.718	0.796	0.804	0.830	0.719	0.556
Appendicitis	**0.900**	**0.868**	0.830	0.840	0.868	0.802
Glass	0.625	0.687	0.715	-	0.533	0.355
Bupa	0.614	0.677	0.670	0.638	0.588	0.580
Banknote Auth	**0.968**	0.962	0.945	**0.983**	0.896	0.555
Yeast	0.492	0.552	0.617	-	0.379	0.312

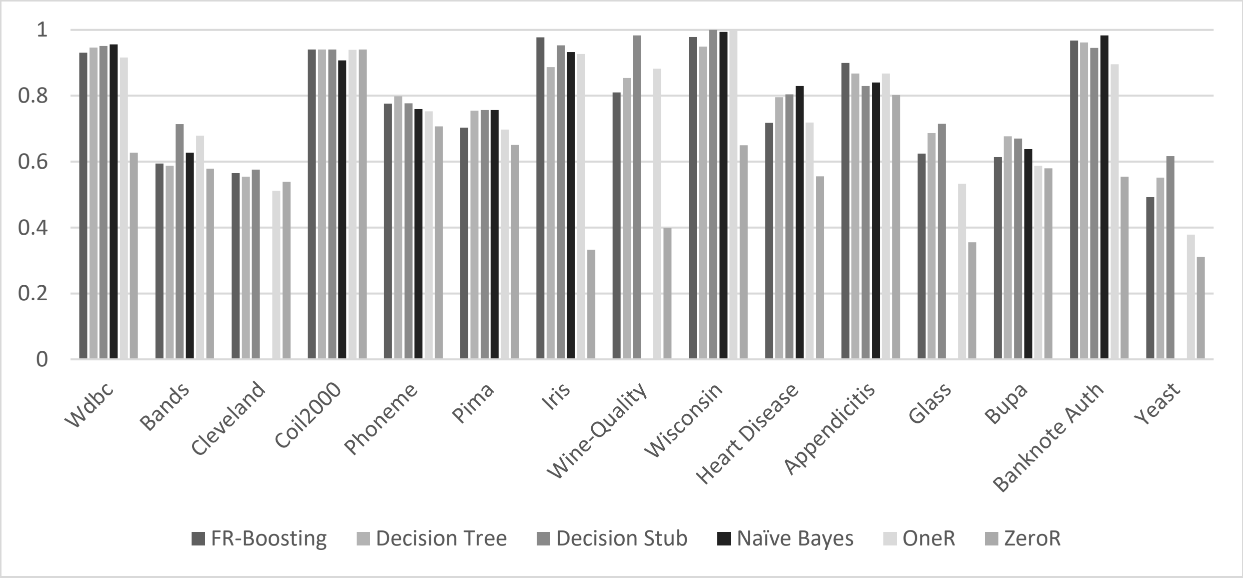


Figure 6.6: Boosting performance for various weak learners

Table 6.7 and Figure 6.6 present the results of our comparative boosting experiments for all considered datasets.

In multi-class experimentation, the NaiveBayes learner is incompatible with the boosting algorithm provided by WEKA, so results are not applicable. These experiments allow us to address the final question posed at the beginning of this study:

- How well are fuzzy classification rules able to perform?

The experiments in Table 6.7 allow us to address this question by directly comparing the predictive performance of the proposed methodology to standard weak learners. The results demonstrate that our methodology is generally on par in terms of classification accuracy, with a few results swinging to either side of the spectrum. In a couple cases, Iris and Appendicitis, our methodology outperforms the all other learners by a small amount. In other cases, Glass, Heart Disease, Yeast and Bands, our learner is noticeably poorer than the top performing experiments. This is not too concerning, as we readily accept that different learners will have different strengths and weaknesses which are still apparent in a boosted ensemble.

We have included the *ZeroR* "learner" in these experiments to help analyze the meaningfulness of certain results. This algorithm is a dummy learner which simply predicts the majority class (the class with the largest number of instances) 100% of the time (or the *weighted* majority class when part of a boosted ensemble). This provides some context to certain results, such as Coil2000, where no predictor provided meaningful insight into the problem, and some learners, such as Naïve Bayes, performed strictly worse than this simplistic method. Additionally, we highlight the Cleveland dataset, where all three compatible algorithms barely outperformed *ZeroR,* indicating that very little knowledge was extracted from the system by any learner. Examining the Appendicitis dataset, the performance of the proposed algorithm exhibits what appears to be a small increase in classification accuracy; however, in the context of *ZeroR* accuracy, we can assert that this performance increase is notably more significant.

A handful of interesting results have been highlighted in Table 6.7 (in bold) where the proposed method performed well with respect to other learners.

6.3.4 Final Remarks

This chapter proposes a methodology for the generation of a novel fuzzy weak learner, in the form of fuzzy classification rules, for use in a boosted ensemble. The goal of this research is to explore the feasibility of applying a fuzzy rule-based system as the component weak learner of a boosted ensemble, and to study the adaptability of such systems in a weighted environment. The topic of fuzzy models in boosted ensembles has not been well-studied, and we present this specific methodology as an initial case-study on the topic.

Through our experimentation, we have asserted the feasibility of the proposed methodology. Through insightful case studies, we have demonstrated the proposed learner's ability to improve classification performance in an ensemble and shown the effects of the sigmoidal modifier on classification performance and model adaptability. Initial boosting experiments demonstrated the improved classification rates achieved by the ensemble as compared to individual classifiers, and further experiments compared the performance of the fuzzy rule ensemble to established weak learners, with the proposed methodology generally performing at a reasonable rate.

We consider this study to be a successful foray into the topic of boosting with fuzzy models. This study is an initial exploration of this topic, as we have only addressed a single case study in what could be an expansive research topic on generalized boosting with fuzzy models. There is ample opportunity for further research, both considering the refinement of our methodology and the application of completely different weak fuzzy learners to boosted ensembles. As the adaptability and diversity of the component learners is critical to successful boosting, future research could consider different avenues for the expansion of learner flexibility, as well as more sophisticated methodologies for classification rule extraction from data.

7 Generating Hierarchical fuzzy rule based models from Data

So far, we have examined fuzzy rules from a number of different perspectives, including the analysis of their stability, their use as weak learners in an ensemble, and the exploration of different generation methodologies and rule formats. The commonality in the presented work is that each chapter is concerned with a traditional, flat, fuzzy rule-based system. This standardized form of FRBS is well-studied and established; however, the modification of the standard form provides substantial opportunity for novel research and potential avenues for improvement.

In this chapter, we propose a novel fuzzy rule-based architecture, defining a hierarchical rule structure in a cascading format. The goals of this research are twofold. First, we seek to improve the predictive performance of the fuzzy model, as compared to the traditional format, through increased rule specificity and division of labor. Second, we target the readability and interpretability of the fuzzy model through the formation of simpler individual rules, which are more easily understood by a human reader. Improving the performance of FRBSs has been addressed extensively in the literature, including many studies proposing novel rule extraction techniques aimed at reducing error. For the most part, research in this area has focused on improving the quality of individual rules through advanced extraction techniques, but with the final rule architecture remaining unchanged from the well-known form. Although less common, studies proposing novel fuzzy rule structures or system architecture do exist. These sophisticated architectures can be used to improve the predictive ability of the model, while still maintaining a high degree of interpretability. Focusing on the topic of this study, we are interested in hierarchical fuzzy rule-based models; specifically, those model topologies which divide the predictive responsibility between multiple layers of fuzzy rules. The literature on this topic is limited, and the existing relevant studies are discussed in detail in Section 2.4.

In this chapter, we propose a hierarchical rule-based architecture and a companion methodology for the extraction of the defined architecture from data. In the proposed format, we apply the predictions of the previous hierarchical layer in the output part of the fuzzy rules. We consider a cascading style of hierarchical structure, where lower level rules serve to refine or fine-tune those coarser predictions made at higher levels of the model. Additionally, we explore strategies for the selection of critical modeling parameters, including the number of fuzzy rules to compute at each level of the hierarchy, and the selection of which features to use at which levels of the hierarchy. These parameters are critical to the successful generation of a high-quality hierarchical model, so their intelligent selection is important. We provide extensive experimental results demonstrating the feasibility of the proposed architecture, case studies outlining the behavior of parameter selection techniques, and some comprehensive experiments showing the relative performance of the proposed methodology versus a standard TS-FRBS.

The work presented in this chapter exhibits several aspects of novelty. First, a complete methodology for the autonomous extraction of a hierarchical FRBS from data is fully novel, as all previous studies have only considered expert generated systems and have not addressed rule extraction from data. Second, the use of feature selection strategies and making autonomous choices regarding the number of rules at each level has not been addressed in previous works. Finally, the existing literature on hierarchical fuzzy models is primarily concerned with the reduction of the number of required fuzzy rules, with a focus on interpretability. This study also demonstrates the high degree of interpretability attained by the proposed format, but additionally demonstrates that, in many cases, we are able to attain significant performance improvement using a hierarchical scheme.

7.1 ARCHITECTURE OF THE HIERARCHICAL FUZZY RULE-BASED MODEL

This section outlines the architecture of the proposed hierarchical fuzzy rule-base, as well as related computations pertaining to feature selection and choosing the number of rules. The model format proposed in this study is constructed in an iterative fashion, and lower levels of the hierarchy are incrementally computed and added to the architecture as training proceeds.

7.1.1 Overview of the Model Architecture

The proposed architecture takes the form of a cascading hierarchical structure. At each level of this structure, some number of input features are used to generate a set of first order TS-fuzzy rules. At the second through *nth* level of the model, the predictions made at the previous layers are considered as a feature in the consequent part of the fuzzy rules. The generalized architecture takes the form shown in Figure 7.1, where it is important to note that the predictions of the previous layer are only applied to the output parts of the subsequent rules. In this format, lower level rules act to fine-tune higher level predictions, using new feature knowledge to construct increasingly specific fuzzy rules. The number of features used at each level is fully configurable but should be kept low to ensure rule specificity.

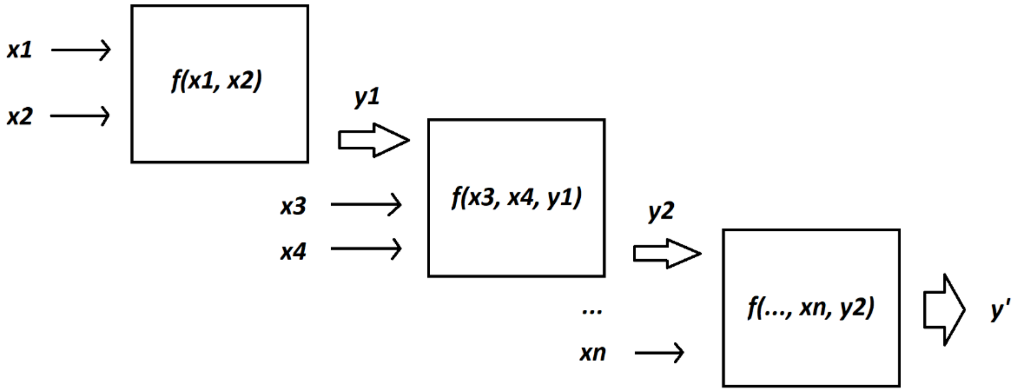


Figure 7.1: Generalized architecture of the hierarchical fuzzy rule-based model

In this format, the contribution of the previous layer is used in the output part of the next layer:

$$y(t) = b_0 y(t-1) + f(\boldsymbol{x_t}, b) \tag{7.1}$$

Where *t* is a given level of the hierarchy for *t* > *1*, b_0 is a functional parameter specific to the previous layers output prediction, $\boldsymbol{x_t}$ is the vector of features used at level *t*, and *b* is the additional set (array) of functional parameters associated with the relevant input features at this layer. This is a modified version of a standard first order TS-fuzzy rule as given in (3.8). As a simple example, consider a hierarchical level where we have selected two features to model, $\boldsymbol{x_s}$ and $\boldsymbol{x_d}$. The consequent of a given rule at this level would take the following format:

$$y_{i,t} = b_0 y_{t-1} + b_1 x_s + b_2 x_d \tag{7.2}$$

Where $y_{i,t}$ is the output of the *ith* rule at the *tth* level.

In Section 3.4.1, we define a methodology for estimation of functional parameters for a first order TS-FRBS using equations (3.13) through (3.18). With the addition of the previous layer's prediction, the problem needs to be reformulated. The equation (3.13) is modified to:

$$y_t = \boldsymbol{b_i}^{\boldsymbol{T}} [\boldsymbol{x_k}, y_{t-1}] \tag{7.3}$$

And (3.15) is similarly changed:

$$\boldsymbol{z_{ik}} = u_{ik}[\boldsymbol{x}_k \,, y_{t-1}]$$

(7.4)

With these modifications, we now consider the prediction from a previous layer to act as the y-intercept of the multi-dimensional linear equation for each rule. This is an intuitive way to handle this problem, as the linear equations for each rule effectively treat the last prediction as a starting point, and the functional parameters associated with the new input features are used to adjust this initial value as described by their relationship to the output feature. This is in line with our previous description of the hierarchical architecture, using lower layers to continuously fine tune the prediction, hopefully resulting in overall better performance.

The prediction of each layer is similar to the standard equation, previously defined in (3.11) with the addition of the previous layer's prediction:

$$y_t = \sum_{i=1}^{c} u_{ik} y_{i,t}$$

(7.5)

Where $y_{i,t}$ is given in (7.2), and u_{ik} is the fuzzy membership of the instance to the rule.

7.1.2 Interpretability of the hierarchical model

This section briefly demonstrates how the described architecture of hierarchical fuzzy-rules can be "unfolded" into rules similar to a flat rule-base (if desired), and how the hierarchical architecture can improve overall interpretability by decreasing the complexity of each individual rule, even though more rules are generated and activated in total.

The hierarchical architecture can have the activated rules from each level combined into a single larger rule if desired. The flattened output equation for the activated rules would look something like the following:

$$\hat{y} = u_L b_{L-1,0}\left(\dots\left(u_2 b_{3,0}\left(u_1 b_{2,0}\left(b_{1,1}x_1 + b_{1,2}x_2\right) + b_{2,1}x_3 + b_{2,2}x_4\right)\right) + \cdots\right) + b_{n,1}x_{n-1} + b_{n,2}x_n$$

(7.6)

Where u_l is the membership of the given input instance to the activated rule at the *lth* level, $b_{a,b}$ is the *bth* functional parameter for the rule activated at the *ath* level, $\boldsymbol{x_i}$, i= 1, 2, … n are the n input features in the order in which they are selected, and L is the number of hierarchical levels. In a real-world application, u

values are constant for a given input instance and b values are static once the model is trained; hence, the equation is significantly simplified as these numeric values are easily reduced algebraically. Obviously, in a fuzzy environment this does not tell the full story, as multiple rules may be activated at each level to different degrees; however, for the purposes of presenting the highly activated rules to provide human readable feedback, this format is useful.

A similar equation can be formulated to provide a flat representation of the activated input parts of the rule:

$$IF\ x_1\ is\ A_{i1}\ AND\ x_2\ is\ A_{i2}\ AND\ \ldots AND\ x_n\ is\ A_{in}\ THEN\ \ldots \tag{7.7}$$

Where $\boldsymbol{x_i}$ are the features in their chosen order for the hierarchical model, and A_{ij} indicates the fuzzy set associated with feature j at rule i, for the activated rule at a given level. Again, there is the potential for multiple rule activations at each level, but for human readability the transparency is still present.

The other aspect of model interpretability which may be of interest is how the division of otherwise complex fuzzy rules from a flat topology into a series of simpler rules is beneficial to human readability. In a traditional flat fuzzy model containing some c fuzzy rules, each considering n input features, it can often be difficult for a human reader to assess the effect of individual features on system behavior, especially when considering linear rule consequents. In our hierarchical topology, the individual rules at each level are significantly simpler, and the number of rules at each level can be tailored to the feature(s) under consideration. This improves model interpretability, both by making the influence of individual features clearer and by making rules easier to digest (as they are less complex).

7.1.3 Features selection in a Hierarchical Model

The hierarchical architecture defined in the previous section is computed incrementally, adding a new layer using new input features to compute new fuzzy rules. This naturally raises the question of which features should be used at what levels of the hierarchy, and how we make this decision.

The general question of feature selection is one which has received significant research attention, although most studies focus on classification problems and the elimination of less useful features from the modelling process. That being said, feature selection for continuous problems does exist, with a few notable strategies. The goal of feature selection is to choose the best features from those available in the dataset according to some metric, usually in order to eliminate less useful features simplifying the model or improving performance. In our case, we use feature selection to determine the order in which features are modelled by the hierarchical architecture, and because this choice is made as a step in an iterative process, we must be cognizant of the computational cost of each potential solution. We should be especially cognizant of any overly complex feature selection strategies, as for a small number of features they may be outperformed by simple enumeration. For the purposes of this study, we consider two feature selection strategies: the use of a correlation coefficient and a performance-based strategy.

In the proposed methodology, we are interested in the selection of a small number of features at each iteration. As such, we first consider the extremely simple strategy of enumerating of the available features, and the selecting the subset of features which results in the best immediate performance. This strategy has some obvious strengths and weaknesses. First, we note that when evaluating our options through enumeration, the complexity of the computation is tied to the total number of features and the number selected, and that as the number of desired features increases, this complexity increases exponentially. Further, we expect that the previous features selected have an impact on the choice of the next features, so the evaluation needs to be made for each remaining feature at each iteration of the rule extraction process. This results in an overall relatively high computational cost, which may negatively affect modelling large or highly dimensional datasets. On the other hand, this selection strategy is very simple, intuitive to understand, and provides some guarantee of performance-based feature selection. This is an example of greedy strategy, where the best immediate choice is always taken.

The second strategy we consider is the use of a simple statistical measure, such as correlation, to select features at each level of the hierarchy. A correlation coefficient can be used to select those features with the strongest linear correlation to the output feature first, and then proceed to fine tune the model with less and less meaningfully correlated features. This strategy has a degree of logical justification, as the fuzzy rules generated at each level are first order TS-fuzzy rules, and contain linear consequents; hence, we would expect those features with a high degree of correlation to produce stronger initial predictions, enabling lower level rules to fulfil their role as fine tuning components. This strategy is not without its drawbacks. A correlation coefficient fails to capture localized linear relationships, which are easily modelled by a set of first order fuzzy rules, and there are other aspects to a features desirability in modelling which are not captured by linear correlation. Regardless, a correlation coefficient is significantly more computationally efficient than the performance strategy discussed previously, and the coefficient values can be calculated once at the beginning of the modelling process, and do not need to be recalculated during iterative model construction.

We consider the Pearson’s Correlation Coefficient, *r,* defined by the following equation:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

(7.8)

Where $\bar{x}$ and $\bar{y}$ are the sample means for the two features under consideration, and the computed value, *r,* is the sample correlation coefficient between the two features. Higher *r* values indicate a higher degree of correlation, so when considering the correlation between an input and output variable a higher value is considered to be a better immediate choice.

7.1.4 Determining the Number of Rules

An important parameter in fuzzy rule generation, when using FCM clustering, is the choice of the number of clusters (rules), denoted c. The selection of this parameter has critical implications regarding the performance of the model, as well as the quality of the resulting knowledge. There several important considerations when selecting *c,* including the identification of the "natural" number of rules for a given dataset or feature, the trade-off between performance and memorization, and the complexity (interpretability) of the resulting model.

In a more complex model topology, such as the hierarchical structure proposed in this chapter, the choice of *c* gains additional importance, as the selection of this parameter is made at each level of the hierarchy. This means that good or bad choices of *c* have far reaching implications on the quality of the model as the effects compound over the course of model construction. Additionally, as features are considered independently or in small sets, we are provided with an opportunity to make feature specific parameter selection, which may improve overall model quality.

Cluster validity indices are a topic which has received significant attention in the existing literature, although few of these studies are recent. Such indices perform some computation on a clustering result (in our case fuzzy clustering) and return a numerical indication of cluster quality. The evaluation of cluster quality is often computed considering factors such as data representation, cluster compactness, and cluster separation. The index is computed for a range of potential *c* values, and the results are compared to determine the "best choice" from the evaluated options. There are a large number of cluster validity indices available, and one index which has shown good overall performance in the literature [96] is the Xie-Beni index [97], defined previously in (3.25). This index considers a combination of data belongingness and inter-cluster separation to give an indication of the feasibility for a given choice of c.

In our experiments, we make use of this index by computing V_{xb} for a range of potential *c* values at each level of the hierarch and using this information to make autonomous choices of *c* during hierarchical model construction.

A second option for the autonomous selection of c is the use of a validation (testing) dataset during training, which enables us to select c based on balancing training and testing accuracies, with the goal of maximizing performance while avoiding memorization. This strategy focuses on finding the value of c at which training performance is maximized, but the difference in error for the validation set is minimized. To accomplish this, we consider two values. First, the difference between the training and testing errors indicates the degree of overtraining, and we target the minimization of this value. Second, we wish to minimize the training error (maximize performance), and we address both these criteria simultaneously through the following index:

$$V_{validation} = E_{train}|E_{train} - E_{test}| \tag{7.9}$$

Where E_{train} and E_{test} are the training and testing RMSE's respectively. We employ this index in a similar manner to a standard cluster validity index, evaluating different choices of c, and selecting the best choice based on the lowest score.

7.2 EXPERIMENTS

This section provides several experimental results covering a range of exploratory case-studies on important parameterizations, and extensive comparative experiments demonstrating the performance of the proposed methodology.

An aspect of the hierarchical FRBS which has not yet been discussed is the choice of how many features should be used at each level. In our experiments, we consider the simplest case, using two features at the first level and one additional feature at all subsequent levels. This maximizes the degree of hierarchy, with the aim of promoting any apparent effects from this structure.

7.2.1 Case Study – Improvement through hierarchy

As an initial case-study, we examine the ability of the hierarchical topology to continuously improve modelling performance as additional levels (features) are added. It is imperative that we establish improvement through additional hierarchical levels, as this is a major justification of the proposed methodology. Additionally, we may find experimentally that not all features contribute positively to predictive performance, or that using too many features results in memorization effects.

For these experiments we use two real-world datasets from the KEEL repository [197]:

- **Forest Fires** – A data set which uses weather metrics to predict the size of a forest fire burn zone. This dataset has been modified to remove date information for compatibility reasons.
- **Concrete** – A problem which uses chemical and physical measurements from concrete samples to predict compressive strength.

Our first set of experiments briefly visualizes the error rates of a model over the course of hierarchical construction. Error values given here indicate the performance of the model if generation was stopped at this iteration. Each dataset is split into a training and testing partition with a 75/25% split. A small range of static *c* values are tested in the range of 3 to 10, and we use the performance-based feature selection strategy from Section 7.1.3.

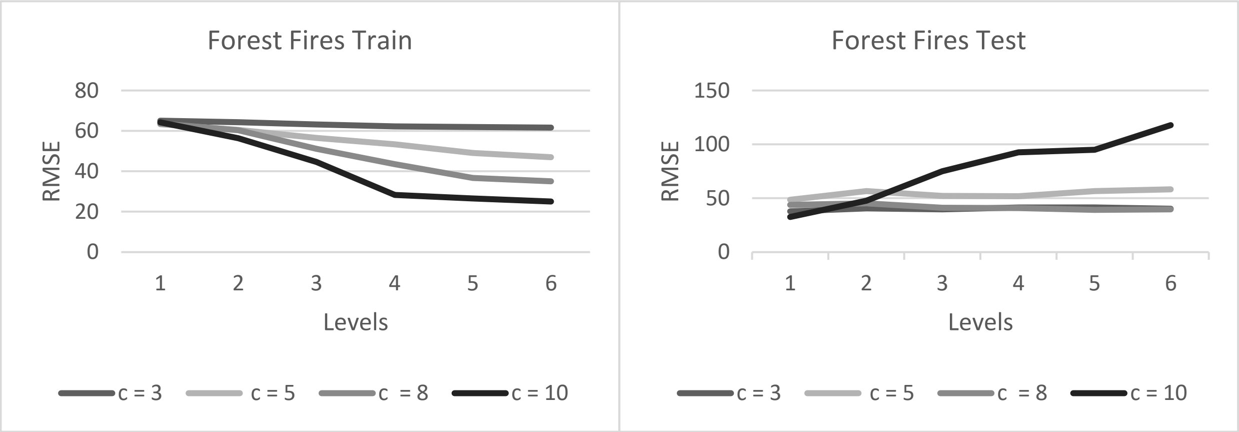


Figure 7.2: Training and Testing errors for the forest fire data set vs number of hierarchical levels

Figure 7.2 provides RMSE's for the forest fires dataset. As shown graphically, over the course of model construction the training error is consistently decreased by the addition of an additional feature (level). The degree to which the model is improved varies, and in certain application it may be beneficial to stop

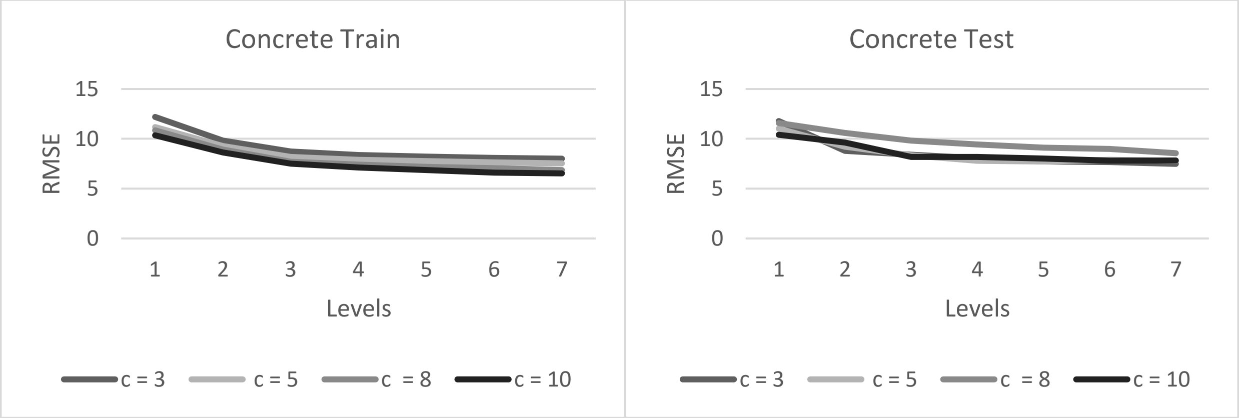


Figure 7.3: Training and Testing errors for the Concrete data set vs number of hierarchical levels

construction early to generate a simpler model. The testing figure indicates that much of this improvement may come at the cost of significant overtraining, as in some cases, particularly *c = 10*, the testing error rises rapidly with the number of levels. Other cases, such as *c = 8,* show that memorization effects can be avoided, while still significantly increasing model performance when good parameterizations are used.

Figure 7.3 provides a similar graphic for the Concrete dataset, generated using the same experimental setup. This experiment results in a standard curve shape for RMSE as model complexity is added, this time with a far smaller degree of memorization. Note that the choice of *c* remains critical as the degree of overfitting varies with respect to this parameter.

7.2.2 Case study – feature selection strategies

Next, we examine case studies exploring the behavior of the proposed feature selection strategies. The considered strategies are those outline in Section 7.1.3, and all experiments are performed with 10-fold cross validation.

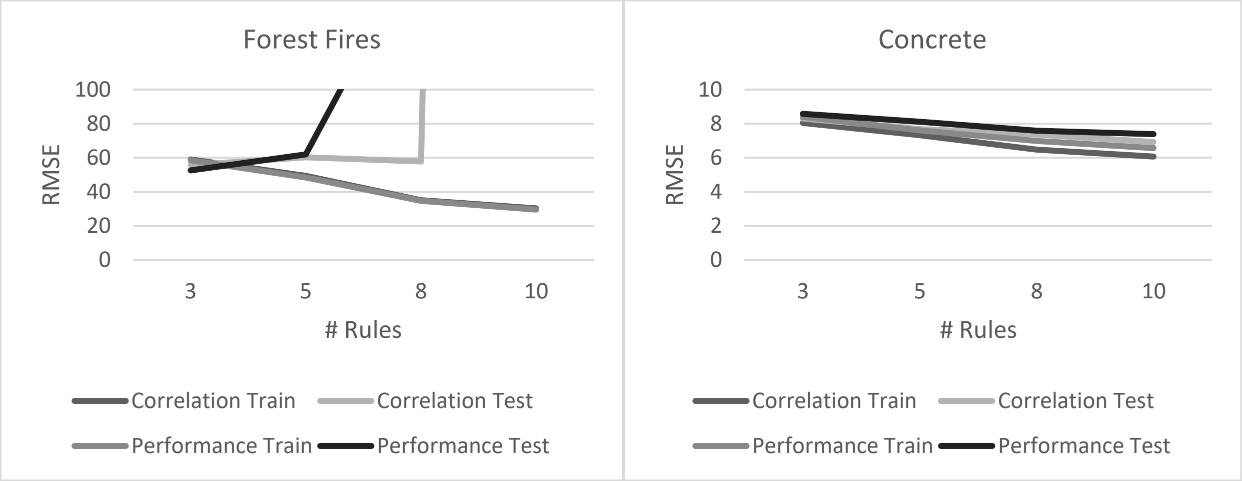


Figure 7.4: Training and testing errors for 3 sample datasets for different feature selection strategies

Figure 7.4 graphically presents the performance of the hierarchical method using two features selection techniques, each evaluated for a small set of *c* values. We observe that the feature selection strategy does not noticeably impact the performance of the final models. In fact, looking at the concrete dataset, the simpler correlation strategy slightly outperforms the performance strategy. This demonstrates that a greedy strategy may not always be the best choice. Regarding the forest fires plot, huge memorization effects are visible as *c* is increased, making analysis somewhat difficult; however, we note that this spike occurs at a smaller *c* value using the performance method (*c* = 8) than with correlation (*c* = 10).

These case studies indicate that there does not seem to be a substantial difference between feature selection strategies, but that, on the balance of the results, the correlation coefficient seems a slightly more stable choice, with the additional benefit of a lower computational cost. Further experiments, or more sophisticated strategies may be able to improve the model to a greater extent than observed in this section.

7.2.3 Case study – tuning the number of rules

As alluded to many times in previous sections, the choice of *c* at each level of the hierarchy is critical to high quality modelling. Section 7.1.4 outlines the candidate strategies for determining the value of this parameter automatically, and in this section, we present experiments to examine the effects of each strategy on the quality of the completed model. All experiments are conducted with 10-fold cross validation and use the standard choice of $m = 2.0$.

Figure 7.5 displays the graphical results using a static *c* value at each level, varying from $c = 2$ through 10. As shown in Section 7.2.2, larger *c* values frequently result in overtraining, and this effect is most visible in the Forest Fire experiments. Using a static *c* is simplistic and is the result of uninformed choices of *c* at each level.

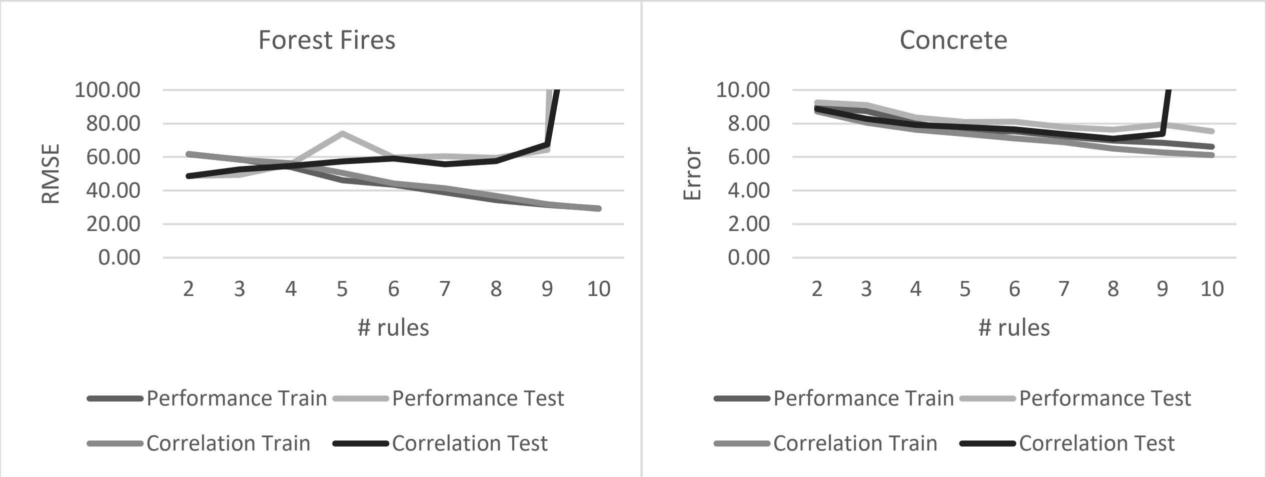


Figure 7.5: Static # rules for example datasets, training and testing data for two feature selection strategies

Next, we examine the use of a cluster validation index, specially the Xie-Beni index as defined in (3.25).

Table 7.1: RMSE when choosing c by validation index

Data Set	Performance		Correlation	
	Train	Test	Train	Test
Forest Fires	54.70	51.42	53.77	52.25
Concrete	7.44	8.06	7.37	7.86

Table 7.1 provides a brief set of reported RMSE's selecting *c* at each level using $V_{XB.}$ First, we observe that this strategy seems to avoid significant memorization effects automatically, with training and testing errors remaining similar throughout. The fact that this strategy may automatically avoid overtraining without external expertise is a major point in its favor.

Table 7.2: Sample results for static c values

Data Set	**# rules**	**Performance**		**Correlation**	
		Train	**Test**	**Train**	**Test**
Forest Fires	4	53.89	55.82	56.28	54.70
Concrete	5	7.52	8.09	7.39	7.78

Table 7.2 provides sample errors for those static *c* values most closely matching the cluster validation results for comparison purposes. With respect to the Forest Fires dataset, the validation index performs slightly better on average. In the case of the Concrete dataset, we would say that the results are very similar between the two cases, with the primary advantage to the validation index being that no prior knowledge of the problem was needed.

Finally, we examine the use of a validation dataset. For this strategy the fitness of a given *c* value is expressed as a function of testing and training errors, as defined in (7.9).

Table 7.3: Errors for choosing c using a validation dataset

Data Set	**Performance**		**Correlation**	
	Train	**Test**	**Train**	**Test**
Forest Fires	38.92	56.70	38.15	57.04
Concrete	8.13	8.49	7.19	7.65

Analyzing the results in Table 7.3, we compare these errors to those from the previously examined strategies. With respect to the forest fires dataset, we note that this strategy results in a greater degree of memorization, but the that *overfitted* testing errors are comparable with those from previous strategies. The concrete dataset results in larger errors when using performance to choose features, and lower errors when considering the correlation coefficient. This implies a potential interaction between the two parameter selection strategies.

These case-studies demonstrate the feasibility of both automatic parameter selection strategies proposed in this study. These experiments indicate that making an autonomous choice of *c* at each level results in higher quality models than a static value, in addition to eliminating an otherwise necessary input parameter. These experiments do not provide definitive evidence in favor of one method or the other but do provide some useful feedback on each strategy.

7.2.4 Comparative experimental results

Previous experimental subsections provide useful case studies exploring the proposed options for feature and *c* selection strategies. The case studies considered allow us to move forward with more comprehensive experiments, using a larger number of real-world datasets. In this section, we provide extensive experimental results which demonstrate the quality of the hierarchical fuzzy models generated using the proposed methodology. To establish a relative degree of performance, we record the error rates of the proposed architecture alongside the error rates of an equivalent flat FRBS, as a baseline.

In addition to the forest fire and concrete datasets considered previously, we provide experimental results for the following datasets from the KEEL repository [197].

- **Abalone** – Predict the age of an Abalone from physical measurements
- **AutoMPG** – Predict the fuel efficiency of an automobile from make and model information
- **Baseball** – Uses baseball statistics to predict the salary of players
- **Compactiv** – Determine user CPU usage percentage from other computational information
- **Pole** – A telecommunications problem
- **Treasury** – Determine the monthly CD rate from various financial data
- **Wizmir** – Various weather data is used to predict a mean temperature
- **Friedman** – A synthetic benchmark dataset
- **Mortgage** – Predict the conventional mortgage rate from financial data
- **Wankara** – Predict the mean temperature of a region from weather information
- **California** – Housing price prediction based on location and house features in California
- **House** – House price prediction based on house features and location in the United States
- **Puma32h** – A dataset concerned with the control systems of a robot arm

This represents a comprehensive set of regression problems with a wide range of problem types, dataset sizes, and dimensionalities.

To demonstrate any statistical significance in error differences between the hierarchical and flat fuzzy models, we perform an unpaired T-Test on the resulting RMSE’s from 10-folds of cross validation and report the resulting *p* values alongside the results. When analyzing *p* to determine significance, we consider a standard threshold of $p < 0.05$ to indicate a significant difference between the errors.

Due to the number of considered datasets, and the complexity of the parameterization choices, comparative experiments are split into two sections. Initial comparative results are presented using a static *c*, and a reduced result set promoting the best performing experiments from the proposed *c* and feature selection strategies is given afterwards.

Table 7.4: Comparative Errors for Hierarchical vs Standard FRBS

		Hierarchical		**Flat**		**T-Tests**	
Data Set	**c**	**Train**	**Test**	**Train**	**Test**	**p train**	**p test**
Abalone	5	*2.14*	2.17	2.09	2.12	0.000	0.496
AutoMPG6	5	2.77	2.98	2.78	3.01	0.637	0.936
Baseball	5	**721**	1049	772	1042	0.000	0.947
Concrete	5	**7.54**	**8.03**	9.31	9.79	0.000	0.000
Compactiv	5	**2.91**	**2.99**	4.41	4.56	0.000	0.000
Pole	5	**14.3**	14.4	18.2	1679.0	0.000	0.113
Treasury	5	*0.206*	0.221	0.181	0.203	0.000	0.179
Wizmir	5	*1.14*	1.19	1.10	1.15	0.000	0.437
Friedman	5	1.80	**1.86**	1.80	117.67	0.913	0.029
Mortgage	5	*0.099*	*0.105*	0.071	0.080	0.000	0.000
Wankara	5	*1.30*	*1.679*	1.14	1.39	0.000	0.001
California	5	*72482*	*72662*	66700	66976	0.000	0.000
House	5	**37904**	38348	43092	704559	0.000	0.088
Puma32h	5	**0.024**	**0.025**	0.026	0.712	0.000	0.000
Forest Fires	5	**46.3**	59.4	61.0	52.0	0.000	0.699

Table 7.4 contains a comprehensive list of comparative experiments using a static *c* value of $c = 5$, chosen as a reasonable general choice given the results of past experimentation. Bolded entries indicate a superior performance and italicized entries indicate a weaker performance of the hierarchical method with respect to the flat topology. Over this large set of regression datasets, we observe generally mixed results. In many instances, the hierarchical topology outperforms the flat topology; however, there are a few instances of equivalent performance, and some cases where the proposed hierarchical scheme is outperformed by a flat FRBS.

Looking first at the positive results, we identify the Concrete, Compactiv, Pole, House, Puma32h, and Forest Fires datasets. In these experiments, the resulting RMSE's and *p* values demonstrate a statistically significant improvement in modelling performance as compared to a flat FRBS, sometimes with a relatively large decrease in error. Those datasets for which the hierarchical topology resulted in equivalent error rates to the flat FRBS include the AutoMPG and Friedman datasets, where, as demonstrated by the t-tests, any differences between errors are not considered significant. Finally, we note that the Abalone, Treasury, Wizmir, Mortgage, Wankara and California datasets show a decrease in modelling performance when compared to a standard FRBS. Note that this is only for the case of a static *c* value.

For some of these datasets (specifically Baseball, Pole, Abalone, Treasury, Wizmir, House, and Forest Fires) the differences between reported RMSE's are only significant for the training partition. Explanations for this behavior vary. In the case of the Pole dataset, we note that the flat model suffers from significant overtraining, potentially causing chaotic behavior, and similar characteristics are observed for the House and Puma32h datasets. In other cases, the specific reasons for poor testing performance are less obvious, but it would seem a fair assumption that certain models demonstrate significant instability between folds, destabilizing the t-test results.

As a final set of experiments, we provide a reduced set of comparative results using the feature and *c* selection strategies, continuing to use of 10-fold cross validation. In the cases where a given *c* selection strategy is used in the hierarchical case, the identical strategy is used to choose the number of rules in the flat model. While this may not always result in directly analogous models, depending on the values chosen, it seems the fairest strategy as both models are constructed and evaluated on identical objective criteria.

Table 7.5: Highlighted Comparative results using feature and rule selection strategies

Dataset	**Parameterizations**		**Hierarchical**		**Flat**		**T-Tests**	
	Features	**Rules**	**Train**	**Test**	**Train**	**Test**	**p train**	**p test**
Abalone	Performance	Index	2.13	2.17	2.15	2.19	0.080	0.859
Auto MPG	Correlation	Index	**2.65**	2.93	2.79	3.03	0.001	0.639
Baseball	Correlation	Validation	736	945	785	879	0.157	0.323
Concrete	Correlation	Validation	**7.30**	**7.64**	8.74	9.18	0.001	0.000
Compactiv	Performance	Validation	**3.08**	**3.58**	4.94	4.99	0.000	0.003
Pole	Performance	Index	**13.16**	**13.22**	23.69	23.83	0.000	0.000
Treasury	Performance	Validation	0.19	0.22	0.19	0.20	0.651	0.464
Wizmir	Correlation	Validation	1.12	1.25	1.09	1.14	0.096	0.153
Friedman	Performance	Index	**1.77**	**1.87**	2.15	3.78	0.000	0.014
Mortgage	Performance	Validation	*0.09*	*0.10*	0.08	0.08	0.039	0.028

Wankara	Performance	Validation	1.19	*1.78*	1.18	1.28	0.545	0.003
California	Correlation	Validation	*72713*	*72815*	66517	66806	0.000	0.000
House	Correlation	Validation	**37600**	**38419**	44627	52970	0.000	0.048
Puma32h	Correlation	Index	**0.01**	**0.01**	0.03	582672	0.000	0.000
Forest Fires	Performance	Index	54.51	50.72	62.44	48.06	0.056	0.896

Table 7.5 presents the reduced results from extensive experimentation using the proposed feature and *c* selection techniques. This table represents the best performing hierarchical models for each dataset from the parameterizations tested. Once again, we indicate positive results as bolded entries, and poor results in italics. In general, the comparative performance of the hierarchical models is improved versus those results given in Table 7.4. These results indicate the best performing choices of feature and *c* selection strategies, and as shown in the table, there is no single dominant strategy for either parameter. This could be construed as a positive as it implies that further work experimenting with more sophisticated strategies may result in further performance improvements. Examining only positive (bolded) results, the split between *c* selection techniques is even; however, feature selection is skewed towards the use of a correlation coefficient. Interestingly, all three poorly performing datasets (italicized) make use of the validation strategy for choosing *c*, and two out of three use the performance-based feature selection protocol.

Examining the relative performance of the hierarchical model versus the flat topology, we observe that, when using the defined parameter selection techniques, the hierarchical methodology matches or beats the flat rule-base in all but a couple cases, including some datasets where this was not the case when *c* was chosen as a static parameter; specifically, Treasury and Wizmir both display neutral results, while Friedman is now a positive result. We further highlight that the *c* selection strategies, in combination with the hierarchical architecture, have autonomously avoided overtraining, which is not the case with the flat rule-based models or the use of a static *c*. The absolute errors are not always lower in these cases than in Table 7.4, but the elimination of overtraining effects and an overall better balance between the training and testing errors is a significant positive which would seem worth the cost.

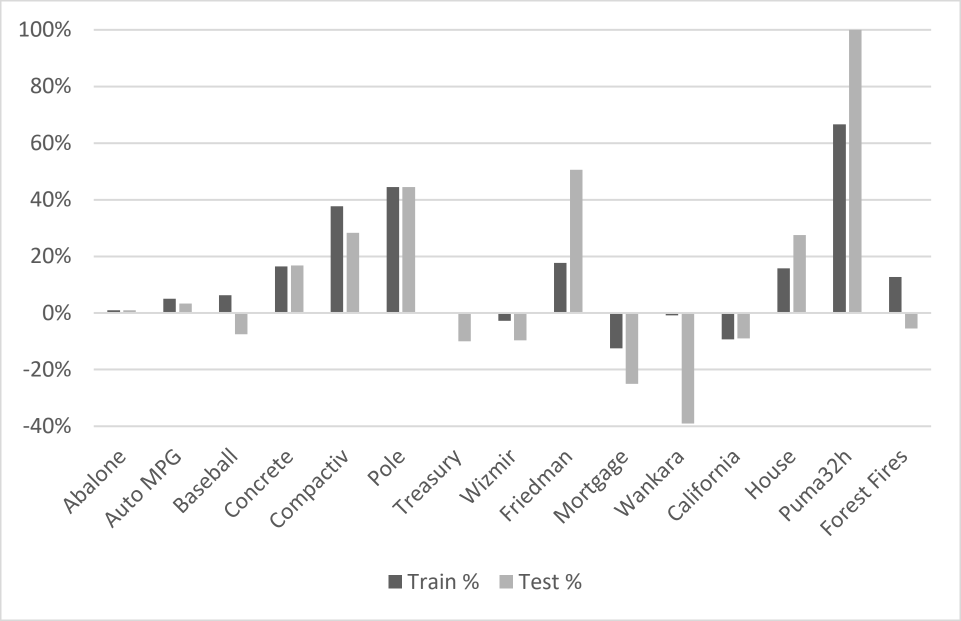


Figure 7.6: Percent reduction of RMSE using a hierarchical topology

These results demonstrate that the proposed methodology can produce high quality interpretable fuzzy rule-based models with a bare minimum of input parameterizations and prior knowledge of the dataset. In those cases where improvement is seen, the degree of error reduction achieved by the hierarchical architecture is high. The percent reduction in RMSE as seen in Table 7.5 is visualized in Figure 7.6. We highlight the Puma32h (66%), House (16%), Compactiv (38%), Pole (44%), Friedman (18%), and Concrete (16%) datasets, where the values given in brackets are the percent reduction in training error, with the degree of improvement being greater or equivalent in the testing case for all datasets other than Compactiv. Many of these datasets are of high dimensionality, and the experiments indicate that the hierarchical scheme likely aids in dealing with highly dimensional datasets, a traditional weak point for FRBS's due to rule explosion and the curse of dimensionality. We acknowledge that these experiments show some cases where the performance of the FRBS is decreased by moving towards a hierarchical topology. Specifically, the Mortgage and California datasets produced a 13% and 9% increase in training error with the hierarchical architecture, respectively. We can only speculate that these are dataset specific issues, which may be resolvable through a different rule generation procedure which is better able to model system knowledge. Over the full range of datasets tested we report an average reduction in training error of 6%, and an average reduction of testing error of 11%.

7.3 FINAL REMARKS

This chapter proposes a hierarchical FRBS architecture, alongside a well-defined procedure for the extraction of this hierarchical structure from data. The proposed architecture contains a cascading structure of fuzzy rules, in which the predictions of the previous layer are considered in the output part of the rules at the next level. This allows for the refinement of model predictions as the input is processed down the hierarchical structure and avoids interpretability issues regarding meaningless intermediate values. We assert that the proposed topology does not negatively impact the overall interpretability of the FRBS, as it results in simpler individual rules, which, if needed, can be recombined into a classical format.

We have provided comprehensive experiments which demonstrate the feasibility of the proposed methodology, established the behavior of the proposed parameter selection strategies, and compared the performance of the hierarchical model to equivalently complex flat FRBS's. We have shown experimentally that the hierarchical FRBS's are able to provide a significant performance improvement versus a flat topology for the majority of datasets tested, without the loss of model interpretability.

This study proposes a methodology for the extraction of a hierarchical FRBS from data using only a single rule extracting algorithm. Future research in this area could include exploring the performance of different rule extraction techniques to observe the quality of the resulting hierarchical model. On a similar note, our study was limited to considering only rudimentary feature selection techniques, and only one cluster validation index as parameter selection strategies. Both of these topics contain a significant amount of existing work which could be applied to this type of model, improving the performance, efficiency, or both of the methodology. There remains significant room for further work in the area of hierarchical rule-based systems, especially in relation to refining the extraction process.

8 CONCLUSIONS AND FUTURE STUDIES

Fuzzy rule-based systems provide a compact and powerful platform for the development of human readable computational models. FRBSs are useful in their own right in the standard TS or Mamdani formats; however, the flexibility of the format, alongside diverse generation options, enables many opportunities for further research, some directions of which are explored in this dissertation. Due to the flexibility and simplicity of the format, FRBSs are a strong candidate for the exploration and analysis of novel architectures, topologies, or data structures. In this thesis, we studied this topic from two perspectives. First, we examined the use of information granules as a component part of a FRBS, applying information granules to the consequent part of the fuzzy rules, with the goals of furthering the applications of information granulation in computational modeling and improving the interpretability of the model. We assessed this rule structure experimentally through coverage and specificity and demonstrated the performance of this new rule format. Second, we have proposed and studied a novel hierarchical fuzzy rule-based architecture in which the decision-making load is divided between several specialized hierarchical levels, with the intent of simplifying the rules involved in decision making as well as improving predictive performance. The ability of the hierarchical fuzzy rule-based architecture to improve predictive performance was demonstrated experimentally, and the implications of this structure on model interpretability have been analyzed in detail.

In addition to the use of new data structures and novel architectures, FRBS's can also be applied as component parts of more complex computational structures. In this dissertation, we explored the concept of using small FRBS as the component weak learners of a boosted ensemble, demonstrating that the classification rate of the boosted ensemble comfortably outperforms a single FRBS, indicating a successful integration of this structure in a boosted context. We further showed that the classification rate of the ensemble composed of FRBS's is generally on par with standard choices of weak learners for the datasets tested, indicating that the proposed learner is at least feasible in this context.

The extensive applications of FRBS's have resulted in significant research effort being applied to certain areas of meta research. This includes the evaluation of fuzzy rule interpretability and human readability, the development of indices for evaluating the difficult choice of selecting the correct number of rules, and other methods for qualitatively evaluating certain aspects of rule quality. In this dissertation, we proposed three quantitative rule quality metrics, aimed at assessing the degree of rule quality through the lens of rule stability. We defined rule stability as the ability of a methodology to consistently produce the same high-level rules, with reproducibility indicating that rules are a good fit for the modeled data. We demonstrated experimentally how the proposed metrics can be used to assess certain aspects of rule quality, and how the evaluated quantities can be used in combination to obtain an overarching evaluation of rule quality.

This dissertation examines fuzzy rule-based systems from many perspectives, including their design, applications, and analysis. We propose novel fuzzy rule architectures, explore novel rule extraction methodologies, examine the application of information granulation, apply fuzzy rules to boosted ensembles, and analyze and define certain aspects of rule stability. The work proposed in this dissertation highlights many interesting research topics regarding fuzzy rule-based systems, and we assert the following key findings from our studies:

- The proposed rule stability metrics successfully indicate the degree of rule reproducibility from data, providing a meaningful assessment of rule quality.
- Different rule stability metrics indicate different aspects of rule quality, with multiplicity and generalization indicating a degree of increased stability, and conflict providing a contrasting perspective.
- The application of information granules in the output part of fuzzy rules significantly improves the human readability of the model as compared to a standard TS-FRBS.
- Fuzzy rules are successfully generated through hierarchical clustering, although we demonstrate that in most cases hierarchical clustering is outperformed by Fuzzy C-Means clustering in the proposed format and in standard Mamdani style fuzzy rules.
- The application of small FRBS's to a boosted ensemble demonstrated increased classification performance when compared to a single equivalent learner, indicating a successful integration of FRBS's and boosting technology.
- The use of a weighted FCM and a sigmoidal modifier on the output part of the classification rules introduced significant flexibility in the FRBS and promoted the effectiveness of boosting.
- The proposed hierarchical architecture is shown experimentally to frequently improve predictive accuracy as compared to a standard flat fuzzy rule-base.
- The rules making up the hierarchical structure are simpler as they employ fewer features at a time, improving the readability of the system as well as producing more specialized rules with respect to individual features.
- The parameterization strategies used in the formation of the hierarchical structure are shown to autonomously avoid overfitting and produce high quality models, successfully simplifying the model generation configuration.

These items highlight the major points of interest as studied and analyzed in this dissertation and represent a brief indication of the knowledge gained through the presented work.

8.1 POSSIBLE LIMITATIONS OF THE RESEARCH

Most of the research presented in this dissertation demonstrated positive results with respect to the goals of each study; however, in each case we identify certain limitations or weaknesses in the presented work.

In the case of our stability metrics, we identify that, in the proposed formulations, the scaling factors do not fully compensate for the apparent favoritism towards a lower number of rules. While we intuitively expect smaller models to produce more repeatable rules, this behavior is undesirable as we would prefer that the metrics point towards those cases where the rules best fit the data. This behavior is most clearly shown through multiplicity, which tends to decrease as the number of rules increases, but this limitation is extended logically to the other two metrics as well. Generalization scores tend to increase with a larger number of rules simply because the problem space becomes larger, and so generalizations become more likely. Similarly, conflicts are also more likely as the number of consequent granules is increased with the number of rules. These limitations are identified in the full study as a significant weakness of the proposed stability metrics, and we suggest that resolving this issue would be a major step towards improving their usability.

In our application of information granules to the output parts of fuzzy rules, there are two primary limitations in the proposed work. First, the application of the information granules is limited to the output part of the rules, and only considers the very simple data structure of an interval. More complex granular data formats exist in the literature which could improve the descriptive or interpretative aspect of the model, and the possibility of extending the proposed methodology to a fully granular FRBS exists. Second, the proposed format is evaluated though coverage and specificity; specifically, the trade-off between these two criteria, as a function of interval size. It is difficult to assess the performance of a granular model due to the obvious incompatibilities between a predicted granule and a numerical target. While coverage and specificity do a good job of capturing the critical aspects of justifiable granularity, the relative importance of each measure is difficult to determine and may be dataset specific. In our study, we do not delve into this topic, instead opting to assess models through an area under the curve measurement, avoiding any focused discussion the optimal choice of the interval generation parameter.

When discussing boosted ensembles, we have limited our research to a single methodology for the generation of the weak learners (FRBS) from weighted data. The literature contains countless studies detailing different methodologies for the extraction of fuzzy rules from data, and many of these could be adapted to a boosted environment, possibly improving the overall performance of the ensemble. Secondly, our study considers only one form of fuzzy classification rules. Other classification rule formats have been proposed in the literature, and, in the interests of developing a complete knowledge-base on boosting with fuzzy rules, the use of these formats should be considered. In addition to these two primary limitations, there are certain minor considerations which have not addressed. These include the detailed examination of modelling parameters, such as the fuzzification coefficient, and the exploration of different boosting algorithms.

The generation of a hierarchical FRBS is also limited to fuzzy rule extraction using only FCM clustering, and there is certainly potential for further improvement of predictive performance through more sophisticated rule extraction methodologies. Further, we have only scratched the surface of feasible parameter selection strategies, having only considered a single cluster validation index alongside simplistic feature selection tools. The features selected, and the number of rules generated at each level, have critical implications on the behavior (and subsequently performance) of the hierarchical model, and our limited study of this topic highlights a limitation of the presented work. Finally, when choosing the number of features, our study is limited to the simplest case of adding one feature at each level. This choice was made to maximize the degree of hierarchy in our experiments; however, as we have left this facet of hierarchical modelling fully unexplored, it represents another limitation of the presented work.

8.2 FUTURE RESEARCH DIRECTIONS

There is significant opportunity for further research in each of the topics addressed in this dissertation. Many of the studies proposed in this thesis are necessarily limited to specific methodologies or parameterizations to keep the scope of the study manageable, and these limitations provide avenues for further research.

Our stability metrics represent an initial foray into the evaluating fuzzy rule quality from the perspective of rule stability, and with additional research it is likely our metrics could be refined or improved. As mentioned in the previous section, certain metrics are biased towards smaller numbers of rules, and the integer abstraction methodology we employ is a very simple one. Both aspects are candidates for improvement and further study. By improving the “parameter independent” variants of each stability metric, the quality of the stability analysis would be greatly improved, as researchers looking to use our metrics to analyze the quality of fuzzy rules would not need to be cognizant of the weaknesses we have discussed. While determining a natural number of rules for a problem is not the primary focus of our stability metrics, eliminating this weakness would significantly improve the usability of the proposed metrics. Another aspect of our methodology which could provide an opportunity for further research is the improvement or modification of the rule abstraction methodology. The goal of rule abstraction is simply to facilitate rule comparison for stability analysis, and any other methodology which meets this criterion could be viable. More sophisticated strategies may result in higher quality rule comparisons, which would hopefully, in turn, result in higher quality stability analysis.

Our study on partially granularized fuzzy rules has limited options regarding future research directions. We experimented with the use of hierarchical clustering as a vehicle for fuzzy rule generation, but ultimately found that it performed consistently worse than FCM clustering. As such, pushing on in this direction seems inadvisable. The other available research direction is further granularization of the FRBS, applying information granules to the condition parts of the fuzzy rules as well. This type of fully granular model may improve model interpretability and knowledge representation; however, granular conditions would result in additional complications, such as determining a degree of membership to granular rules, and in many senses such a study would no longer be on the topic of fuzzy rule-based systems, as the fuzziness is largely removed from the system.

When constructing a boosted ensemble using FRBS's, we have only considered a rudimentary classification rule format, and only one rule extraction procedure. Both items provide ample opportunity for additional research. As we have mentioned a few times already in this dissertation, many fuzzy rule extraction methodologies exist, and it is possible that a different rule extraction methodology would result in a greater degree of improvement through boosting. Similarly, more sophisticated fuzzy classification rule formats may also prove beneficial to overall performance and would certainly be worthy of future research effort. The study of these variations would eventually lead to a well-defined generalized fuzzy rule boosting methodology. In the interest of defining a complete fuzzy boosting methodology, we may also wish to consider different boosting algorithms or other types of fuzzy classifier. The application of fuzzy models to a boosted environment has very limited existing literature, and as a result, this topic has plenty of opportunity for further research. Our work represents an initial case study on the topic of applying fuzzy models to boosted ensembles, and, on the more general topic of boosting with fuzzy models, there remains a great deal of unexplored potential.

Our hierarchical fuzzy rule-based architecture also provides opportunity for further study. In the presented methodology and subsequent experiments, we have limited our study to only consider a single additional feature at each level of the hierarchy. Future research could employ some strategy to make an informed choice of this configuration parameter, possibly adjusting the number of features used at each level dynamically during the hierarchical construction process. While there is no guarantee, this direction of experimentation may prove beneficial to overall predictive performance, and there are opportunities for interpretability improvements as well (e.g. grouping linguistically related features). As discussed in the previous section highlighting certain limitations of our research, there is additional opportunity for improvement with respect to the parameter selection strategies employed by our methodology. First, when choosing the number of rules to compute at each level, we have considered only one cluster validation index out of many available options, and other strategies for selecting this parameter exist. Second, we examine only a couple of simple feature selection techniques, and improvement, or at least useful research, could be made regarding this aspect of model construction. Finally, as with our boosting study and the partially granular FRBS, there are many rule extraction methodologies available, some of which are compatible with the proposed hierarchical structure.

Generally, the work presented in this thesis is limited to a single rule extraction strategy, and this specificity in our work allows for variations on the proposed methodologies to be explored. Another common theme in this section is the identification of the limited parameterization options tested in our studies. This is largely the result of maintaining a reasonable scope for each topic; however, we recognize this facet as both a weakness in the existing work as well as an opportunity for further research wishing to generalize our initial studies.

References

[1] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. Plenum Press, 1981.

[2] J. Cózar, L. delaOssa, and J. A. Gámez, “Learning compact zero-order TSK fuzzy rule-based systems for high-dimensional problems using an Apriori + local search approach,” *Information Sciences*, vol. 433–434, pp. 1–16, Apr. 2018.

[3] J. Dhar and S. Arora, “Designing fuzzy rule base using Spider Monkey Optimization Algorithm in cooperative framework,” *Future Computing and Informatics Journal*, vol. 2, no. 1, pp. 31–38, Jun. 2017.

[4] I. Rodríguez-Fdez, M. Mucientes, and A. Bugarín, “FRULER: Fuzzy Rule Learning through Evolution for Regression,” *Information Sciences*, vol. 354, pp. 1–18, Aug. 2016.

[5] A. Amouzadi and A. Mirzaei, “Hierarchical fuzzy rule-based classification system by evolutionary boosting algorithm,” in *2010 5th International Symposium on Telecommunications*, 2010, pp. 909–913.

[6] M. Korytkowski, L. Rutkowski, and R. Scherer, “Fast image classification by boosting fuzzy classifiers,” *Information Sciences*, vol. 327, pp. 175–182, Jan. 2016.

[7] M. Pota, M. Esposito, and G. De Pietro, “Designing rule-based fuzzy systems for classification in medicine,” *Knowledge-Based Systems*, vol. 124, pp. 105–132, May 2017.

[8] L.-X. Wang and J. M. Mendel, “Generating fuzzy rules by learning from examples,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[9] A. Ollero, J. Aracil, and A. García-Cerezo, “Robust design of rule-based fuzzy controllers,” *Fuzzy Sets and Systems*, vol. 70, no. 2–3, pp. 249–273, Mar. 1995.

[10] T.-P. Hong and C.-Y. Lee, “Induction of fuzzy rules and membership functions from training examples,” *Fuzzy Sets and Systems*, vol. 84, no. 1, pp. 33–47, Nov. 1996.

[11] Te-Min Chang and Y. Yih, “Generating fuzzy rule-based systems from examples,” in *Soft Computing in Intelligent Systems and Information Processing. Proceedings of the 1996 Asian Fuzzy Systems Symposium*, 1996, pp. 37–42.

[12] Y. Yuan and H. Zhuang, “A genetic algorithm for generating fuzzy classification rules,” *Fuzzy Sets and Systems*, vol. 84, no. 1, pp. 1–19, Nov. 1996.

[13] X. Z. Wang, Y. D. Wang, X. F. Xu, W. D. Ling, and D. S. Yeung, “A new approach to fuzzy rule generation: fuzzy extension matrix,” *Fuzzy Sets and Systems*, vol. 123, no. 3, pp. 291–306, Nov. 2001.

[14] J. L. Castro, L. D. Flores-Hidalgo, C. J. Mantas, and J. M. Puche, “Extraction of fuzzy rules from support vector machines,” *Fuzzy Sets and Systems*, vol. 158, no. 18, pp. 2057–2077, Sep. 2007.

[15] B. Zhu, C.-Z. He, P. Liatsis, and X.-Y. Li, “A GMDH-based fuzzy modeling approach for constructing TS model,” *Fuzzy Sets and Systems*, vol. 189, no. 1, pp. 19–29, Feb. 2012.

[16] E.-H. Kim, S.-K. Oh, and W. Pedrycz, “Reinforced rule-based fuzzy models: Design and analysis,” *Knowledge-Based Systems*, vol. 119, pp. 44–58, Mar. 2017.

[17] L. A. Zadeh, “Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic,” *Fuzzy Sets and Systems*, vol. 90, no. 2, pp. 111–127, Sep. 1997.

[18] W. Pedrycz, “Shadowed sets: representing and processing fuzzy sets.,” *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, vol. 28, no. 1, pp. 103–9, Jan. 1998.

[19] W. Pedrycz, “Interpretation of clusters in the framework of shadowed sets,” *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2439–2449, Nov. 2005.

[20] W. Pedrycz and A. Bargiela, “An optimization of allocation of information granularity in the interpretation of data structures: toward granular fuzzy clustering.,” *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, vol. 42, no. 3, pp. 582–90, Jun. 2012.

[21] A. Pedrycz, K. Hirota, W. Pedrycz, and F. Dong, “Granular representation and granular computing with fuzzy sets,” *Fuzzy Sets and Systems*, vol. 203, pp. 17–32, Sep. 2012.

[22] S.-K. Oh, W.-D. Kim, W. Pedrycz, and K. Seo, “Fuzzy Radial Basis Function Neural Networks with information granulation and its parallel genetic optimization,” *Fuzzy Sets and Systems*, vol. 237, pp. 96–117, Feb. 2014.

[23] A. Balamash, W. Pedrycz, R. Al-Hmouz, and A. Morfeq, “An expansion of fuzzy information granules through successive refinements of their information content and their use to system modeling,” *Expert Systems with Applications*, vol. 42, no. 6, pp. 2985–2997, Apr. 2015.

[24] Y. Freund, R. Schapire, and N. Abe, “A short introduction to boosting,” *Journal-Japanese Society*

For Artificial Intelligence, vol. 14, no. 771–780, p. 1612, 1999.

[25] Y. Freund and R. E. Schapire, “Experiments with a New Boosting Algorithm,” in *ICML’96 Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, 1996, p. Pages 148-156.

[26] F. Hoffmann, “Combining boosting and evolutionary algorithms for learning of fuzzy classification rules,” *Fuzzy Sets and Systems*, vol. 141, no. 1, pp. 47–58, Jan. 2004.

[27] F. Hoffmann, “Boosting a genetic fuzzy classifier,” in *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference (Cat. No. 01TH8569)*, 2001, vol. 3, pp. 1564–1569.

[28] L. A. Zadeh, “Fuzzy sets,” *Information and Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.

[29] E. Sanchez, “Resolution of composite fuzzy relation equations,” *Information and Control*, vol. 30, no. 1, pp. 38–48, Jan. 1976.

[30] K. Menger, “Statistical Metrics.,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 28, no. 12, pp. 535–7, Dec. 1942.

[31] B. Schweizer and A. Sklar, “Associative functions and statistical triangle inequalities,” *Publicationes Mathematicae Debrecen*, vol. 8, pp. 169–186, Jan. 1961.

[32] C. C. Lee, “Fuzzy logic in control systems: fuzzy logic controller. I,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, no. 2, pp. 404–418, 1990.

[33] D. Dubois and H. Prade, “What are fuzzy rules and how to use them,” *Fuzzy Sets and Systems*, vol. 84, no. 2, pp. 169–185, Dec. 1996.

[34] L. A. Zadeh, “Fuzzy logic = computing with words,” *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 2, pp. 103–111, May 1996.

[35] E. H. Mamdani, “Application of fuzzy algorithms for control of simple dynamic plant,” *Proceedings of the Institution of Electrical Engineers*, vol. 121, no. 12, p. 1585, 1974.

[36] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132, Jan. 1985.

[37] E. G. Mansoori, M. J. Zolghadri, and S. D. Katebi, “SGERD: A Steady-State Genetic Algorithm for Extracting Fuzzy Classification Rules From Data,” *IEEE Transactions on Fuzzy Systems*, vol.

16, no. 4, pp. 1061–1071, Aug. 2008.

[38] J. A. Sanz, D. Bernardo, F. Herrera, H. Bustince, and H. Hagras, “A Compact Evolutionary Interval-Valued Fuzzy Rule-Based Classification System for the Modeling and Prediction of Real-World Financial Applications With Imbalanced Data,” *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 4, pp. 973–990, Aug. 2015.

[39] N.-S. Lin, “Rule extraction for fuzzy modeling,” *Fuzzy Sets and Systems*, vol. 88, no. 1, pp. 23–30, May 1997.

[40] Y. Jin, W. Von Seelen, and B. Sendhoff, “On generating FC(3) fuzzy rule systems from data using evolution strategies.,” *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, vol. 29, no. 6, pp. 829–45, Jan. 1999.

[41] O. Cordon and F. Herrera, “A two-stage evolutionary process for designing TSK fuzzy rule-based systems.,” *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, vol. 29, no. 6, pp. 703–15, Jan. 1999.

[42] L. Sánchez and J. Otero, “A fast genetic method for inducting descriptive fuzzy models,” *Fuzzy Sets and Systems*, vol. 141, no. 1, pp. 33–46, Jan. 2004.

[43] Y. Chen, B. Yang, A. Abraham, and L. Peng, “Automatic Design of Hierarchical Takagi–Sugeno Type Fuzzy Systems Using Evolutionary Algorithms,” *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 3, pp. 385–397, Jun. 2007.

[44] H. Wang, S. Kwong, Y. Jin, W. Wei, and K. F. Man, “Multi-objective hierarchical genetic algorithm for interpretable fuzzy rule-based knowledge extraction,” *Fuzzy Sets and Systems*, vol. 149, no. 1, pp. 149–186, Jan. 2005.

[45] S. Elsayed, R. Sarker, and C. A. Coello Coello, “Fuzzy Rule-Based Design of Evolutionary Algorithm for Optimization,” *IEEE Transactions on Cybernetics*, pp. 1–14, 2017.

[46] A. A. A. Esmin, “Generating Fuzzy Rules from Examples Using the Particle Swarm Optimization Algorithm,” in *7th International Conference on Hybrid Intelligent Systems (HIS 2007)*, 2007, pp. 340–343.

[47] C.-F. Juang and C. Lo, “Zero-order TSK-type fuzzy system learning using a two-phase swarm intelligence algorithm,” *Fuzzy Sets and Systems*, vol. 159, no. 21, pp. 2910–2926, Nov. 2008.

[48] Euntai Kim, Minkee Park, Seunghwan Ji, and Mignon Park, “A new approach to fuzzy modeling,” *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 3, pp. 328–337, 1997.

[49] N. L. Tsakiridis, J. B. Theocharis, and G. C. Zalidis, “A fuzzy rule-based system utilizing differential evolution with an application in vis-NIR soil spectroscopy,” in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2017, pp. 1–7.

[50] Chia-Feng Juang and Po-Han Chang, “Designing Fuzzy-Rule-Based Systems Using Continuous Ant-Colony Optimization,” *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 138–149, Feb. 2010.

[51] C.-F. Juang and C.-D. Hsieh, “TS-fuzzy system-based support vector regression,” *Fuzzy Sets and Systems*, vol. 160, no. 17, pp. 2486–2504, Sep. 2009.

[52] W.-Y. Cheng and C.-F. Juang, “An incremental support vector machine-trained TS-type fuzzy system for online classification problems,” *Fuzzy Sets and Systems*, vol. 163, no. 1, pp. 24–44, Jan. 2011.

[53] E. D’Andrea and B. Lazzerini, “A hierarchical approach to multi-class fuzzy classifiers,” *Expert Systems with Applications*, vol. 40, no. 9, pp. 3828–3840, Jul. 2013.

[54] T. D. Gedeon and D. Tikk, “Constructing hierarchical fuzzy rule bases for classification,” in *10th IEEE International Conference on Fuzzy Systems. (Cat. No.01CH37297)*, 2001, vol. 2, pp. 1388–1391.

[55] C. Zou, H. Deng, J. Wan, Z. Wang, and P. Deng, “Mining and updating association rules based on fuzzy concept lattice,” *Future Generation Computer Systems*, vol. 82, pp. 698–706, May 2018.

[56] L.-C. Dutu, G. Mauris, and P. Bolon, “A Fast and Accurate Rule-Base Generation Method for Mamdani Fuzzy Systems,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 2, pp. 715–733, Apr. 2018.

[57] M. Setnes, “Supervised fuzzy clustering for rule extraction,” *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 4, pp. 416–424, 2000.

[58] G. Tsekouras, H. Sarimveis, E. Kavakli, and G. Bafas, “A hierarchical fuzzy-clustering approach to fuzzy modeling,” *Fuzzy Sets and Systems*, vol. 150, no. 2, pp. 245–266, Mar. 2005.

[59] J. Y. Su and C. C. Kung, “Affine Takagi-Sugeno fuzzy modelling algorithm by fuzzy c-regression models clustering with a novel cluster validity criterion,” *IET Control Theory & Applications*, vol.

1, no. 5, pp. 1255–1265, Sep. 2007.

[60] M. Delgado, A. F. Gómez-Skarmeta, and F. Martín, “A methodology to model fuzzy systems using fuzzy clustering in a rapid-prototyping approach,” *Fuzzy Sets and Systems*, vol. 97, no. 3, pp. 287–301, Aug. 1998.

[61] A. F. Gómez-Skarmeta, M. Delgado, and M. A. Vila, “About the use of fuzzy clustering techniques for fuzzy model identification,” *Fuzzy Sets and Systems*, vol. 106, no. 2, pp. 179–188, Sep. 1999.

[62] X. Zhang, X. Pan, and S. Wang, “Fuzzy DBN with rule-based knowledge representation and high interpretability,” in *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, 2017, pp. 1–7.

[63] E. G. Mansoori, “FRBC: A Fuzzy Rule-Based Clustering Algorithm,” *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 5, pp. 960–971, Oct. 2011.

[64] H. Zhang and J. Lu, “Creating ensembles of classifiers via fuzzy clustering and deflection,” *Fuzzy Sets and Systems*, vol. 161, no. 13, pp. 1790–1802, Jul. 2010.

[65] A. M. Palacios, L. Sanchez, and I. Couso, “Boosting fuzzy rules with low quality data in multi-class problems: Open problems and challenges,” in *2013 IEEE International Workshop on Genetic and Evolutionary Fuzzy Systems (GEFS)*, 2013, pp. 28–35.

[66] L. Sánchez and J. Otero, “Boosting fuzzy rules in classification problems under single-winner inference,” *International Journal of Intelligent Systems*, vol. 22, no. 9, pp. 1021–1034, Sep. 2007.

[67] D. S. Yeung, X. Z. Wang, and E. C. C. Tsang, “Learning weighted fuzzy rules from examples with mixed attributes by fuzzy decision trees,” in *IEEE SMC’99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*, 1999, vol. 3, pp. 349–354.

[68] P. Krömer and J. Platoš, “Simultaneous Prediction of Wind Speed and Direction by Evolutionary Fuzzy Rule Forest,” *Procedia Computer Science*, vol. 108, pp. 295–304, Jan. 2017.

[69] N. T. Thong and L. H. Son, “HIFCF: An effective hybrid model between picture fuzzy clustering and intuitionistic fuzzy recommender systems for medical diagnosis,” *Expert Systems with Applications*, vol. 42, no. 7, pp. 3682–3701, May 2015.

[70] V. Lacagnina, M. S. Leto-Barone, S. La Piana, G. La Porta, G. Pingitore, and G. Di Lorenzo,

"Comparison between statistical and fuzzy approaches for improving diagnostic decision making in patients with chronic nasal symptoms," *Fuzzy Sets and Systems*, vol. 237, pp. 136–150, Feb. 2014.

[71] R. Cheruku, D. R. Edla, V. Kuppili, and R. Dharavath, "RST-BatMiner: A fuzzy rule miner integrating rough set feature selection and Bat optimization for detection of diabetes disease," *Applied Soft Computing*, vol. 67, pp. 764–780, Jun. 2017.

[72] F. Mansourypoor and S. Asadi, "Development of a Reinforcement Learning-based Evolutionary Fuzzy Rule-Based System for diabetes diagnosis," *Computers in Biology and Medicine*, vol. 91, pp. 337–352, Dec. 2017.

[73] J. A. Mendez *et al.*, "Improving the anesthetic process by a fuzzy rule based medical decision system," *Artificial Intelligence in Medicine*, vol. 84, pp. 159–170, Jan. 2018.

[74] A. Jindal, A. Dua, N. Kumar, A. K. Das, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Providing Healthcare-as-a-Service Using Fuzzy Rule-Based Big Data Analytics in Cloud Computing," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–1, 2018.

[75] P. Shukla, Abhishek, and S. Verma, "A compact fuzzy rule interpretation of SVM classifier for medical whole slide images," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, 2017, pp. 1588–1592.

[76] S. Das and S. Meher, "A novel shadow detection method using fuzzy rule based model," in *2017 IEEE 15th Student Conference on Research and Development (SCOReD)*, 2017, pp. 493–498.

[77] P. Angelov and X. Gu, "A cascade of deep learning fuzzy rule-based image classifier and SVM," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 746–751.

[78] S.-M. Chen and T.-K. Li, "Evaluating students' learning achievement based on fuzzy rules with fuzzy reasoning capability," *Expert Systems with Applications*, vol. 38, no. 4, pp. 4368–4381, Apr. 2011.

[79] G. T. S. Ho, W. H. Ip, C. H. Wu, and Y. K. Tse, "Using a fuzzy association rule mining approach to identify the financial data association," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9054–9063, Aug. 2012.

[80] P. Hajek, "Predicting corporate investment/non-investment grade by using interval-valued fuzzy rule-based systems—A cross-region analysis," *Applied Soft Computing*, vol. 62, pp. 73–85, 2018.

[81] X. Liu, H. An, L. Wang, and Q. Guan, "Quantified moving average strategy of crude oil futures market based on fuzzy logic rules and genetic algorithms," *Physica A: Statistical Mechanics and its Applications*, vol. 482, pp. 444–457, Sep. 2017.

[82] A. Dziech and M. B. Gorzałczany, "Decision making in signal transmission problems with interval-valued fuzzy sets," *Fuzzy Sets and Systems*, vol. 23, no. 2, pp. 191–203, Aug. 1987.

[83] X. Zhang, E. Onieva, A. Perallos, E. Osaba, and V. C. S. Lee, "Hierarchical fuzzy rule-based system optimized with genetic algorithms for short term traffic congestion prediction," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 127–142, Jun. 2014.

[84] A. Sauerländer-Biebl, E. Brockfeld, D. Suske, and E. Melde, "Evaluation of a transport mode detection using fuzzy rules," *Transportation Research Procedia*, vol. 25, pp. 591–602, 2017.

[85] J. Chen, M. Weiszer, E. Zareian, M. Mahfouf, and O. Obajemu, "Multi-objective fuzzy rule-based prediction and uncertainty quantification of aircraft taxi time," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–5.

[86] M. Chowdhury, J. Gao, and R. Islam, "Fuzzy rule based approach for face and facial feature extraction in biometric authentication," in *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, 2016, pp. 1–5.

[87] N. Naik, R. Diao, and Q. Shen, "Dynamic Fuzzy Rule Interpolation and its Application to Intrusion Detection," *IEEE Transactions on Fuzzy Systems*, pp. 1–1, 2017.

[88] S. van der Heijden and U. Haberlandt, "A fuzzy rule based metamodel for monthly catchment nitrate fate simulations," *Journal of Hydrology*, vol. 531, pp. 863–876, Dec. 2015.

[89] S. Gautam, D. Kumar, and L. M. Patnaik, "Fuzzy rule base solution of Bi-criterion software release time problem," in *2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS)*, 2017, pp. 297–302.

[90] R. Dridi, S. Zammali, and K. Arour, "Fuzzy Rule-Based Situational Music Retrieval and Recommendation," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 2017, pp. 549–556.

[91] S.-T. Ung, "Development of a weighted probabilistic risk assessment method for offshore engineering systems using fuzzy rule-based Bayesian reasoning approach," *Ocean Engineering*, vol. 147, pp. 268–276, 2018.

[92] R. Logambigai, S. Ganapathy, and A. Kannan, “Energy–efficient grid–based routing algorithm using intelligent fuzzy rules for wireless sensor networks,” *Computers & Electrical Engineering*, vol. 68, pp. 62–75, 2018.

[93] N. M. Amaitik and C. D. Buckingham, “Developing a hierarchical fuzzy rule-based model with weighted linguistic rules: A case study of water pipes condition prediction,” in *2017 Computing Conference*, 2017, pp. 30–40.

[94] D. Zhang, M. Ji, J. Yang, Y. Zhang, and F. Xie, “A novel cluster validity index for fuzzy clustering based on bipartite modularity,” *Fuzzy Sets and Systems*, vol. 253, no. c, pp. 122–137, Oct. 2014.

[95] J. C. Bezdek, “Cluster validity with fuzzy sets,” *Journal of Cybernetics*, vol. 3, no. 3, pp. 58–73, Jan. 1973.

[96] W. Wang and Y. Zhang, “On fuzzy cluster validity indices,” *Fuzzy Sets and Systems*, vol. 158, no. 19, pp. 2095–2117, Oct. 2007.

[97] X. L. Xie and G. Beni, “A validity measure for fuzzy clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 841–847, 1991.

[98] B. Rezaee, “A cluster validity index for fuzzy clustering,” *Fuzzy Sets and Systems*, vol. 161, no. 23, pp. 3014–3025, Dec. 2010.

[99] R. J. G. B. Campello and E. R. Hruschka, “A fuzzy extension of the silhouette width criterion for cluster analysis,” *Fuzzy Sets and Systems*, vol. 157, no. 21, pp. 2858–2875, Nov. 2006.

[100] S. Bandyopadhyay, S. Saha, and W. Pedrycz, “Use of a fuzzy granulation–degranulation criterion for assessing cluster validity,” *Fuzzy Sets and Systems*, vol. 170, no. 1, pp. 22–42, May 2011.

[101] J. Kerr-Wilson and W. Pedrycz, “Some new qualitative insights into quality of fuzzy rule-based models,” *Fuzzy Sets and Systems*, vol. 307, pp. 29–49, Jan. 2017.

[102] S.-M. Zhou and J. Q. Gan, “Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy system modelling,” *Fuzzy Sets and Systems*, vol. 159, no. 23, pp. 3091–3131, Dec. 2008.

[103] Y. Jin, “Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement,” *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 2, pp. 212–221, Apr. 2000.

[104] M. I. Rey, M. Galende, M. J. Fuente, and G. I. Sainz-Palmero, “Multi-objective based Fuzzy Rule

Based Systems (FRBSs) for trade-off improvement in accuracy and interpretability: A rule relevance point of view.," *Knowledge-Based Systems*, vol. 127, pp. 67–84, Jul. 2017.

[105] R. Mikut, J. Jäkel, and L. Gröll, "Interpretability issues in data-based learning of fuzzy systems," *Fuzzy Sets and Systems*, vol. 150, no. 2, pp. 179–197, Mar. 2005.

[106] A. Gonzalez and R. Perez, "Completeness and consistency conditions for learning fuzzy rules," *Fuzzy Sets and Systems*, vol. 96, no. 1, pp. 37–51, May 1998.

[107] D. P. Pancho, J. M. Alonso, O. Cordon, A. Quirin, and L. Magdalena, "FINGRAMS: Visual Representations of Fuzzy Rule-Based Inference for Expert Analysis of Comprehensibility," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 6, pp. 1133–1149, Dec. 2013.

[108] W. Pedrycz, "Allocation of information granularity in optimization and decision-making models: Towards building the foundations of Granular Computing," *European Journal of Operational Research*, vol. 232, no. 1, pp. 137–145, Jan. 2014.

[109] W. Pedrycz, "From fuzzy data analysis and fuzzy regression to granular fuzzy data analysis," *Fuzzy Sets and Systems*, vol. 274, pp. 12–17, Sep. 2015.

[110] A. Fernández, M. Calderón, E. Barrenechea, H. Bustince, and F. Herrera, "Solving multi-class problems with linguistic fuzzy rule based classification systems based on pairwise learning and preference relations," *Fuzzy Sets and Systems*, vol. 161, no. 23, pp. 3064–3080, Dec. 2010.

[111] J. Kerr-Wilson and W. Pedrycz, "Design of rule-based models through information granulation," *Expert Systems with Applications*, vol. 46, pp. 274–285, Mar. 2016.

[112] X. Hu, W. Pedrycz, and X. Wang, "From fuzzy rule-based models to their granular generalizations," *Knowledge-Based Systems*, vol. 124, pp. 133–143, May 2017.

[113] O. F. Reyes-Galaviz and W. Pedrycz, "Granular fuzzy models: Analysis, design, and evaluation," *International Journal of Approximate Reasoning*, vol. 64, pp. 1–19, Sep. 2015.

[114] W. Lu, L. Zhang, W. Pedrycz, J. Yang, and X. Liu, "The granular extension of Sugeno-type fuzzy models based on optimal allocation of information granularity and its application to forecasting of time series," *Applied Soft Computing*, vol. 42, pp. 38–52, May 2016.

[115] M. A. Sanchez, O. Castillo, J. R. Castro, and P. Melin, "Fuzzy granular gravitational clustering algorithm for multivariate data," *Information Sciences*, vol. 279, pp. 498–511, Sep. 2014.

[116] X. Wang, X. Liu, and L. Zhang, "A rapid fuzzy rule clustering method based on granular

computing," *Applied Soft Computing*, vol. 24, pp. 534–542, Nov. 2014.

[117] W. Pedrycz, R. Al-Hmouz, A. S. Balamash, and A. Morfeq, "Designing granular fuzzy models: A hierarchical approach to fuzzy modeling," *Knowledge-Based Systems*, vol. 76, pp. 42–52, Mar. 2015.

[118] W. Froelich and W. Pedrycz, "Fuzzy cognitive maps in the modeling of granular time series," *Knowledge-Based Systems*, vol. 115, pp. 110–122, Jan. 2017.

[119] W. Pedrycz, B. Russo, and G. Succi, "Knowledge transfer in system modeling and its realization through an optimal allocation of information granularity," *Applied Soft Computing*, vol. 12, no. 8, pp. 1985–1995, Aug. 2012.

[120] Y. Yao, S. Wang, and X. Deng, "Constructing shadowed sets and three-way approximations of fuzzy sets," *Information Sciences*, vol. 412–413, pp. 132–153, Oct. 2017.

[121] J. Zhou, W. Pedrycz, and D. Miao, "Shadowed sets in the characterization of rough-fuzzy clustering," *Pattern Recognition*, vol. 44, no. 8, pp. 1738–1749, Aug. 2011.

[122] J. Zhou, Z. Lai, D. Miao, C. Gao, and X. Yue, "Multigranulation rough-fuzzy clustering based on shadowed sets," *Information Sciences*, May 2018.

[123] J. Zhou, Z. Lai, C. Gao, D. Miao, and X. Yue, "Rough Possibilistic C-Means Clustering Based on Multigranulation Approximation Regions and Shadowed Sets," *Knowledge-Based Systems*, Jul. 2018.

[124] Y. Zhou, H. Su, and H. Zhang, "A novel data selection method based on shadowed sets," *Procedia Engineering*, vol. 15, pp. 1410–1415, Jan. 2011.

[125] P. Grzegorzewski, "Fuzzy number approximation via shadowed sets," *Information Sciences*, vol. 225, pp. 35–46, Mar. 2013.

[126] Z. Pawlak, "Rough Sets," *Internat. J. Comput. Inform*, vol. 11, no. 5, pp. 341–356, 1982.

[127] S. Nanda and S. Majumdar, "Fuzzy rough sets," *Fuzzy Sets and Systems*, vol. 45, no. 2, pp. 157–160, Jan. 1992.

[128] Z. Pawlak and A. Skowron, "Rudiments of rough sets," *Information Sciences*, vol. 177, no. 1, pp. 3–27, Jan. 2007.

[129] Y. Qian, Q. Wang, H. Cheng, J. Liang, and C. Dang, "Fuzzy-rough feature selection accelerator," *Fuzzy Sets and Systems*, May 2014.

[130] A. Hassanien, “Fuzzy rough sets hybrid scheme for breast cancer detection,” *Image and Vision Computing*, vol. 25, no. 2, pp. 172–183, Feb. 2007.

[131] A. Petrosino and A. Ferone, “Rough fuzzy set-based image compression,” *Fuzzy Sets and Systems*, vol. 160, no. 10, pp. 1485–1506, May 2009.

[132] C. Chen, N. Mac Parthaláin, Y. Li, C. Price, C. Quek, and Q. Shen, “Rough-fuzzy rule interpolation,” *Information Sciences*, vol. 351, pp. 1–17, Jul. 2016.

[133] J. H. Friedman, “Greedy Function Approximation: A Gradient Boosting Machine,” *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[134] Y. Zhang and A. Haghani, “A gradient boosting method to improve travel time prediction,” *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, 2015.

[135] J. H. Friedman, “Stochastic gradient boosting,” *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[136] D. Ren, F. Qu, K. Lv, Z. Zhang, H. Xu, and X. Wang, “A gradient descent boosting spectrum modeling method based on back interval partial least squares,” *Neurocomputing*, vol. 171, pp. 1038–1046, 2016.

[137] J. Zhu, S. Rosset, H. Zou, and T. Hastie, “Multi-class AdaBoost,” 2006.

[138] Y. Sun, S. Todorovic, and J. Li, “Unifying multi-class AdaBoost algorithms with binary base learners under the margin framework,” *Pattern Recognition Letters*, vol. 28, no. 5, pp. 631–643, Apr. 2007.

[139] H. Altun and G. Polat, “Boosting selection of speech related features to improve performance of multi-class SVMs in emotion detection,” *Expert Systems with Applications*, vol. 36, no. 4, pp. 8197–8203, May 2009.

[140] N. Simidjievski, L. Todorovski, and S. Džeroski, “Predicting long-term population dynamics with bagging and boosting of process-based models,” *Expert Systems with Applications*, vol. 42, no. 22, pp. 8484–8496, Dec. 2015.

[141] D. R. Nayak, R. Dash, and B. Majhi, “Brain MR image classification using two-dimensional discrete wavelet transform and AdaBoost with random forests,” *Neurocomputing*, vol. 177, pp. 188–197, Feb. 2016.

[142] M. Javadian, S. Bagheri Shouraki, and S. Sheikhpour Kourabbaslou, “A novel density-based fuzzy

clustering algorithm for low dimensional feature space," *Fuzzy Sets and Systems*, vol. 318, pp. 34–55, 2017.

[143] T. Chen and S. Lu, "Accurate and Efficient Traffic Sign Detection Using Discriminative AdaBoost and Support Vector Regression," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 4006–4015, Jun. 2016.

[144] L. Li, C. Wang, W. Li, and J. Chen, "Hyperspectral image classification by AdaBoost weighted composite kernel extreme learning machines," *Neurocomputing*, vol. 275, pp. 1725–1733, Jan. 2018.

[145] Y. Zhao, L. Gong, B. Zhou, Y. Huang, and C. Liu, "Detecting tomatoes in greenhouse scenes by combining AdaBoost classifier and colour analysis," *Biosystems Engineering*, vol. 148, pp. 127–137, Aug. 2016.

[146] M. Wang, L. Guo, and W.-Y. Chen, "Blink detection using Adaboost and contour circle for fatigue recognition," *Computers & Electrical Engineering*, vol. 58, pp. 502–512, Feb. 2017.

[147] S. Yin, P. Ouyang, X. Dai, L. Liu, and S. Wei, "An AdaBoost-Based Face Detection System Using Parallel Configurable Architecture With Optimized Computation," *IEEE Systems Journal*, vol. 11, no. 1, pp. 260–271, Mar. 2017.

[148] J. Rajeshwari, K. Karibasappa, and M. T. Gopalkrishna, "Adaboost modular tensor locality preservative projection: face detection in video using Adaboost modular-based tensor locality preservative projections," *IET Computer Vision*, vol. 10, no. 7, pp. 670–678, Oct. 2016.

[149] E. Y. Kim, "Vision-based wheelchair navigation using geometric AdaBoost learning," *Electronics Letters*, vol. 53, no. 8, pp. 534–536, Apr. 2017.

[150] L. Zhang, J. Yin, J. Lin, X. Wang, and J. Guo, "Detection of coronal mass ejections using AdaBoost on grayscale statistic features," *New Astronomy*, vol. 48, pp. 49–57, Oct. 2016.

[151] C. Gao, P. Li, Y. Zhang, J. Liu, and L. Wang, "People counting based on head detection combining Adaboost and CNN in crowded surveillance environment," *Neurocomputing*, vol. 208, pp. 108–116, Oct. 2016.

[152] M. Yang, J. Crenshaw, B. Augustine, R. Mareachen, and Y. Wu, "AdaBoost-based face detection for embedded systems," *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1116–1125, Nov. 2010.

[153] J. Zheng, “Cost-sensitive boosting neural networks for software defect prediction,” *Expert Systems with Applications*, vol. 37, no. 6, pp. 4537–4543, Jun. 2010.

[154] I. Martin-Diaz, D. Morinigo-Sotelo, O. Duque-Perez, and R. de J. Romero-Troncoso, “Early Fault Detection in Induction Motors Using AdaBoost With Imbalanced Small Data and Optimized Sampling,” *IEEE Transactions on Industry Applications*, vol. 53, no. 3, pp. 3066–3075, May 2017.

[155] M.-J. Kim, D.-K. Kang, and H. B. Kim, “Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction,” *Expert Systems with Applications*, vol. 42, no. 3, pp. 1074–1082, Feb. 2015.

[156] J. Heo and J. Y. Yang, “AdaBoost based bankruptcy forecasting of Korean construction companies,” *Applied Soft Computing*, vol. 24, pp. 494–499, Nov. 2014.

[157] J. Sun, M. Jia, and H. Li, “AdaBoost ensemble for financial distress prediction: An empirical comparison with data from Chinese listed companies,” *Expert Systems with Applications*, vol. 38, no. 8, pp. 9305–9312, Aug. 2011.

[158] E. Alfaro, N. García, M. Gámez, and D. Elizondo, “Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks,” *Decision Support Systems*, vol. 45, no. 1, pp. 110–122, Apr. 2008.

[159] L. Guelman, “Gradient boosting trees for auto insurance loss cost modeling and prediction,” *Expert Systems with Applications*, vol. 39, no. 3, pp. 3659–3667, Feb. 2012.

[160] J. Huang and M. Perry, “A semi-empirical approach using gradient boosting and k-nearest neighbors regression for GEFCom2014 probabilistic solar power forecasting,” *International Journal of Forecasting*, vol. 32, no. 3, pp. 1081–1086, 2016.

[161] L. Wang, S. Lv, and Y.-R. Zeng, “Effective sparse adaboost method with ESN and FOA for industrial electricity consumption forecasting in China,” *Energy*, Apr. 2018.

[162] M. Yousefi, M. Yousefi, R. P. M. Ferreira, J. H. Kim, and F. S. Fogliatto, “Chaotic genetic algorithm and Adaboost ensemble metamodeling approach for optimum resource planning in emergency departments,” *Artificial Intelligence in Medicine*, vol. 84, pp. 23–33, Jan. 2018.

[163] T. Danisman, I. M. Bilasco, and J. Martinet, “Boosting gender recognition performance with a fuzzy inference system,” *Expert Systems with Applications*, vol. 42, no. 5, pp. 2772–2784, Apr. 2015.

[164] K. M. Asim, A. Idris, T. Iqbal, and F. Martínez-Álvarez, “Seismic indicators based earthquake predictor system using Genetic Programming and AdaBoost classification,” *Soil Dynamics and Earthquake Engineering*, vol. 111, pp. 1–7, Aug. 2018.

[165] H. Hong *et al.*, “Landslide susceptibility mapping using J48 Decision Tree with AdaBoost, Bagging and Rotation Forest ensembles in the Guangchang area (China),” *CATENA*, vol. 163, pp. 399–413, Apr. 2018.

[166] M. Liu, “Fingerprint classification based on Adaboost learning from singularity features,” *Pattern Recognition*, vol. 43, no. 3, pp. 1062–1070, Mar. 2010.

[167] S. Wu, F. Yuan, Y. Yang, Z. Fang, and Y. Fang, “Real-time image smoke detection using staircase searching-based dual threshold AdaBoost and dynamic analysis,” *IET Image Processing*, vol. 9, no. 10, pp. 849–856, Oct. 2015.

[168] Y. Zhao, Q. Li, and Z. Gu, “Early smoke detection of forest fire video using CS Adaboost algorithm,” *Optik - International Journal for Light and Electron Optics*, vol. 126, no. 19, pp. 2121–2124, Oct. 2015.

[169] C. Tan, H. Chen, and C. Xia, “Early prediction of lung cancer based on the combination of trace element analysis in urine and an Adaboost algorithm,” *Journal of Pharmaceutical and Biomedical Analysis*, vol. 49, no. 3, pp. 746–752, Apr. 2009.

[170] M. J. del Jesus, F. Hoffmann, L. JuncoNavascues, and L. Sanchez, “Induction of Fuzzy-Rule-Based Classifiers With Evolutionary Boosting Algorithms,” *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 3, pp. 296–308, Jun. 2004.

[171] L. I. Kuncheva, “‘Fuzzy’ versus ‘nonfuzzy’ in combining classifiers designed by boosting,” *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 729–741, Dec. 2003.

[172] W. Pedrycz and K.-C. Kwak, “Boosting of granular models,” *Fuzzy Sets and Systems*, vol. 157, no. 22, pp. 2934–2953, Nov. 2006.

[173] D. G. Stavrakoudis, I. Z. Gitas, and J. B. Theocharis, “A hierarchical genetic fuzzy rule-based classifier for high-dimensional classification problems,” in *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*, 2011, pp. 1279–1285.

[174] M. A. Kbir, H. Benkirane, K. Maalmi, and R. Benslimane, “Hierarchical fuzzy partition for pattern classification with fuzzy if-then rules,” *Pattern Recognition Letters*, vol. 21, no. 6–7, pp. 503–509, Jun. 2000.

[175] Chi-Hsing Tsai *et al.*, "An effective and efficient hierarchical fuzzy rule based classifier," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, pp. 2173–2178.

[176] Shuqing Zeng, Yongbao He, and Jie Jiang, "A Hierarchical Fuzzy System with Automatical Rule Extraction," in *The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ '05.*, pp. 477–482.

[177] A. Fernández, M. J. del Jesus, and F. Herrera, "Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets," *International Journal of Approximate Reasoning*, vol. 50, no. 3, pp. 561–577, Mar. 2009.

[178] Chia-Feng Juang, Che-Meng Hsiao, and Chia-Hung Hsu, "Hierarchical Cluster-Based Multispecies Particle-Swarm Optimization for Fuzzy-System Optimization," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 14–26, Feb. 2010.

[179] P. Salgado, "Rule generation for hierarchical collaborative fuzzy system," *Applied Mathematical Modelling*, vol. 32, no. 7, pp. 1159–1178, Jul. 2008.

[180] S. Jin and J. Peng, "Towards hierarchical fuzzy rule interpolation," in *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, 2015, pp. 267–274.

[181] M. G. Joo and J. S. Lee, "Universal approximation by hierarchical fuzzy system with constraints on the fuzzy rule," *Fuzzy Sets and Systems*, vol. 130, no. 2, pp. 175–188, Sep. 2002.

[182] M. G. Joo and J. S. Lee, "A class of hierarchical fuzzy systems with constraints on the fuzzy rules," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 2, pp. 194–203, Apr. 2005.

[183] M.-L. Lee, H.-Y. Chung, and F.-M. Yu, "Modeling of hierarchical fuzzy systems," *Fuzzy Sets and Systems*, vol. 138, no. 2, pp. 343–361, Sep. 2003.

[184] R. L. Cannon, J. V. Dave, and J. C. Bezdek, "Efficient Implementation of the Fuzzy c-Means Clustering Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 2, pp. 248–255, Mar. 1986.

[185] B. . Chaudhuri, "An efficient hierarchical clustering technique," *Pattern Recognition Letters*, vol. 3, no. 3, pp. 179–183, May 1985.

[186] E. Diday and J. . Moreau, "Learning hierarchical clustering from examples — application to the

adaptive construction of dissimilarity indices," *Pattern Recognition Letters*, vol. 2, no. 6, pp. 365–378, Dec. 1984.

[187] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, vol. 4, pp. 1942–1948.

[188] X.-S. Yang, *Nature-Inspired Optimization Algorithms*. Elsevier, 2014.

[189] A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Computers & Operations Research*, vol. 33, no. 3, pp. 859–871, Mar. 2006.

[190] A. H. Gandomi, G. J. Yun, X.-S. Yang, and S. Talatahari, "Chaos-enhanced accelerated particle swarm optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 2, pp. 327–340, Feb. 2013.

[191] X.-S. Yang and N.-I. O. Algorithms, "Chapter 7-Particle Swarm Optimization," *Nature-Inspired Optimization Algorithms*, vol. 7, pp. 99–110, 2014.

[192] W. Pedrycz and F. Gomide, *Fuzzy Systems Engineering: Toward Human-Centric Computing*. John Wiley & Sons, 2007.

[193] Y. Fukuyama and M. Sugeno, "A new method of choosing the number of clusters for fuzzy c-means method," in *In 5th Fuzzy Systems Symposium*, 1989.

[194] I. Gath and A. B. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 773–780, Jul. 1989.

[195] S. Joopudi, S. S. Rathi, S. Narasimhan, and R. Rengaswamy, "A New Cluster Validity Index for Fuzzy Clustering," *IFAC Proceedings Volumes*, vol. 46, no. 32, pp. 325–330, Dec. 2013.

[196] D. Dheeru and E. Karra Taniskidou, "UCI Machine Learning Repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml.

[197] F. H. J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, "KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2–3, pp. 255–287, 2011.