UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

DEGREE FINAL PROJECT

BACHELOR DEGREE IN COMPUTER SCIENCE AND ENGINEERING

Fuzzy Modelling to deal with uncertainty and explainability

25 June 2020

Author: *Srivathsan Govindarajan*

School of Computing,
SASTRA University.

Director: *Àngela Nebot*
Computer Science Department

Abstract

The use of fuzzy logic and specifically fuzzy systems is widely accepted in all types of applications where data is imprecise or incomplete. Among its main features is the ability to model nonlinear functions of arbitrary complexity, great flexibility and simplicity, as well as its possible customization in terms of natural language. One such fuzzy system is the Fuzzy Inductive Reasoning (FIR), which is a hybrid methodology that combines fuzzy approaches with well-known pattern recognition techniques. The FIR methodology uses the data given to identify the most relevant variables (feature selection) that have the strongest causal relationship with the output variable. Then, it uses this information to derive the set of pattern rules that contain the knowledge of the behavior of the system under study. There exist in the literature publications that prove that FIR is a powerful modelling methodology for the prediction not only of dynamic systems (e.g. electricity load forecasting etc.) but also of static systems (e.g. design of energy-efficient buildings, etc.). However, a problem that arises in the FIR methodology when the available data set is large is the size of the pattern rule set generated, which substantially decreases its interpretability (explainability), making it difficult to be use for decision-making. The objective of this project is twofold. On the one hand, to explore and understand the world of fuzzy logic and fuzzy systems and study this branch of Artificial Intelligence that I did not know before. On the other hand, to develop a set of computer programs that allow us to address this problem and facilitate the development of different types of fuzzy models that address the tradeoff between precision and model understandability (simplicity). The software developed offers five modelling options, all based on fuzzy logic: classical FIR, Mamdani and Sugeno based on FIR and FIR-Mamdani and FIR-Sugeno mixed schemes. The FIR-based Mamdani and Sugeno models use specific algorithms to process and adjust membership functions to develop the fuzzy model. The generated set of rules will be concise and interpretable, but is not always possible to handle the uncertainty that the system presents. This is accomplished in the FIR-Mamdani and FIR-Sugeno mixed schemes, where a small relevant subset of the initial pattern rules are maintained to capture the uncertainty that classical fuzzy systems cannot express. These five models are tested using four datasets, some of them extracted from the UCI repository and other from real data. The results obtained are analyzed and compared between these fuzzy models. Models are also compared based

on their input parameters, such as the fitting method and the size of the pattern rule subset.

Contents

List of Figures

List of Tables

1 Introduction and Contextualization

1.1 Context

This project is a Final Project Degree (TFG) at Barcelona School of Informatics and is directed by Àngela Nebot Castells. The project has been created by the director and the author must develop an appropriate solution, achieving all requirements. This project is research-oriented and aims at improving fuzzy system models by implementing an approximation centered on decision making and explainability.

1.2 Fuzzy Logic and Fuzzy Systems

This section is intended to give a general brushstroke of the concepts of fuzzy logic and fuzzy systems to situate the reader who is unfamiliar with this branch of artificial intelligence.

Fuzzy Logic (FL) is a concept of reasoning that resembles human thinking. This is acheived by imitating the way of decision making in humans that involves all intermediate values between logical YES (1) and NO (0). The conventional computer takes precise input and produces an output as TRUE or FALSE. The inventor of fuzzy logic concept, Lotfi Zadeh [1], observed that unlike computers, the decision making among humans includes a range of possibilities between TRUE and FALSE and used this idea to develop this concept. The definite variables which are converted into the range of possibilites are called fuzzified sets.

The architecture of a fuzzy system (Figure 1) consists of the following:

- **Fuzzification Module:** It transforms the system inputs, which are crisp numbers, into fuzzy sets.
- **Knowledge Base / Rules Base:** It stores IF-THEN rules provided by experts or sometimes learned by the system itself (when data is available).
- **Inference Engine:** Intelligence in Figure 1. It simulates the human reasoning process by making fuzzy inference on the inputs and IF-THEN rules in the knowledge base.

- **Defuzzification Module:** It transforms the fuzzy set obtained by the inference engine into a crisp value.

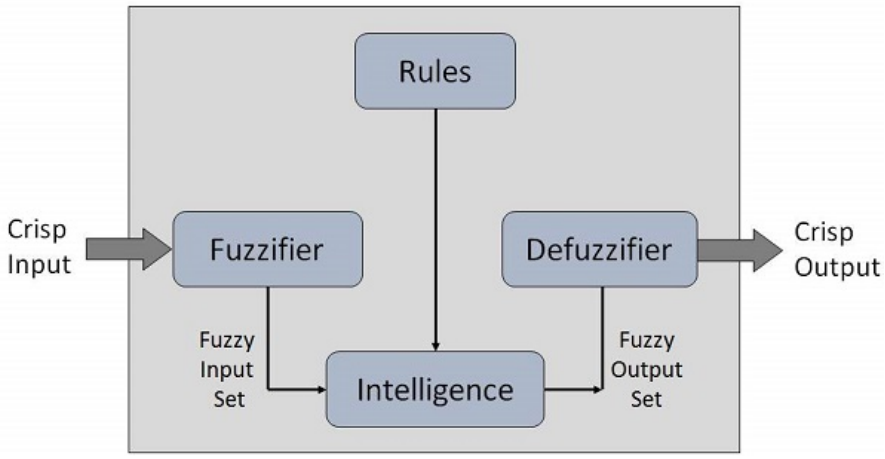

Figure 1: Fuzzy Systems general architecture.

The fuzzification and defuzzification process of a fuzzy system depends on certain functions called membership functions. These functions allow to quantify linguistic terms and represent a fuzzy set graphically. A membership function for a fuzzy set A on the universe of discourse X is defined as:

$$\mu_A : X \rightarrow [0,1] \tag{1}$$

Here, each element of X is mapped to a value between 0 and 1. It is called membership value or degree of membership. It quantifies the degree of membership of the element in X to the fuzzy set A. The Gaussian and triangular membership functions shapes are most common among various other membership function shapes such as trapezoidal, singleton, etc. An example of a triangular and Gaussian membership functions is presented in Figure 2 as given by [2].

A deep description of fuzzy logic, fuzzy sets and fuzzy systems accompanied with examples, can be found in [3]. Section 6 presents the main fuzzy methodologies on which this work focuses, describing them in more detail and highlighting the main differences between them.

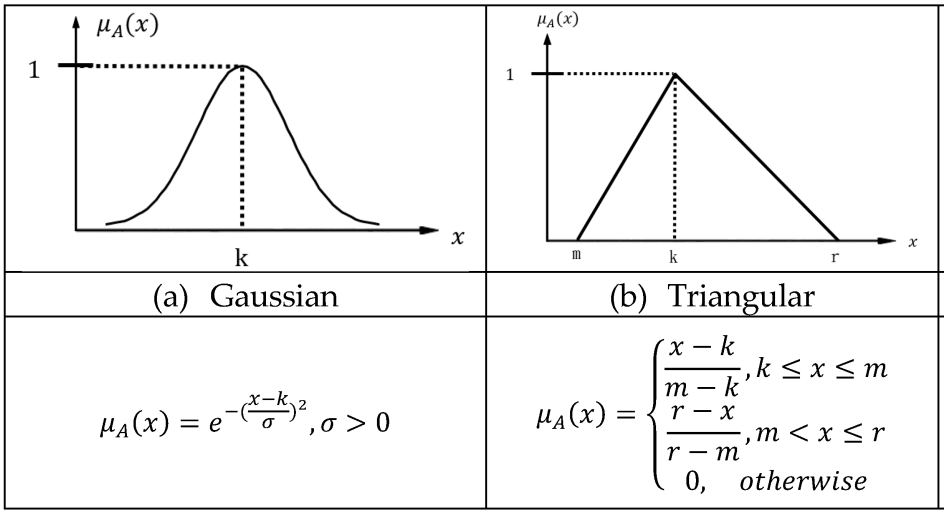

Figure 2: Example of Gaussian and triangular membership functions.

1.3 Project Description

This project focuses on the understanding and deepening of the concepts of fuzzy logic and fuzzy systems, and on the implementation of different algorithms for the modelling of systems based on fuzzy rules. All these algorithms will be based on the hybridization of FIR methodology and Mamdani's and Sugeno's classical fuzzy systems. Each of these models has a different trade-off between forecasting accuracy and model interpretability, with each being useful depending on the objective of the modeler/user.

This project includes a framework with five fuzzy systems modelling algorithms: I - FIR, II - FIR-based Mamdani, III - FIR-based Sugeno, IV - FIR-Mamdani mixed scheme, and V - FIR-Sugeno mixed scheme.

As can be deduced from the name of the algorithms, all the proposed and implemented approaches start from the FIR methodology and have different levels of hybridization. Figure 3 schematically presents the hybridizations defined in each algorithm.

The FIR approach can capture very efficiently the behavior of the system in its pattern rule base, and to perform a feature selection process to identify the more relevant input variables. Therefore, the inference process using the FIR model is usually high performing and obtains very accurate predictions. However, the number of pattern rules is usually very high, being difficult to analyze the system's behavior analytically and,

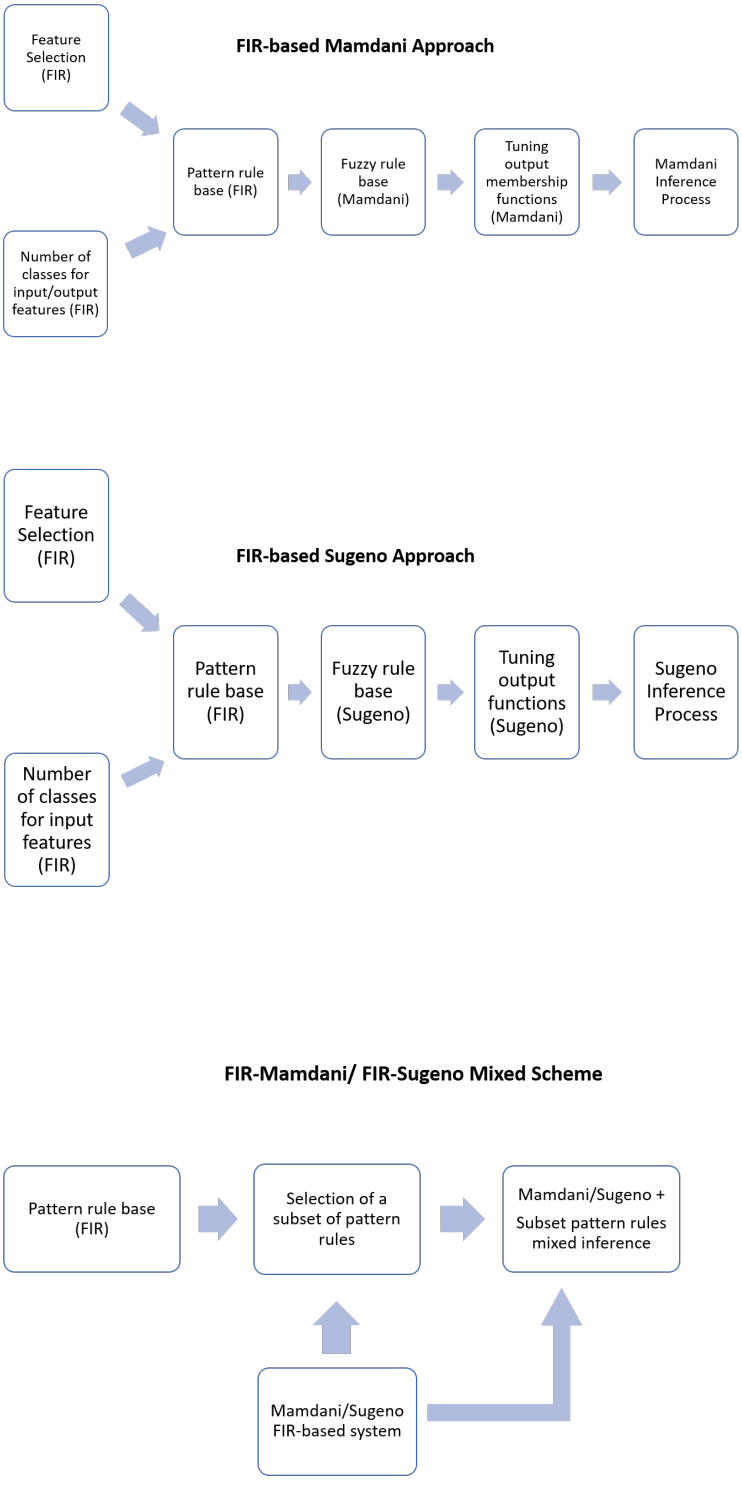

Figure 3: Schema of the FIR - Classical fuzzy system hybridizations.

therefore, not being useful for decision making and explainability.

In this work, four different hybridization algorithms are implemented to deal with this limitation of FIR models. Figure 3 presents schematically each of these combinations between FIR and classical fuzzy rule-based systems. FIR-based Mamdani and Sugeno approaches are classical Mamdani and Sugeno models that take as initial parameters the number of classes that the FIR model has chosen as best discretization (number of classes for input and output variables in Mamdani and the number of classes only for the inputs in Sugeno). Moreover, the Mamdani and Sugeno set of fuzzy rules are derived directly from the pattern rule base obtained by the FIR approach. Once the initial Mamdani/Sugeno model is created (i.e. discretization of the input and output variables and fuzzy rule base), the tuning process takes place to adapt the rules and the output as much as possible to the training data available for the problem under study. Finally, the Mamdani/Sugeno model is ready to perform its specific inference process.

Therefore, FIR-based Mamdani and Sugeno approaches obtain a classical Mamdani/Sugeno model, respectively, and their inference process is classical of all Mamdani/Sugeno. This means that the set of pattern rules, that can contain different levels of uncertainty (i.e. different thicknesses in the space) are converted to a Mamdani/Sugeno surface. On the one hand, this implies that the number of fuzzy rules that describe the behavior of the system is reduced. Therefore, it is a much simpler model, which allows the reasoning process to be followed and, therefore, a very good approach for decision making. On the other hand, if the real system contains a high level of uncertainty, these models eliminate a large part of it, reducing considerably the prediction accuracy.

FIR-Mamdani and FIR-Sugeno mixed schemes try to avoid as much as possible this last drawback of FIR-based Mamdani and Sugeno approaches. To this end, the uncertainty lost by the Mamdani and Sugeno models (represented as surfaces), is identified and captured as a subset of pattern rules. That is, the new models try to keep a small portion of FIR pattern rules besides the classical Mamdani/Sugeno models available. In this way, the mixed scheme can offer explanations of the inference process since the set of fuzzy Mamdani/Sugeno rules are available and, also, can explain the special behavior captured by the small set of pattern rules, where the system contains the larger level of uncertainty. It is expected that these models will be able to obtain better prediction

accuracy than FIR-based Mamdani/Sugeno approaches, but at the same time, provide explainability and is useful for decision making. In this project, all these approaches are implemented, tested, and applied to four different real problems, to study how they behave and extract some conclusions.

1.4 State of the Art

The methodology of Fuzzy Inductive Reasoning (FIR) has proven to be a tool with great capacity to identify, model and simulate complex dynamic systems using pattern rules based [4]. These pattern-based rules allow capturing the behavior of complex dynamic systems with a good degree of accuracy.

The basis of fuzzy systems is the theory of fuzzy sets, originally proposed in [5], which provides a strict mathematical framework for transferring imprecise conceptual activities to the necessary operational calculation [6]. However, a disadvantage of the fuzzy logic is the lack of fixed rules to decide the optimal membership functions, their range and degree of evaluation [7].

Fuzzy systems are used when the processes are more complex for analysis with conventional quantitative techniques or when the information on the process is qualitative, inaccurate or uncertain [8]. A fuzzy inference system can model the quality aspects of the human knowledge process and, therefore, without undertaking a precise analysis [9].

In order to construct a fuzzy inference system, two types of information can be used: linguistic data obtained from human experts or measured numerical data [10]. Although the first type of knowledge is natural in fuzzy systems, it exists with great difficulty during the phase of acquisition of knowledge through interviews with human experts [11].

Moreover, the methods that exist to transform knowledge and the experience of a human being into a base of rules for a fuzzy inference system cannot be completely automated due to the great need to use heuristic knowledge during the adjustment process. However, it is difficult to apply fuzzy linguistic systems to problems with many input variables because the number of rules grows exponentially and it is difficult to verify

their continuity, completeness and consistency [7]. It is necessary to use effective methods to tune the membership functions, as well as to minimise the error of selection or maximise the index of compliance [9].

A number of methods of automatic rule construction have been proposed based on numerical information [9, 10, 12, 13, 14]. The topic seems to be approached in parallel with substantially different proposals rather than developing on the basis of ideas fully accepted by the scientific community. The present thesis proposes an approach based on fuzzy inductive reasoning.

Different methods of automatic construction of fuzzy rules have been analyzed. These methods can be classified into three groups according to the paradigm used: genetic algorithms, artificial neuronal networks and fuzzy systems.

Genetic algorithms (GA) can be used to tune the parameters of a fuzzy system. In the literature there are different works in this line, such as [12], in which the authors try to tune more than one parameter simultaneously, using a GA for the simultaneous design of the membership functions and fuzzy rules. In another work, Ishibuchi et al. proposes a GA to select a small number of fuzzy rules [10]. This method assumes the existence of an extensive base of fuzzy rules from which it tries to eliminate all those rules that are not essential. Another search proposes the extraction of fuzzy rules from the input/output data using a GA [15]. Artificial Neural Networks (ANN) are an alternative paradigm that can be used to tune the parameters of a fuzzy system. In [16] a scheme of neuro-fuzzy control with self-learning is presented. It proposes an artificial neural network (ANN) with robust, fault-tolerant and adaptive learning, but excluded for dynamic systems with delays. In [13] an incremental learning ANN is proposed, which allows to acquire new knowledge without affecting the existing one. This method has no parameters, the same adaptive ANN is an inference system that generates the values of the output.

In [14], an ANN that learns from IF-THEN rules obtained from experts is presented. It is an ANN with a multilayer architecture that tunes its weights to imitate the rules provided by the expert. Another outstanding proposal is the ANFIS method (Adaptive-Network-Based Fuzzy Inference System) by Roger Jang [9], which is acquired partly from numbers and partly from an expert.

In [17, 18], a method for the control of non-linear dynamic systems with ANN is proposed. A further work in this direction and under this paradigm is presented in [19], where it uses a Dynamic-Adaptive Fuzzy Neural Network (D-AFNN) to learn the dynamics of systems. It is a Sugeno type neuro-differential inference system with polynomial functions.

There are other proposals in the literature that have the same purpose, but use substantially different methods. Such is the case of [20], where it presents a method based on the transition from state cells to fuzzy hypercubs. Given a set of data with vagueness and uncertainty, this method allows the generation of rules that guarantee the stability and robustness of fuzzy dynamic systems in a tank cycle.

Another work proposes to represent a fuzzy system using a matrix formulation [21], which allows to give a unified treatment to the different types of rules and inference mechanisms proposed. The disadvantage of this method is that it starts from the fact that there is a fuzzy system that can be represented in a matrix manner. Along the same line, [22] presents a method that uses the theory of directed graphs labeled to represent the cause-effect relationships between the variables of a system.

Within the paradigm of fuzzy systems, several methods of automatic rule construction have been proposed. There is a line of research that consists of activating and inhibiting fuzzy hypercubs and, from them, generating the rule base. It is initially proposed in [23], where an ANN is used. In [11] a fuzzy system is presented that pursues the same objective, but with a much lower computational cost.

The work presented on [24] is based on Abe and Lan ideas [11], and focuses on the extraction or selection of characteristics to determine the most relevant fuzzy hypercubs, under the assumption that, for any given set of hypercubs, there is a subset that contains the most relevant characteristics of the original set of hypercubs.

A further method, within the paradigm of fuzzy systems, is the proposed by Lu and Chen [25] with which the set of rules is generated. This method is composed of two algorithms. The first allows tuning the membership functions and the second generates the set of fuzzy rules. Another method that only tunes the membership functions of a fuzzy system is proposed in [7].

The method proposed by Nozaki et al. [26, 8] for the construction of multi-diffusion rules is simple and powerful, but it seems to be very heuristic. In [25] the authors present a method to enhance the fuzzy control by modifying the set of rules, but this requires the existence of the rule base. In [27] an adaptive algorithm to learn the behavior of a system at an intermediate level is described; this set of fuzzy rules is then converted into a conventional set of rules.

Article [28] proposes a method for extracting fuzzy rules from examples. This method extends its algorithm to non-fuzzy induction.

The method proposed by Takagi and Sugeno at [29], based on multidimensional fuzzy reasoning, is an excellent way to build fuzzy models of dynamic systems. In [30] this method is used to identify the structure of a system. In [31], an automatic tuning algorithm of several parameters of a typical fuzzy system is presented.

One element that stands out in this bibliographic study is that the proposed methodologies do not take into account the selection of variables that govern the behavior of the system under study. In almost all cases, a very small set of variables is used and they do not deal with the temporary delays that must be considered and that are fundamental in dynamic systems. In general terms, the methods proposed assume simple preconceived structures trying to adjust the parameters of the fuzzy system or the membership functions, or the rule weights and the number of fuzzy rules applied.

The results of this study can be summarised as follows: there is no method that complements the requirements of the problem that is to be solved in this research work. Therefore, the starting point of this study is based on the pattern rules set generated by the FIR methodology. The number of classes and the membership functions are already defined by the FIR model previously identified. Moreover, these pattern rules base have some particular characteristics that can be used, as will be seen below.

2 Project Scope

This research project will implement a fuzzy prediction model tool useful for decision-making and decision support. This tool will include 5 different hybrid fuzzy modelling approaches that will offer different tradeoffs between prediction accuracy and model understandability. The code will be developed from scratch in MATLAB and the FIR parameter files required will be obtained from the execution of the FIR methodology.

The project would be divided into the following stages:

- ➔ **Stage 1:** Study and understand fuzzy systems and the Mamdani and Sugeno approaches.
- ➔ **Stage 2:** Understand the foundation and working of FIR methodology and obtain the required intermediate files from the VisualFIR environmet.
- ➔ **Stage 3:** Implement the initial models of FIR-based Mamdani and FIR-based Sugeno using the generated intermediate files.
- ➔ **Stage 4:** Tune Sugeno using different possible algorithms.
- ➔ **Stage 5:** Tune Mamdani using different possible algorithms.
- ➔ **Stage 6:** Identify and extract the data points which increase the uncertainty in the original dataset and aggregate them as a subset to the tuned models.
- ➔ **Stage 7:** Test and validate all the the code implemented with small datasets to check that all the algorithms are working well.
- ➔ **Stage 8:** Application of the hybrid fuzzy models to four different datasets.
- ➔ **Stage 9:** Analysis and evaluation of the results obtained on the 4 real datasets.

2.1 Objectives and Requirements

2.1.1 Main Objective

On the one hand, to explore and understand the world of fuzzy logic and fuzzy systems and study this branch of Artificial Intelligence that I did not know before. On the other hand, to develop different types of fuzzy models that address the tradeoff between

forecast precision and model understandability. The fuzzy models should deal with robustness, interpretability and accuracy.

2.1.2 General Objectives

- To capture the systems behavior.
- To provide a mixed prediction scheme.
- To incorporate and record the uncertainty in data.
- To be interpretable on the prediction process.
- To make ease the user's decision-making process.
- To make the program efficient so as to reduce it's time complexity.

2.1.3 Requirements

- A workstation with sufficient compute capability.
- MATLAB 2018b or higher version with Fuzzy toolbox, Optimization library and Graphical Interface libraries installed.
- A dataset with well-defined input and output values to be trained and tested on.
- Prior knowledge on Fuzzy systems and models.

2.2 Risks and Limitations

- **Problem Complexity**
  Due to the project's inherent nature, the space allocated and the run time of the application depends how large the dataset is and how well the program code is written. Since the dataset size cannot be determined by the user, defining a poor code will result in large time and space complexities.
- **Time Management**
  The project to be developed has a limited time period. So, to complete the project successfully efficient planning and management has been done prior to the initial

development stages. Tasks have been given appropriate deadlines and meetings have been conducted in regular intervals. Before the confinement period, I had a weekly meeting with my tutor. During confinement period, e-mail communications have been produced each three or four days, and skype meetings have been performed when was needed.

- **Bugs**
  This project would require files generated by a pre-existing FIR project. Since the two projects need to be integrated, there is a lot of possibilities for bugs and poor working of the application. Each step must be validated to avoid this consequence.
- **Computational Power**
  Since the project's run time is proportional to the amount of data points, a workstation with the required amount of RAM and GPU memory must be utilised.

2.3 Methodology

Agile methodology[32] is used for the development of this project. A subset of Agile method, Scrum methodology (Figure 4.), is chosen for the development process since it is well suited for projects with shorter deadlines. Unlike other methodologies, Scrum process provides flexibility and faster development. Even though Scrum is designed for team-driven projects, it is also suited for solo projects.

The development process of the project is divided as follows:

- **Initial Phase:** The planning, study, analysis and design of the software are done in this phase.
- **Iterative Phase:** This phase consists of the actual implementation of the project. It is divided into:
  - Analysis of sprint backlog
  - Implementation
  - Integration
  - Testing

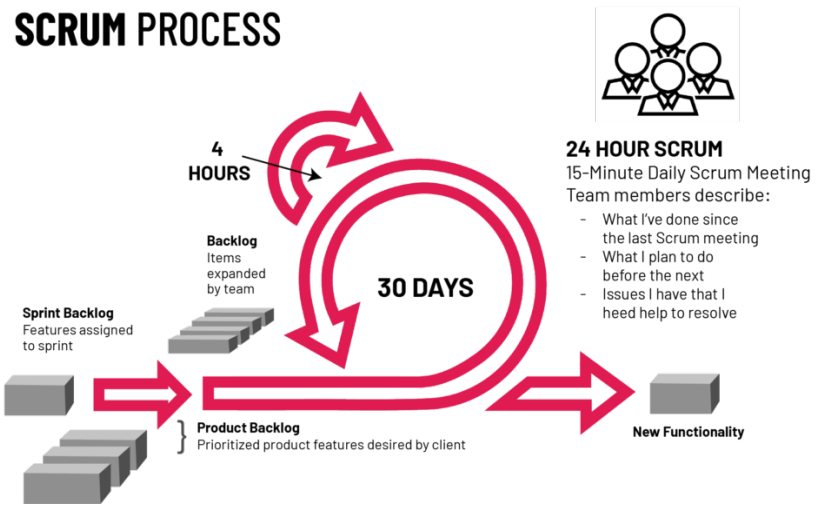

Figure 4: Scrum process.

Each iteration would be followed by a regular meeting, where the tasks that have been completed, the succeeding task and the issues encountered will be discussed.

- **Final Phase:** In this phase the project would have been completed and all the specified objectives and requirements would be met.

3 Project Planning

3.1 Duration

The project duration is estimated to be 5 months, starting from 17th February 2020 to 3rd July 2020 with a total duration of 138 days. The total duration of the project is 420 hours, that is an average of 3 hours per day. The final review (defence) of the project is scheduled on 3rd July 2020.

3.2 Task Definition

The project is divided into the following phases, each subdivided into tasks as listed in Table 1.

- **Project Management**
  - ➔ **P1-** Context and Scope: Define the context and scope of the project in MS Word.
  - ➔ **P2-** Time Planning: Define a structured plan and schedule each task by allocating the required amount of time to it using MS word and Gantter tool.
  - ➔ **P3-** Budget and Sustainability: Allocate the required budget for each task and develop a sustainability report using MS Word.
  - ➔ **P4-** Project Definition: Integrate the documents generated in P1, P2, P3 to define the whole project.
  - ➔ **P5-** Meetings: Weekly meeting with the director of the project to discuss the task being implemented at hand and also resolve of any issues faced during execution. This also means that the agendas discussed in previous meetings to be completed and updated to the director.
- **Analysis**
  - ➔ **A1-** Study of research topic: Fuzzy Logic and Fuzzy Inductive Systems are studied in depth to understand the concepts to be implemented.

➔ **A2-** Study of existing solutions: Existing fuzzy models (Mamdani and Sugeno) are studied to be incorporated in the project.

➔ **A3-** Study of FIR: FIR application is studied and the different files generated by the program are analysed.

➔ **A4-** Defining Functionalities: The functionalities of the project along with its objectives are defined.

- **Design, Implementation and Testing**

  ➔ **D1-** VisualFIR: Use the given dataset and generate the intermediate files, for each fold, using the VisualFIR environmet.

  ➔ **D2-** Mamdani and Sugeno: Use the files from the FIR application to build a rule base for Mamdani and Sugeno models.

  ➔ **D3-** Tuning and mixed prediction: Tune the above models and use a subset of the dataset to perform a mixed prediction.

- **Verification and Documentation**

  ➔ **V1-** Verification: Verify if the project satisfies all the requirements.

  ➔ **V2-** Documentation: Document the whole project into a thesis, explaining the theory and procedure behind the work.

  ➔ **V3-** Oral Presentation: Prepare for the final defence of the project.

Task		Hours	Dependencies
Project Management			
P1	Context and Scope	10	-
P2	Time Planning	10	-
P3	Budget and Sustainability	10	-
P4	Project Definition	10	**P1, P2, P3**
P5	Meetings	50	
Analysis			
A1	Study of research topic	25	-
A2	Study of existing solutions	20	**A1**

Task		Hours	Dependencies
A3	Study of FIR	10	**A2**
A4	Defining Functionalities	10	**A3**
Design, Implementation and Testing			
D1	VisualFIR	30	**P4, A1, A2, A3**
D2	Mamdani and Sugeno	60	**D1**
D3	Tuning and mixed prediction	130	**D2**
Verification and Documentation			
V1	Verification	10	**D3**
V2	Documentation	20	**V1**
V3	Oral Presentation	15	**V2**
Total		**420**	

Table 1: Time allocation.

3.3 Resources

The resources required for the development and implementation of the project are listed in Table 2.

Resource	Use	Tasks
Laptop	Studying, documentation, coding and processing.	All
MATLAB	Writing and executing project scripts.	D1, D2, D3, V1
MS Office	Documentation and presentation.	P1, P2, P3, P4, P5, V2, V3
Google Drive	For storing datasets and documents.	All
Google Chrome	Browser for searching and gaining information.	All
Gmail/ FIB mail	To contact the director for meetings and issues if any.	P5
Gantter	For generating gantt chart.	P2

Resource	Use	Tasks
MikTex	Latex editor for generating thesis document.	V2

Table 2: Resources required.

3.4 Gantt Chart

The following diagram, Figure 5, shows the Gantt chart of the project.

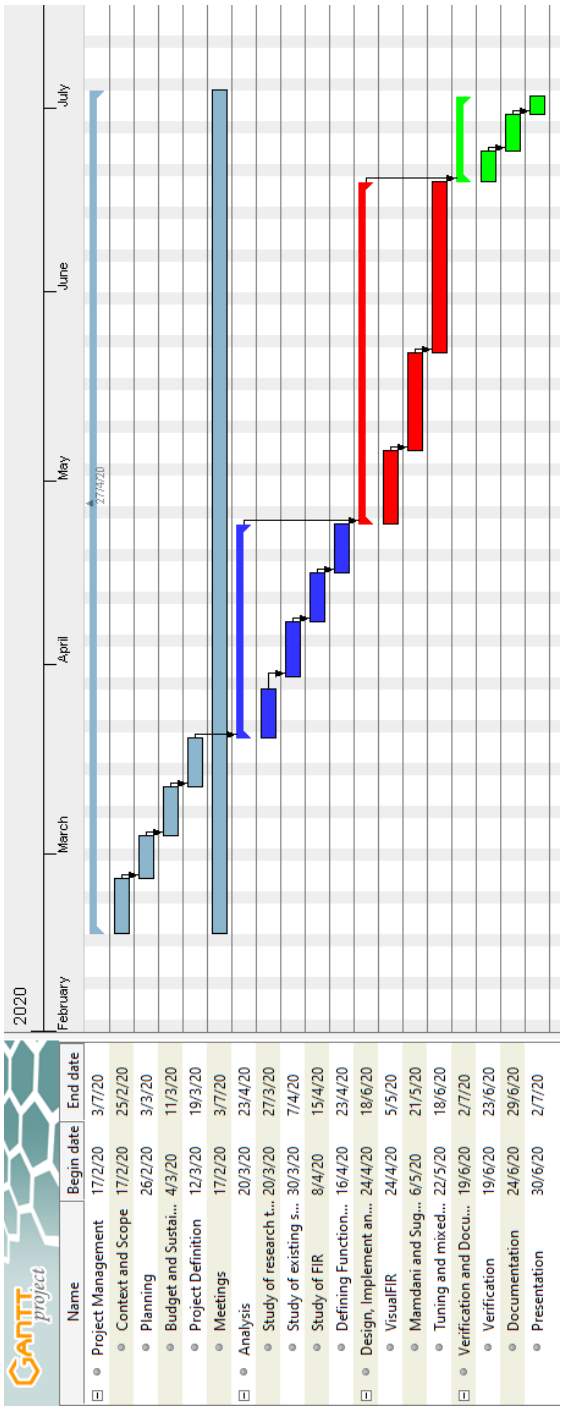

Figure 5: Gantt Chart.

3.5 Risk Management

The possible risks and obstacles that could be encountered during the course of this project were discussed in Section 2.2. The following are the possible strategies that could be adopted to mitigate those risks:

- **Complexity**
  To keep in mind the complexity of the problem and improve the efficiency of the application, each code written is executed by tracking the running time and space utilised by the program.
- **Time Management**
  The project is well planned during the management phase and sufficient time is allocated for each task depending on the task's difficulty. If any task could not be completed due to personal reasons or due to any issues, that task will be given higher priority on the next day's load.
- **Bugs**
  Each program is executed with several datasets to test for any bugs. By running different tests on the program, bugs, if present, can be resolved.
- **Computational power**
  This risk is avoided by choosing a workstation which has sufficient compute capability to handle very large datasets. The choosen workstation has 8 GB RAM and 4GB AMD Radeon R5 M330 graphics card.

4 Estimation of Budget

4.1 Human Resource Budget

The project will be carried out by one student from FIB (Erasmus program), who will assume the roles required for the development of this project. The roles required for the project are listed in Table 3 with each role assigned their corresponding tasks and their costs. The average salaries of these roles were obtained by referring [33] to determine the cost per hour of each role in Table 3.

Role	Tasks	Cost per hour	Total Hours	Cost
Project Manager	P1, P2, P3, P4, P5, V1, V2, V3	€ 10	135	€ 1350
Analyst	A1, A2, A3, A4	€ 8	65	€ 520
Software Designer	D1, D2, D3	€ 9	60	€ 540
Programmer	D1, D2, D3	€ 10	90	€ 900
Tester	D2, D3	€ 9	70	€ 630
Total			**420**	**€ 3940**

Table 3: Human Resource Budget

Since Table 3 consists of the tasks assigned to each role, the budget for each task for the project can be calculated as shown in Table 4.

Task	Hours	Cost
P1	10	€ 168.75
P2	10	€ 168.75
P3	10	€ 168.75
P4	10	€ 168.75
P5	50	€ 168.75
A1	25	€ 130
A2	20	€ 130
A3	10	€ 130
A4	10	€ 130

Task	Hours	Cost
D1	30	€ 480
D2	60	€ 795
D3	130	€ 795
V1	10	€ 168.75
V2	20	€ 168.75
V3	15	€ 168.75

Table 4: Budget for each task.

4.2 Non-human Resource Budget

The non-human resources budget of this project refers to the generic costs which will be incurred during the different phases of development of the project. These are: hardware, software and license, and other costs which includes rent, gas, electricity, internet etc.

4.3 Hardware

The hardware cost is listed in Table 5, which is a laptop, as that is the only hardware requirement of this project.

Resource	Cost	Unit	Life	Cost per Hour	Hours	Cost
HP Pavilion 17t	€ 849.99	1	4	€ 0.25	420	€ 101
Total						**€ 101**

Table 5: Hardware Resources Budget.

4.4 Software and Licences

All software used in this project are open-sourced, that is free of cost, which are: Google Drive, Google Chrome, Gmail, Git hub, Gantter and MikTex. Even though MS Office requires licensing and is used for this project, it comes pre-installed with the laptop and hence no cost is incurred. Similarly, MATLAB is a licensed software, but a student version can be obtained for free from FIB.

4.5 Other Costs

The other costs include the indirect expenses which are necessary for the development of the project, such as those listed in Table 6. The costs of these resources are expressed in cost per month, as they are generally billed in a monthly basis.

Resource	Cost per month	Months	Cost
Rent	€ 350	5	€ 1750
Electricity	€ 80	5	€ 400
Internet	€ 36	5	€ 180
Gas	€ 20	5	€ 100
Water	€ 15	5	€ 75
Food	€ 200	5	€ 1000
Total			**€ 3505**

Table 6: Other costs.

4.6 Total Costs

The total costs of the project estimates to be € 8546, as shown in Table 7. An extra factor of miscellaneous cost is added to the list to take care of unexpected expenses.

Cost Type	Cost
Human Resources	€ 3940
Hardware	€ 101
Software and Licences	-
Other Costs	€ 3505
Miscellaneous	€ 1000
Total	**€ 8546**

Table 7: Total Costs.

4.7 Budget Management

The budget of this project is divided into human and non-human based resources and is estimated on a timely basis. After each iteration during the development of the project, a budget analysis will be done to calculate the cost incurred during that phase. The analysis will be done based on the following formulas:

- ➔ Real Cost= Calculation of the actual cost for a task.
- ➔ Cost deviation= Estimated Cost – Real Cost.

With completion of each task, the total number of worked hours and other non-human resource expenses will be taken into account to arrive at the actual cost of the task (Real Cost). The cost mentioned for each task in Table 4 will be used as Estimated Cost to calculate the Cost Deviation. If the cost deviation is negative, it will mean that the actual cost of the task was greater than expected. Then to cover the extra expenses, miscellaneous budget will be used. If the cost deviation is positive, it will mean that the actual cost was lesser than the estimated one. In this case, the extra leftover budget can be passed on to the succeeding task or be added to the miscellaneous budget.

5 Sustainability

5.1 Self-evaluation

In each project there exists an element, which is the sustainability component, which plays a vital role in the project's development. This component of the project creates an impact in three different ways: economic, environmental and social. By studying the sustainability of the project along with its impact on the end users, more ideas can be gathered to improve the current project to make it more sustainable. Thus, the project must also be developed with an economic, an environmental and a social perspective.

5.2 Economic Dimension

An estimation of all the costs necessary for the development of this project were laid out in the previous section. The budget cover human resources and non-human resources like hardware required, software required, rent, electricity, food, miscellaneous costs etc. It can be seen that most of the budget is allocated to human resources.

A decision-making system which captures the uncertainty in data and still remains interpretable is a need of the hour. This cost-effective project might provide the required solution in a decision-making problem when deployed in real-time.

5.3 Environmental Dimension

The project requires the usage of a single laptop for its components and the environmental impact of a single laptop is minimal. Other environmental impacts include, electricity, water, gas etc. which can also be considered negligible, since it is the consumption of a single student.

After the completion of the project, this can be used as a decision-making system even for environmental data. For example, this project can be used in an industrial process to take decisions which have less effect on the environment.

5.4 Social Dimension

Since this project provides an interpretable system, a system whose decision can be explained, it can be used to understand the problem being dealt at hand even more effectually. This a necessary solution, as most of the existing solutions are less explainable and with better understanding of the process real world problems can be dealt with much ease.

6 Fuzzy Systems Approaches

This section discusses the most commonly used classical fuzzy systems which include: Fuzzy Inductive Reasoning (FIR) approach, Mamdani fuzzy inference system and Sugeno fuzzy inference system.

6.1 Fuzzy Inductive Reasoning

The Fuzzy Inductive Reasoning (FIR) methodology emerged from the General Systems Problem Solving (GSPS) architecture developed by George Klir [34]. FIR is a data driven methodology based on systems behavior rather than structural knowledge. It is a very useful tool for modelling and simulating those systems for which no previous structural knowledge is available [35]. FIR is composed of four main processes (Figure 6), namely:

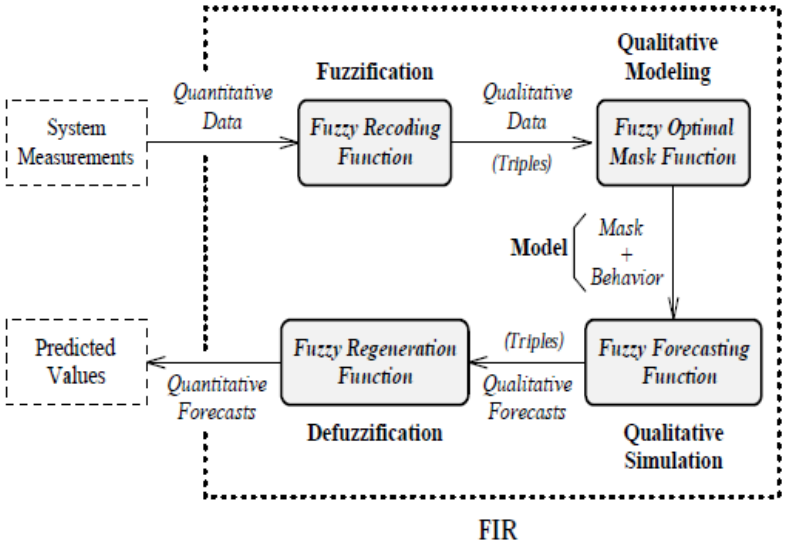

Figure 6: Fuzzy Inductive Reasoning (FIR) architecture.

- **Fuzzification:** The fuzzy recoding function converts quantitative values into qualitative triples. The first element of the triple is the class value, the second element is the fuzzy membership value, and the third element is the side value. The class value represents a coarse discretization of the original real valued variable. The fuzzy membership value denotes the level of confidence expressed in

the class value chosen to represent a particular quantitative value. Finally, the side value tells us whether the quantitative value is to the left, to the right or to the center of the peak value of the membership function. The side value, which is a peculiarity of FIR methodology since it is not commonly introduced in fuzzy logic, is responsible for preserving the complete knowledge in the qualitative triple that had been contained in the original quantitative value.

- **Qualitative Model Identification:** The fuzzy optimal mask function realizes the process of qualitative modelling. It is able to establish qualitative relationships between different variables of the model. It does so by a process of search in the discrete space of the class values. This mask identifies casual relations (spatial and temporal) between the input variables.
- **Fuzzy Forecasting:** Fuzzy simulation is performed by means of the fuzzy forecasting function, which is able to predict future qualitative outputs (qualitative triples) from past similar experiences. Fuzzy simulation interpolates between previous occurrences of similar behavioral patterns, and uses the interpolated values to extrapolate the output variable.
- **Defuzzification:** The fuzzy regeneration facility implements the inverse process of the fuzzy recoding module. It converts qualitative triples back to quantitative values. Since fuzzy recoding preserves the complete information of the original quantitative value, an immediate cascade of a fuzzy recoding operation followed by a fuzzy regeneration operation restores the original signal without any error. This is a special feature of FIR particular dialect of fuzzy logic as most fuzzy logic signals lose information in the process of fuzzification, information that cannot be retrieved by means of defuzzification.

To understand the different algorithms developed in this project it is necessary to explain in a little bit more detail the Qualitative model identification and the Fuzzy forecasting processes of the FIR methodology.

6.1.1 Qualitative Model Identification

At this point, the dataset recorded from the system has been fuzzified to a qualitative data stream. In the process of modelling in FIR, the optimal mask function is used,

which is responsible for finding causal, spatial and temporal relations between variables that offer the best likelihood for being able to predict the future system behavior from its past, thereby obtaining the best model (composed by the mask and the pattern rule base in the FIR terminology) that represents the system. A mask represents the possible relationships among the qualitative variables. Let us introduce the concept of a mask using a simple example composed of two inputs, *u1* and *u2*, and one output, *y*. A possible mask for a system with two inputs and one output is shown in Figure 7.

x / t	u_1	u_2	y
t - 2δt	-1	0	-2
t - δt	0	-3	0
t	-4	0	+1

Figure 7: Example of a mask for a system with two inputs (*u1* and *u2*) and one output (*y*).

The negative elements in the matrix of Figure 7 are referred to as m-inputs (mask inputs). They denote input arguments of the qualitative functional relationship. They can be either inputs or outputs of the system to be modeled, and they can have different time stamps. The above example contains four m-inputs. The sequence in which they are enumerated is immaterial. The single positive value denotes the m-output, and the zero elements represent unused connections. In the above example, the first m-input corresponds to the input variable *u1* two sampling intervals back, $u1(t-2\delta t)$, whereas the second m-input refers to the output variable *y* two sampling intervals into the past,$y(t-2\delta t)$, etc.

How is a mask found that, within the framework of all allowable masks, represents the most deterministic state transition matrix? The optimal mask function searches through all legal masks of complexity two, i.e., all masks with a single m-input, and finds the best one; it then proceeds by searching through all legal masks of complexity three, i.e., all masks with two m-inputs, and finds the best of those; and it continues in the same manner until the maximum allowed complexity (a parameter) has been reached. Other search strategies have been developed, i.e. variants of hill-climbing

and genetic algorithms as well as statistical approaches based on cross-correlation and spectral coherence functions.

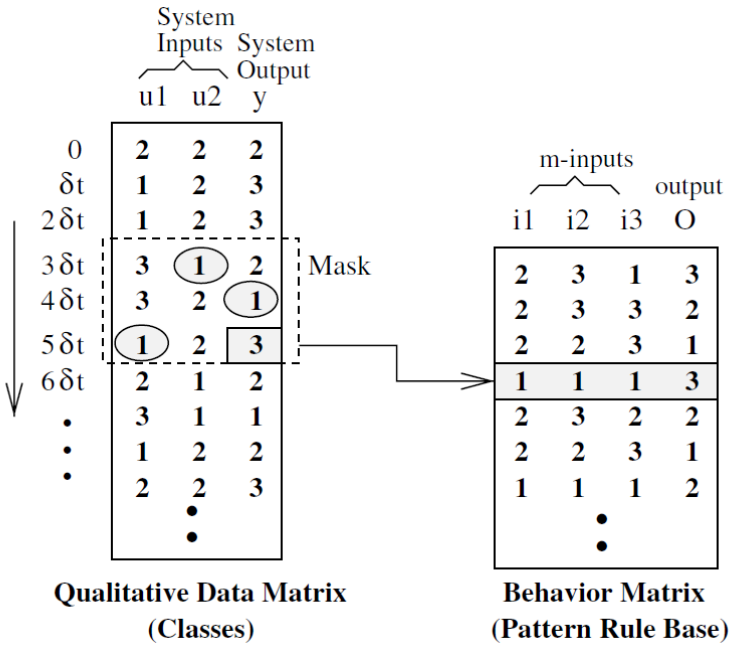

Figure 8: FIR pattern rule base construction.

The optimality of the mask is evaluated for the maximization of its forecasting power that is quantified using a quality measure, based mainly on Shannon entropy. Once the best mask has been identified, it can be applied to the qualitative data matrices that were previously obtained in the fuzzification process, resulting in a fuzzy pattern rule base that, in FIR terminology, is called the behavior matrix. Figure 8 shows the process of constructing the pattern rule base. The dashed box symbolizes the mask that is shifted downwards along with the class qualitative data matrix. The round shaded "holes" in the mask denote the positions of the m-inputs, whereas the square shaded "hole" indicates the position of the output. The class values are read out from the class qualitative data matrix through the "holes" of the mask and are placed next to each other in the behavior matrix (pattern rule base) that is shown on the right side of the Figure 8. For example, the shaded rule of this figure can be read as follows: "If the first m-input, i1, has a value of '1' (corresponding to 'low'), and the second and third m-inputs, i2 and i3, have also values of '1' (corresponding to 'low') then the output, o, assumes a value of '3'

(corresponding to 'high').

6.1.2 Fuzzy Forecasting

The FIR inference engine is based on a variant of the k-nearest neighbor rule. The forecast of the output variable is obtained as a weighted average of the potential conclusions that result from firing the k rules, whose antecedents best match the actual state. The prediction procedure is presented in the diagram of Figure 9 for an example containing three inputs and one output.

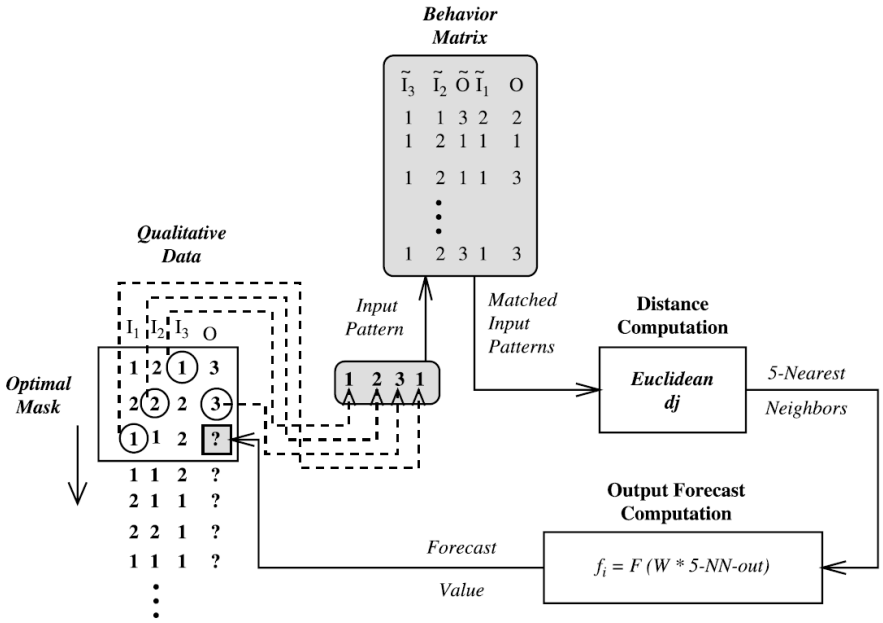

Figure 9: FIR forecasting process diagram.

The optimal mask is placed on top of the qualitative data matrix in such a way that the m-output matches with the first element to be predicted. The values of the m-inputs are read out from the mask, and the behavior matrix (pattern rule base) is used to determine the future value of the m-output, which can then be copied back into the qualitative data matrix. The mask is then shifted further down by one position to predict the next output value. The contribution of each neighbor to the estimation of the prediction of the new output state is a function of its proximity. This is expressed by giving a distance-weight to each neighbor, as shown in Figure 9.

For a detailed description of the FIR methodology the reader is referred to publication [35].

6.2 Mamdani Fuzzy System

This system was proposed in [36]. Basically, it was anticipated to control a steam engine and boiler combination by synthesizing a set of fuzzy rules obtained from people working on the system, that is rules were obtained from domain experts. After obatining the set of fuzzy rules from the experts, the input would be made fuzzified using the membership functions. The rule strength is then obtained by combining the fuzzified inputs according to fuzzy rules. By combining the rule strength and the output membership function we obtain the consequent of the rule using the *min* operation. The consequent of all such rules are combined to obtain the final output distribution using the *max* operation. This distribution is defuzzified using the output membership function to get the crisp value. The whole Mamdani process is explained in Figure 10 where x and y are inputs and z is the output. Notice that in a Mamdani rule-based system, both the inputs and the output are fuzzy sets and are represented by membership functions.

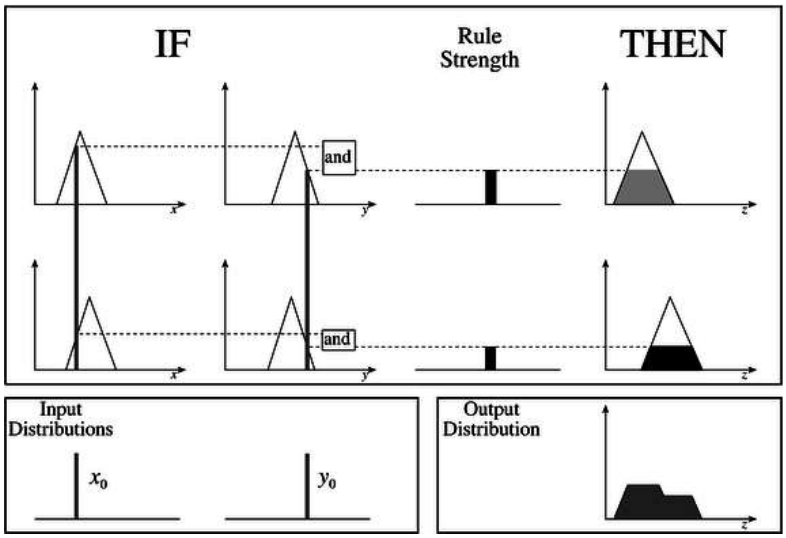

Figure 10: Mamdani inference process.

For a much more detailed explanation the reader is referred to [37].

6.3 Sugeno Fuzzy System

This model was proposed in [29] and has a specific rule format which is given below:

$$\text{IF } x \text{ is A AND } y \text{ is B THEN } z = f(x,y)$$

x and y are inputs with classes A and B respectively and z is the output which is determined using a function on x and y as $f(x,y)$. The Sugeno model differs from Mamdani in the consequent determination step where the Sugeno directly predicts the crisp output using a function. When $f(x,y)$ is a constant, the inference system is called a zero-order Sugeno model, which is a special case of the Mamdani system in which each rule's consequent is specified as a fuzzy *singleton*. When $f(x,y)$ is a linear function of x and y, the inference system is called a first-order Sugeno model. The overall output is the weighted average of each rule's outfunction $f(x,y)$, as shown in Figure 11. This process avoids the time-consuming methods of defuzzification which is necessary in the Mamdani model. For a much more detailed explanation the reader is referred to [37].

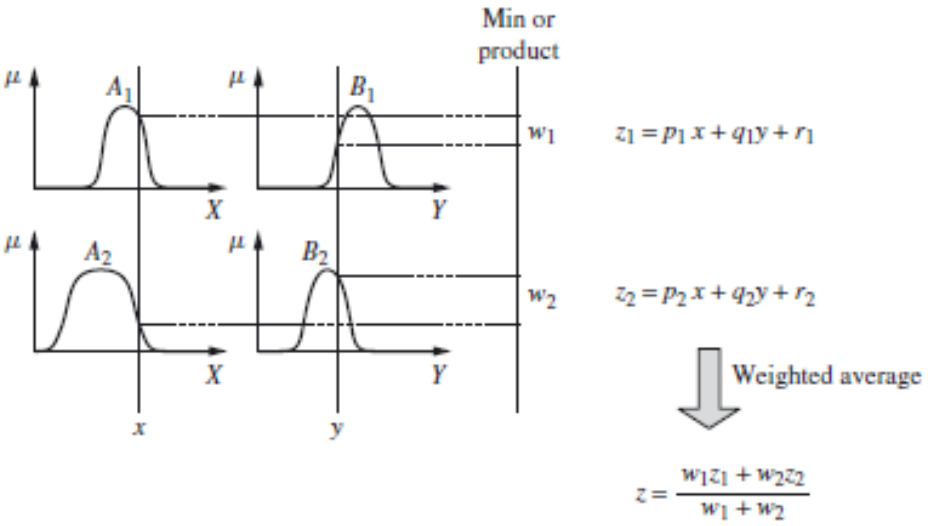

Figure 11: Sugeno inference process.

7 New hybrid Fuzzy System Approaches

This section focuses on explaining the different fuzzy systems approaches developed in this thesis, which was previously briefed in Section 1.3. The five modelling algorithms that conform the framework of this project are: FIR, FIR-based Mamdani, FIR-based Sugeno, FIR-Mamdani mixed scheme and FIR-Sugeno mixed scheme. The main idea is to expand the modelling capacity of the Fuzzy Inductive Reasoning (FIR) methodology, which emerged from the General Systems Problem Solver [34], allowing it to work with classical fuzzy rules. This is done by allowing the FIR model to generate a pattern rule base from the data provided in the form of a behavior matrix which comprises of classes, membership values, and its corresponding sides from the peak of the curve. The pattern rule base model is, usually, able to predict accurately future values of the modelled system. As already mentioned in chapter 1, it has been proved to be a good approach for modelling biological, medical, energy, e-learning and other systems[38, **?**, 39, 40] which is used option I of the framework.

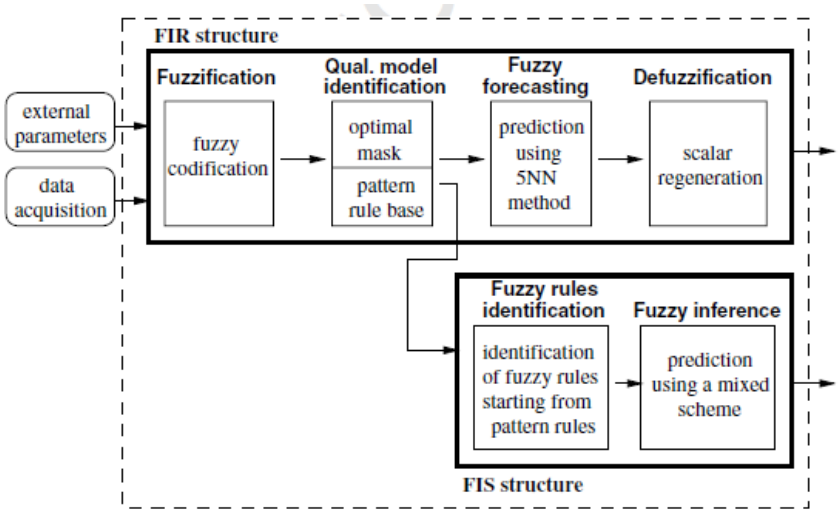

Figure 12: Overall architecture of FIR and classical fuzzy systems hybridization.

The pattern rule base generated from FIR is large and is far from interpretable. These rules also produce different output classes for equivalent inputs, in order to capture as much as possible, the uncertainty existent in the data. To alleviate these problems, the purpose of the current project is to generate automatically classical fuzzy rules models from the pattern rule base. The classical fuzzy rules models are much more interpretable

and useful for decision making, although it is expected that the prediction accuracy will be reduced. Therefore, Mamdani[36] and Sugeno[30] (options II and III) models are identified and generated from the pattern rule based (as can be seen in Figure 12 given in [41]), with the goal of making the rules more interpretable.

For the Mamdani model, for each group of rules, the average of the membership values of the consequents is computed. Then, the class of the consequent that has the greatest average is the one selected as the consequent of that Mamdani fuzzy rule. The consequent of each Sugeno fuzzy rule is obtained by computing the mean of the crisp output values associated to the pattern rules that have the same set of antecedents. These models are then further tuned using learning algorithms to improve the prediction accuracy.

Since the rules are transformed into a more interpretable format, i.e. a simple surface, the models will not be able to handle the inherent uncertainty present in the original data. So, a subset of pattern rules, which captures the main uncertainty of the original data, is aggregated with the classical fuzzy models. This step increases the robustness of the process and results in a mixed scheme prediction approach. In this project, first, the generation and tuning of the classical fuzzy rule models (Mamdani and Sugeno) are implemented and validated. Second, the mixed prediction scheme is designed and implemented. This corresponds to options IV and V of the framework. The whole approach will be tested with a set of benchmarks available in the UCI repository[42]. It is also expected to test it with a real data set.

7.1 FIR-based Mamdani and FIR-based Sugeno

The Mamdani and Sugeno are classical fuzzy models which are widely used in applications involving fuzzy logic. Mamdani system uses a fuzzy operation (min or max) on each rule based on the input, and the output of these rules are aggregated and defuzzified to obtain the crisp output. These rule bases are genrally defined by a domain expert thus making this method more interpretable. The Sugeno model is different from Mamdani, as the output membership functions of Sugeno are singleton, that is either linear or constant functions. Hence the Sugeno system has virtually no defuzzification process and directly produces the crisp output. This makes the model less interpretable

than Mamdani but computationally more efficient than it.The FIR-based Mamdani and Sugeno models differ only in their rule base generation process. The process of creating the FIR-based models are explained below.

7.1.1 Membership Function generation

Both FIR-based approahes use the FDadesModel.mat file generated from *VisualFIR* to obtain the set of pattern rules and the membership functions of the variables. The *mask* variable from the file is used to identify all the relevant input variables. The number of membership functions for each input variable is obtained from *VClass*. All membership functions are represented as a Gaussian curve and hence require a mean (c) and a standard deviation (σ) parameters to deffine the shape of the Gaussian function. A Gaussian curve is represented as:

$$f(x;c,\sigma) = e^{\frac{-(x-c)^2}{2\sigma^2}} \tag{2}$$

The landmarks (borders between neighboring classes) of these membership functions are present in *VFrom*, but these landmarks are marked at the level of 0.5 membership value and hence we need to use these values to obtain the c and σ of each class for each variable. The c of the membership function can simply be obtained as the average of the two landmarks of the class. To find σ we inverse Equation 2. For a variable with n membership functions, we use two equations to obtain σ. For the first and last membership functions we use:

$$\sigma = \sqrt{\frac{-(l_2 - l_1)^2}{2 \cdot \log(0.5)}} \tag{3}$$

where l_1 and l_2 represent the two landmarks of the membership function. For the rest of the $n-2$ membership functions we use:

$$\sigma = \sqrt{\frac{-(c - l_1)^2}{2 \cdot \log(0.5)}} \tag{4}$$

In this way the membership functions of each variable are defined.

7.1.2 FIR-based Mamdani Rule Base Generation

To complete the Mamdani model we need to create the fuzzy rule base which is obtained from *ba* and *mba* variables of FDadesModel.mat file. *ba* stores the classes of the pattern rule base and *mba* the membership values associated to these classes. This behavior matrix contains rules having antecedents (input variables of the rules) and consequents (output variables of the rules). The rule base is generated as given in Figure 13 ([41]) where for each rule, other rules with the same antecedents are identified and the average of the membership values for each type of rule is obtained. Then the rule with highest membership value is added to the Mamdani model. This completes the fuzzy rule base and the Mamdani model.

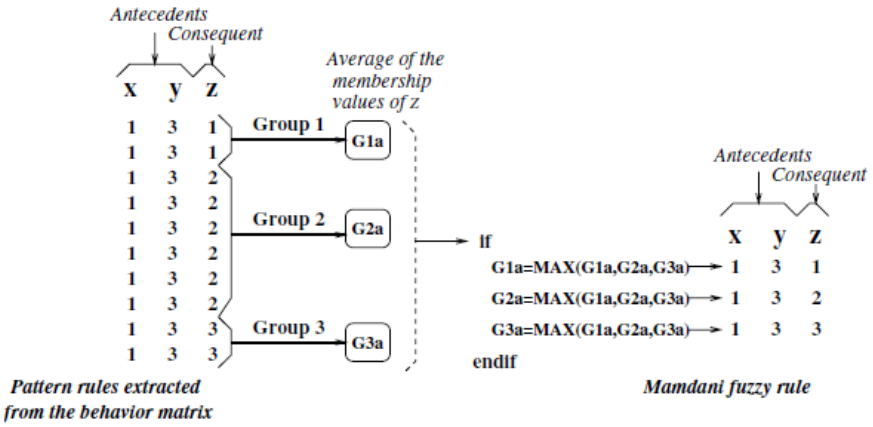

Figure 13: Mamdani rule base generation.

7.1.3 FIR-based Sugeno Rule Base Generation

For the Sugeno model, the process of identifying the input variables and the membership functions is the same as that of Mamdani. The difference is in the membership functions of the output variable and in the process of creating the fuzzy rule base. The output membership function of Sugeno is a singleton (i.e. a constant), which is created during the process of generating the rule base. As given in Figure 14 ([41]), the rules with similar antecedents are grouped and their corresponding crisp output is averaged. This

averaged crisp output is used a constant membership function and is the consequent of that rule.

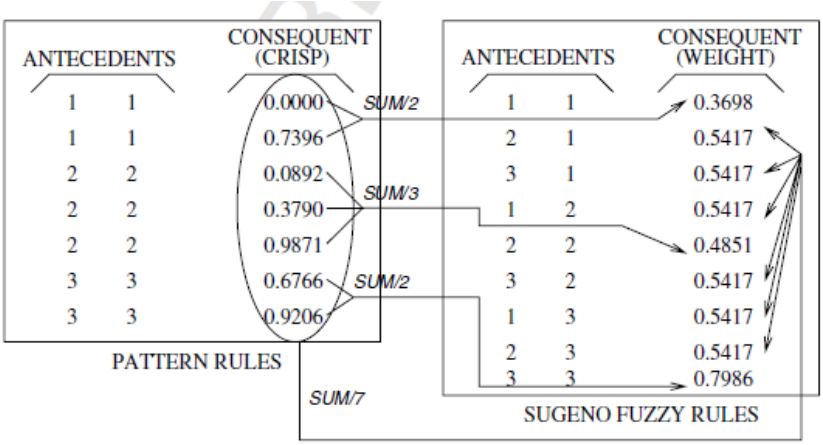

Figure 14: Sugeno rule base generation.

7.1.4 Tuning

After creating the model (Mamdani or Sugeno), tuning is performed on the rules and the output membership functions of the model. The tuning of input variables is disabled to maintain consistency of the model, i.e. the FIR discretization of the input variables should be kept to assure interpretability of the hybrid model. The model can be tuned with the training dataset on any one of the following methods which are available in MATLAB:

- Genetic Algorithm
- Particle Swarm
- Pattern Search
- Simulated Annealing

The tuned model obtained can then be used for prediction of output in test datasets.

7.2 FIR-Mamdani and FIR-Sugeno Mixed Schemes

The fourth and fifth models of this project are an extension of the FIR-based models previously discussed. The FIR-based Mamdani and Sugeno, by generating a compact rule base from the behavior matrix, helps in improving the interpretability of the inference system. But with a compact rule set, the model tends to handle uncertainty in data poorly. To alleviate this, a mixed prediction approach is used. The procedure for creating the mixed models is the same as that for creating FIR-based Mamdani or Sugeno until the tuning step. After tuning, a subset of rules from the original behavior matrix which are identified as having high degree of uncertainty are aggregated with the model.

The size of the subset of rules is defined by the user with the help of the parameter p, which takes values between 0 and 1 and represents the percentage of original FIR pattern rules to be retained. To obtain the rules with uncertainty, the output of training data is predicted using the tuned FIR-based Mamdani or Sugeno model. The error between the predicted and original output is calculated for each data point. The rules (ba), membership values (mba) and the side values(sba) are then arranged in decreasing order of their corresponding error values. Then the first $p\%$ rules, membership and side values are saved.

During prediction, these values are passed to the *regenerate* function in *VisualFIR* to defuzzify and obtain the original data points. The data points are then normalized between -1 and 1, to obtain the uncertain data points, using the following formula:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{5}$$

The mask is passed over the test dataset and the corresponding input variables and output variable are identified for each data point. The data points are normalized and for each test data point the closest data point from the subset of pattern rules (i.e. uncertain rule base) is chosen. The Euclidian distance between the two points is calculated and normalized between $[0,1]$ as follows:

$$d_{real} = \sqrt{\sum_{i=1}^{N} (x_{test_i} - x_{pattern_i})^2} \tag{6}$$

$$d_{norm} = \frac{d_{real}}{d_{max}} \tag{7}$$

where d_{max} is generally taken as 0.25. The obtained value of d_{norm} is used to determine the mixture level termed as f_{mix}. This is the value which determines how much the uncertain rule base influences the predicted output. It is obtained as follows:

$$f_{mix}(d_{norm}) = \begin{cases} 100 & d_{norm} \in [0, d_{min}] \\ \frac{1}{1-e^{-d_{norm}}} & d_{norm} \in (d_{min}, d_{max}) \\ 0 & d_{norm} \in [d_{max}, 1] \end{cases} \tag{8}$$

where d_{min} is genrally 0.01. For each test data point two outputs are predited. One is from the tuned FIR-based Mamdani or Sugeno model. Another is from the closest uncertain rule that is obtained previously. These two predictions and the f_{mix} value are used to produce the final prediction using the following equation:

$$y_{mix} = y_{pattern} \cdot \left(\frac{f_{mix}}{100}\right) + y_{fuzzy} \cdot \left(\frac{100 - f_{mix}}{100}\right) \tag{9}$$

The y_{mix} obtained is the final prediction of the mixed scheme model.

8 Design and Implementation

This project can be represented as a step-by-step process on an abstract level. This is best represented using a flow chart diagram shown in Figure 15. So the whole project can be divided into four steps that captures the whole picture, namely:

1. **Input:** This step includes providing the datasets and necessary parameters as input by the user. This is a very important point since not only the training and testing data from the problem to be studied are necessary, but also all the parameters related to the FIR model that are going to be the basis of the hybridizations of the four fuzzy approaches developed in this work.

2. **Modelling:** This comprises the creation of Mamdani or Sugeno models and their respective tuning process based on the method provided. It also includes the process of creating uncertain rule base (i.e. small subset of pattern rules) if specified.

3. **Prediction:** In this step the mixed prediction scheme is developed with both Mamdani and Sugeno options. Once available, the test dataset provided is evaluated and prediction results are produced. The prediction might be pure FIR, fuzzy rule-based inference (i.e. FIR-based Mamdani or Sugeno) or mixed prediction scheme (i.e. FIR-Mamdani or FIR-Sugeno mixed scheme).

4. **Error calculation:** This step includes the process of calculating the error between the original output and the predicted output.

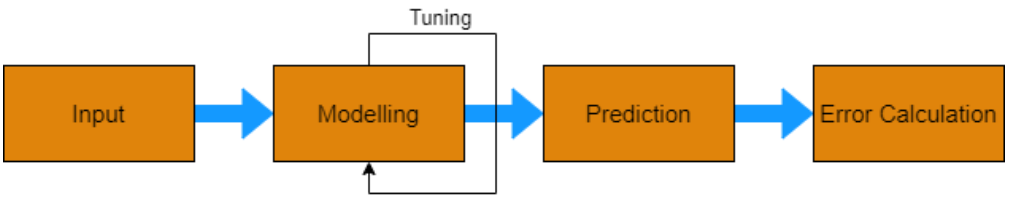

Figure 15: Flow Chart diagram.

Before continuing, it is important to accurately describe all the necessary information derived from the FIR model for the development of the different hybridization methodologies. In that way, it is much clear how the implementation has been done.

8.1 FIR Parameters Description

The FIR methodology is implemented in an environment called *VisualFIR* in MATLAB. This module generates 2 important .mat files which are crucial for this project. The first one is the Parametres.mat file which contains all the necessary global variables to be declared and initialized. The second one is the FDadesModel.mat file which contains all the necessary information, as MATLAB variables, from the dataset that is required for prediction. The required variables from this file are:

1. *VFrom:* This variable consists of the landmarks of each variable present in the dataset that define the different classes in which this variable has been discretized. The landmarks represent the values of that variable that correspond to the borders between neighboring classes, i.e. where membership values are 0.5.

2. *VClass:* This variable contains the list of the number of classes for each variable.

3. *Camps:* This variables holds the names of each variable.

4. *mask:* This variable is the key piece in identifying important rules which capture the spatial and temporal relations in data. This is a matrix of elements which contains one or more negative integers, one positive integer and some zeros. The negative elements correspond to the input variables and implies that these variables are related with the output variable which is denoted as a positive integer, which is the last element of the matrix. The zeros state that the variables in those places are not related to the output variable. The columns of the matrix represent the variables in the dataset, and the last column is the output variable. The number of rows represent the depth of the mask. The depth defines how much the output variable depends on the inputs and, also, on previous values of the same output, i.e. temporal causality. For example, for a mask of depth 3, the output variable at current time depends on 2 previous data points. In this way both spatial and temporal relations are captured among the variables.

5. *ba:* This variable is the behavior matrix obtained from the mask. The behavior matrix contains the class values of the set of rules which are determined from the qualitative data matrices by using the mask. The mask is passed over the fuzzified data and acts as filter to extract only the necessary variables to add to the fuzzy

rule.

6. *mba:* This holds the membership values of each variable of each rule in the behavior matrix.

7. *sba:* This variable is used to represent the side value of each variable of each rule. That is, to denote which side of the curve the given membership value of the data point lies on. If -1 then it lies on the left side of the cure, if 1 it lies on right side and if zero it lies in the maximum value of the membership function.

After obtaining these variables, the fuzzification, prediction and defuzzification of data must be performed. This is done by using the following functions from *VisualFIR*:

1. *recode:* This function is used to fuzzify the given dataset by specifying the number of classes and landmarks of each variable.

2. *forecast:* This function predicts the output by using the fuzzified values, the mask, behavior matrix, membership and side values.

3. *regenerate:* This function uses the predicted values to defuzzify and produce the final crisp output.

By using the above *VisualFIR* functions and the FDadesmodel.mat file, the output of test dataset can be predicted.

8.2 Implementation Steps

On a more concrete level, the project is designed based on an activity diagram. The steps shown in Figure 15, are broken down into smaller pieces to with each representing an activity as shown in Figure 16. The Input and the Error calculation step are not changed, while the Modelling step is broken into Mamdani/Sugeno creation, Tuning and Uncertain rule base creation. The Prediction step is split into Prediction using *VisualFIR*, Prediction using fuzzy model, Prediction using uncertain rule base and Mixed Prediction. The flow of these activities is totally dependent on the model option which is given as an input parameter by the user. But all these flows are merged together in the end to calculate the error.

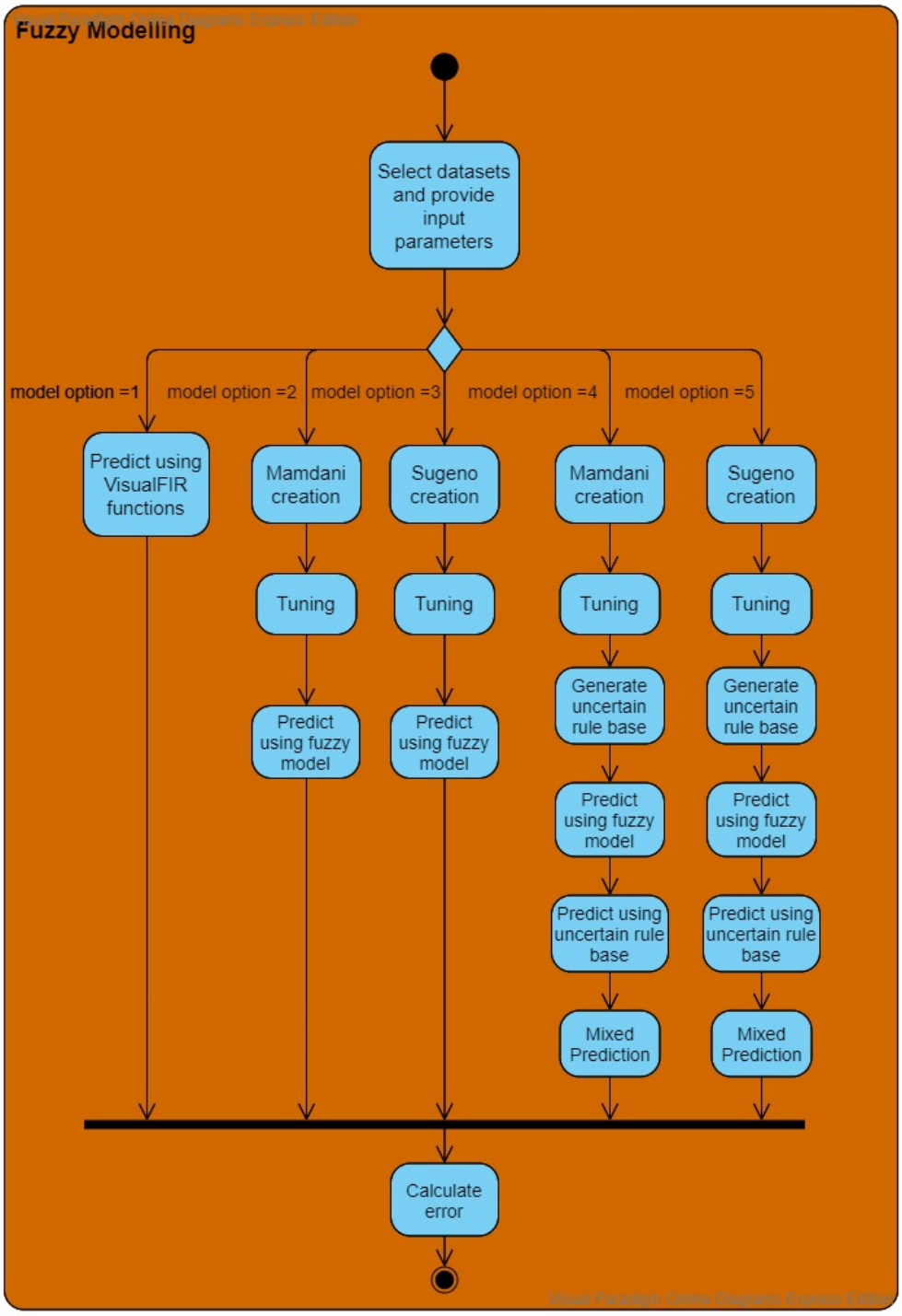

Figure 16: Activity diagram.

During implementation, each of the activities shown in Figure 16 are treated as individual modules thus making them independent and indivisible. Since the project is developed purely in MATLAB, each of these modules are written as *.m* script files. Before executing these scripts, the following MATLAB toolboxes must be installed:

- Fuzzy Logic toolbox
- Optimization toolbox
- Global Optimization toolbox
- Symbolic math toolbox
- Statistics and Machine Learning toolbox

These toolboxes provide the basic commands and functions that are used in the scripts. The script files created during the development of this project are listed as follows:

1. *prediction_scheme.m:* This file is the whole interface of the framework as it acts the center point connecting all other scripts. It accepts input parameters such as the FDadesModel.mat file, Parameters.mat file, training and testing datasets, model option, tuning method, iteration count and the *p* parameter for the mixed prediction. The other scripts are called based on these parameters.
2. *TestPrediction.m:* This script uses the *VisualFIR* functions to predict the output of test data.
3. *mamdani.m:* This file creates the FIR-based Mamdani fuzzy model by using the training dataset and the FDadesModel.mat file.
4. *sugeno.m:* This file creates the FIR-based Sugeno fuzzy model by using the training dataset and the FDadesModel.mat file.
5. *tune.m:* This script is used to tune the fuzzy model created given the tuning method, iteration count and training dataset as input parameters.
6. *uncertain_rules.m:* This script accepts the FDadesModel.mat file and the *p* parameter to generate the uncertain rule base.
7. *predict.m:* This file returns the predicted output from the tuned model for the

provided test dataset.

8. *uncertain_rules_predict.m:* This script returns the predicted output from the generated uncertain rule base for the provided test dataset.

9. *mixed_predict.m:* This file combines the fuzzy model prediction and the uncertain rule base prediction using the Equation 9.

10. *calculateRMSE.m:* This file returns the error value between the original and predicted output.

The interaction between these functions is through the *prediction_scheme.m* file. It accepts input parameters and calls other files. Hence this script acts as the main program and all other scripts are the subprograms. This relationship between the files is better represented as a call graph as shown in Figure 17. In this diagram, each link between the functions is labelled with a number and an alphabet. The number denotes the model option and the alphabet denotes the step in that process. For example 2b denotes tuning of the Mamdani model in *tune.m* and 5f represents the mixed prediction step of Sugeno model in *mixed_predict.m*. In this way we can see how the main program calls the different subprograms based on the option provided to it.

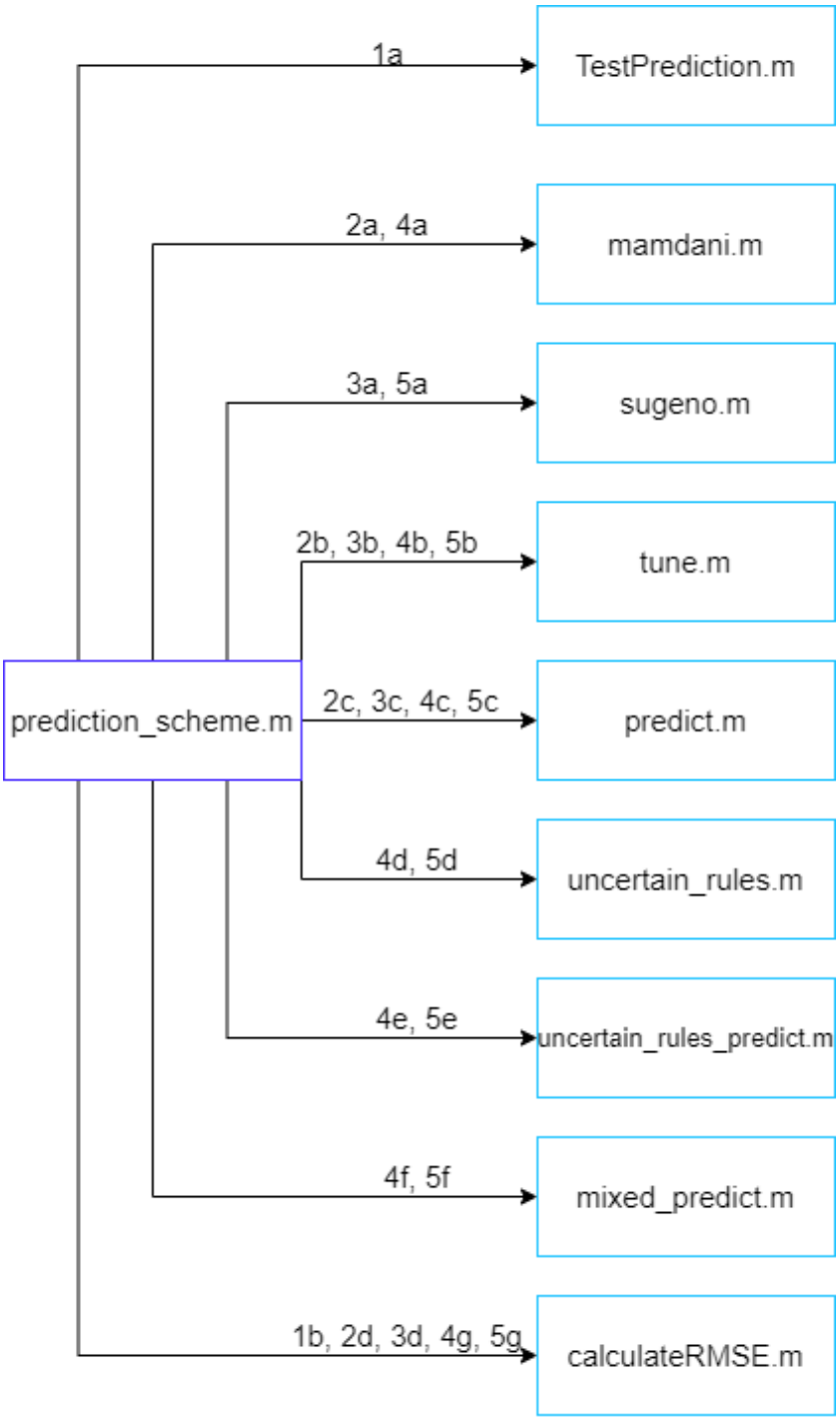

Figure 17: Call Graph.

9 Dataset Overview and Results

The following section discusses in brief, the different datasets used explaining the variables of each dataset, the FDadesModel.mat file generated for it and explaining the uncertainty inherent in each. It also includes the results of each dataset for all the options.

Four datasets were used to validate this project and each of them were tested with all 5 options. The datasets used are: Energy in buildings, Sheffield anesthesia, Mackey-Glass time series and Combined cycle power plant. The results are calculated based on two standard error metrics: Root Mean Square Error (RMSE), as given in Equation 10 and Mean Absolute Error (MAE), as given in Equation 11.

$$rmse = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} (y_i - \widehat{y}_i)^2} \tag{10}$$

$$mae = \frac{1}{N} \cdot \sum_{i=1}^{N} |y_i - \widehat{y}_i| \tag{11}$$

where y_i is the true output, $\widehat{y}_i$ is the predicted output, and N is the test sample size. For all the dataset results, the parameter iteration count is set to 5. Regarding this parameter, different tests have been done to determine that a value of 5 maintains a good balance between computation time and prediction accuracy. However, as mentioned in future work, it would be very interesting and convenient to deepen this analysis and make a comparison based on a statistical analysis.

9.1 Energy in Buildings

9.1.1 Dataset Description

Energy in buildings or Energy efficiency dataset is obtained from the UCI repository and was introduced in [43] to develop a statistical machine learning model which identifies the heating load and cooling load of a building. They used energy analysis methods using 12 different buildings and varied the parameters to obtain a dataset containing of

768 different building shapes. This dataset comprises 8 input attributes and 2 output variables (heating and cooling load). But for this project only heating load is considered as output variable, since both outputs have similar characteristics. The variables defined in this dataset are:

1. **Input variables**
   (a) *X1:* Relative compactness of a building.
   (b) *X2:* Surface area of a building.
   (c) *X3:* Wall area of a building.
   (d) *X4:* Roof area of a building.
   (e) *X5:* Overall height of the building.
   (f) *X6:* Orientation of the building.
   (g) *X7:* Glazing area of a building.
   (h) *X8:* Glazing area distribution.
2. **Output variable**
   (a) *Y1:* Heating load of a building.

This dataset containing 768 samples is split into training set, with 691samples, and test set, with 77 samples. The 8 input and the output variables are used by *VisualFIR* to obtain a FIR model that captures as much as possible the behavior of the system, and generates the dataset's FDadesModel.mat file. This identifies the relevant input variables through a search process that generates the *mask* matrix. The depth of the mask is chosen to be 1, as each building will not have an effect on another building heating load, i.e. this application has static relations that does not include time. The mask generated is:

$$\begin{matrix} X1 & X2 & X3 & X4 & X5 & X6 & X7 & X8 & Y1 \\ \begin{pmatrix} 0 & -1 & -2 & -3 & 0 & 0 & -4 & -5 & 1 \end{pmatrix} \end{matrix}$$

From the above matrix we can see that for a depth of 1 (number of rows of the matrix),

variables *X2* , *X3*, *X4*, *X7*,and *X8* are chosen as mask input variables, meaning that these are the antecedents of the rules that have as consequent the output variable *Y1*. The *VClass* variable holds the number of classes (membership functions) for each antecedent and consequent. *X2* , *X3*, *X4*, *X7*,and *X8* are split into 3, 3, 2, 2, 3 classes, respectively and the output *Y1* is split into 3 classes. The landmarks of these classes are stored in the *VFrom* variable. Using these variables and the mask the *VisualFIR* creates the behavior matrix (pattern rules) that is composed of three matrices that contain: the class values *ba*, the membership values *mba*, and the side values *sba*, each having 691 rows and each row representing a sample.

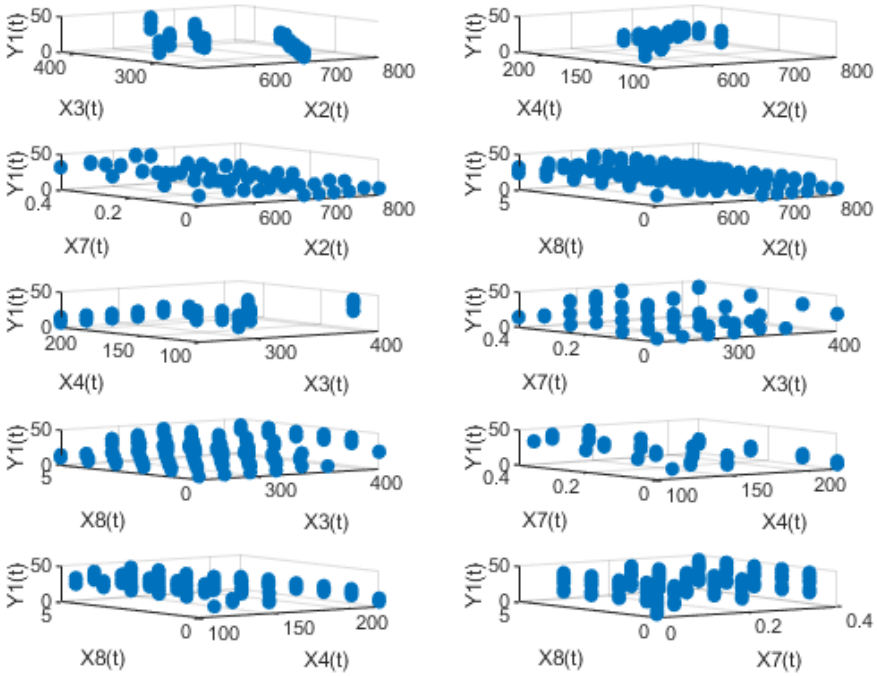

Figure 18: FIR pattern rules for the Energy in buildings dataset.

The uncertainty present in this dataset is identified by plotting all the pattern rules. This is shown in Figure 18. This plot is obtained by passing the mask along the original (crisp) dataset and obtaining the values in the pattern rule base format, i.e. antecedents and then followed by consequent. Then the crisp antecedents are taken in pairs against the crisp consequent. The uncertainty can be observed where for the same value of

antecedents different values of consequent occurs. While plotting such data the thickness of certain regions will be large indicating the inherent uncertainty present in it. In Figure 18, the 2^{nd} row and the 5^{th} row plots where multiple output values are present for the same input signify that there is uncertainty present in them. Some uncertainty is also seen in 4^{th} row 1^{st} column of the plot, where the thickness of the points occur in intervals.

9.1.2 Dataset Results

The five modelling algorithms were applied on the Energy in buildings dataset and the results are tabulated in Table 8. From this table we can see that the options FIR-based Mamdani (option II), FIR-based Sugeno (option III), FIR-Mamdani mixed scheme (option IV) and FIR-Sugeno mixed scheme (option V), have a small rule base when compared to the FIR model. This reduces the model complexity enormously and makes it more understandable. Options II, III, IV, V were tested with all the tuning methods to find that Genetic Algorithm (GA) works best. But this algorithm is an highly exhaustive one and, hence, the reason for high training time. The FIR model outperforms all other models with respect the prediction accuracy but at the cost of high rule-based model complexity. The output predictions of each of the models along with the original output is shown in Figure 19.

Model	No. of fuzzy rules	No. of pattern rules	Best tuning method	Training Time (sec.)	RMSE	MAE
FIR	-	691	-	1	0.54	0.38
FIR-based Mamdani	48	-	GA	80.46	4.68	3.63
FIR-based Sugeno	48	-	GA	216.58	4.43	3.39
FIR-Mamdani mixed	48	69	GA	79.59	3.7	2.74
FIR-Sugeno mixed	48	69	GA	215.48	3.32	2.35

Table 8: Results for Energy in buildings dataset.

As expected the lower prediction error is obtain when the FIR model is used. The difference in the errors are quite relevant, as it is also the training time. Then, both mixed prediction schemes (options IV and V) have lower errors than its corresponding FIR-

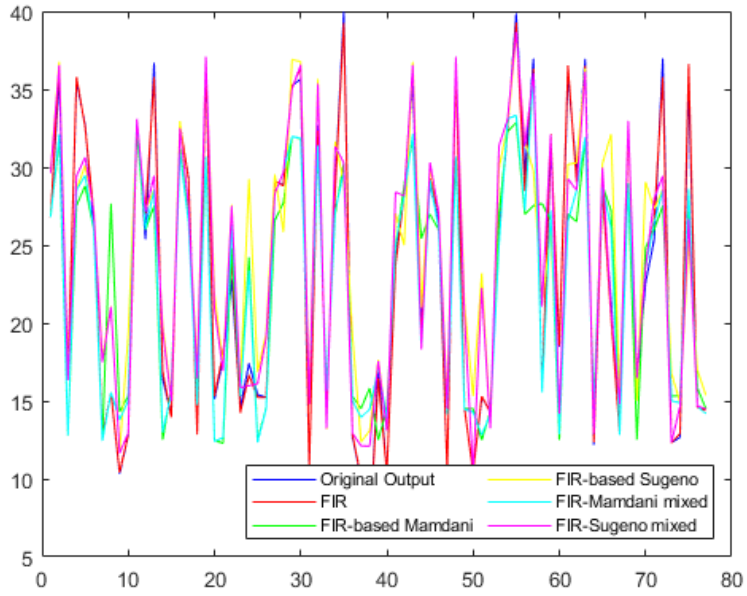

Figure 19: Predictions of all models on Energy in buildings dataset.

based models. The inclusion of a subset of 69 pattern rules that contain an important level of uncertainty associated to the data allow to reduce the RMSE and MAE error up to one point. However, the mixed models are conformed of 48 fuzzy rules and 69 pattern rules, a number that is far away from the FIR model complexity, but is less interpretable than the set of 48 fuzzy rules that the FIR-based Mamdani and Sugeno models have. Therefore, the model that should be uses will depend on the modelling goal of the user/modeler.

The effect of p value on the RMSE values of this dataset is shown in Figure 20. The curve eventually tends to lower errors as the p value tends to 1. As expected, if the number of pattern rules that we keep for using in the prediction process becomes higher, the prediction error becomes lower, but the complexity of the model increases. We can see some plateau regions in the figure, after 0.2, 0.5 and near 0.7 p values. This is due to the fact that the level of uncertainty present in the rules between these two p values is almost the same.

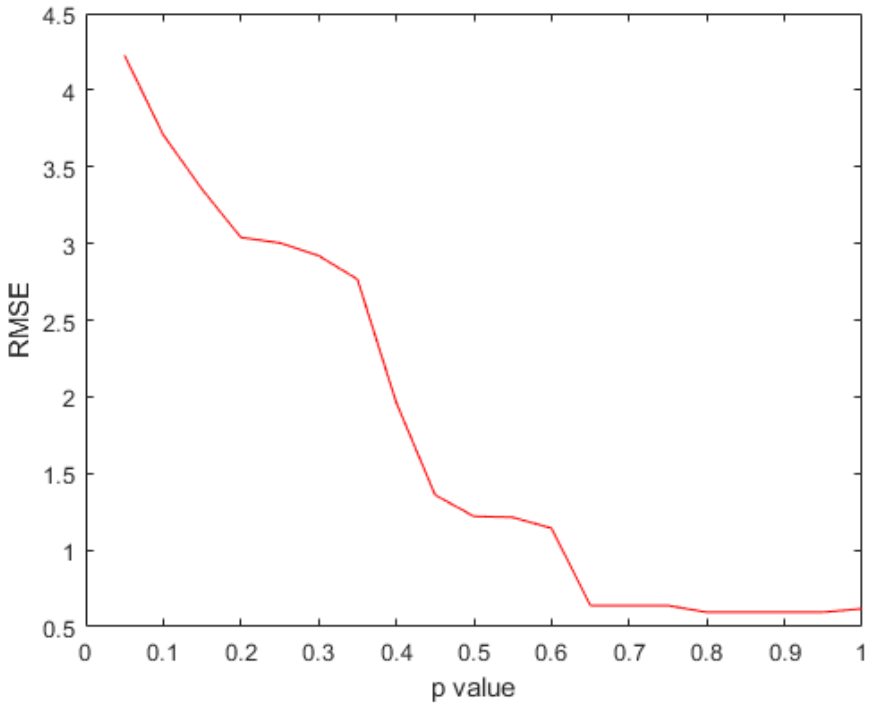

Figure 20: Effect of p value on Energy in buildings dataset.

9.2 Sheffield Anesthesia

9.2.1 Dataset Description

The Sheffield dataset is a collection of medical data with aim of identifying the dosage of anesthesia to be administered to the patient. It was introduced in [44] and was collected directly from an hospital of Sheffield (England). This clinical data is collected by consulting various anesthetists and by also monitoring a controller module called RESAC, which observes the patients' medical status. This dataset consists of 3 input variables and one out variable and they had used a FIR modelling process to develop a FIRAD (FIR Anesthetic Dosage) which indicates the dosage of anesthesia to be administered. The variables defined in this dataset are:

1. **Input variables**
   (a) *HR:* Heart rate of the patient.

(b) *RR:* Respiration rate of the patient.

(c) *SAP:* Systolic arterial pressure of the patient.

2. **Output variable**

(a) *ISO:* Dosage of anesthetic adminsitered.

The dataset comprises 163 samples of clinical data, each sampled at rate of 1 min. These samples are split into training and testing datasets with 135 and 28 samples, respectively. The dataset is passed to *VisualFIR* to generate a mask for which the depth is chosen as 2. This means that the current dosage of anesthesia to be administered to de patient depends on the current clinical data and the previous (one minute in the past) clinical data. The mask generated is:

$$\begin{matrix} HR & RR & SAP & ISO \end{matrix}$$
$$\begin{pmatrix} 0 & 0 & 0 & -1 \\ -2 & 0 & -3 & 1 \end{pmatrix}$$

The mask generated shows that the current dosage to be administered is temporally dependent on the previous dosage administered and corresponds to the first input variable. The *HR* and *SAP* are the second and third input variables (antecedents), with the *RR* having no casual relation with the current *ISO* (consequent). Each of the input variable is fuzzified into 2 classes while the output variable is fuzzified into 3 classes. This information is stored in the *VClass* variable and the *VFrom* stores the landmarks for these classes. The behavior class value matrix (*ba*), membership value matrix (*mba*), and side value matrix (*sba*) are obtained and contain only 134 data points. This is because since the depth of the mask (n) is greater than 1 the initial $n-1$ rows cannot have a pattern rule generated and can only be used to create a pattern rule for the n^{th} sample.

The uncertainty present in the Sheffield dataset is shown in Figure 21, where all the pattern rules are ploted. The mask generated from *VisualFIR* is passed through the crisp values to obtain the crisp antecedents (*ISO*, *HR*, *SAP*) and the consequent (*ISO*) which is used to generate these plots. The thickness of the data points can be seen in all three plots but is of smaller width. This means that there is less uncertainty present in the dataset.

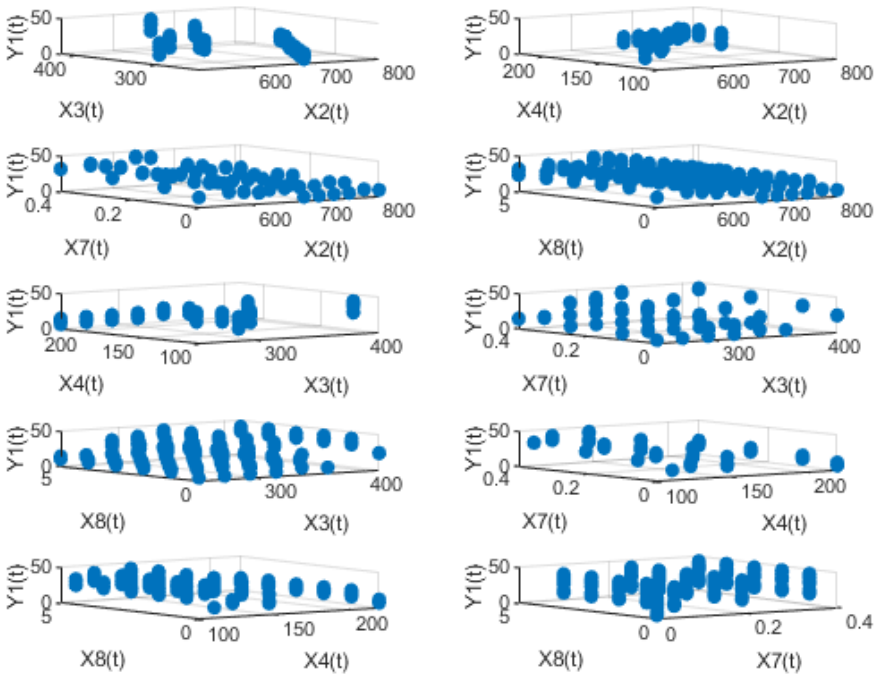

Figure 21: FIR pattern rules for the Sheffield dataset.

9.2.2 Dataset Results

On applying the different options on Sheffield dataset we generate the results given in Table 9. This table shows that options II and IV as well as options III and IV have the same best tuning method, i.e. Pattern Search (PS) and Genetic Algorithm (GA), respectively. This is because the mixed scheme models (option 4 and 5) are just an extension of the FIR-based models (option II and III) and, hence, the same tuning method works better for both.

For this dataset, as expected, the lower prediction errors are obtained by the FIR model, being the prediction really very good as can be seen in Figure 22. However, the model uses 134 pattern rules, being too complex to be useful as a decision making tool. The Rule-based Mamdani and Sugeno models, with only 9 rules, are able to capture quite well the behavior of the system since the RMSE and MAE errors obtained are also low. In fact, for this dataset, the FIR-Mamdani and FIR-Sugeno mixed schemes obtain similar results than FIR-based models, but the complexity of these models are increased

with 13 pattern rules.

Model	No. of fuzzy rules	No. of pattern rules	Best tuning method	Training Time (sec.)	RMSE	MAE
FIR	-	134	-	1	0.013	0.002
FIR-based Mamdani	9	-	PS	1.63	0.18	0.15
FIR-based Sugeno	9	-	GS	19.2	0.22	0.17
FIR-Mamdani mixed	9	13	PS	1.74	0.16	0.14
FIR-Sugeno mixed	9	13	GA	19.14	0.18	0.14

Table 9: Results for Sheffield anesthesia dataset.

The effect of p value for this dataset is shown in Figure 23. We can see some plateau regions. The reason for this plateau is the same as that given for Energy in buildings dataset. A sudden dip in the error is seen between 0.5 and 0.6 p values. This shows that rules with higher uncertainty are predominant between those values.

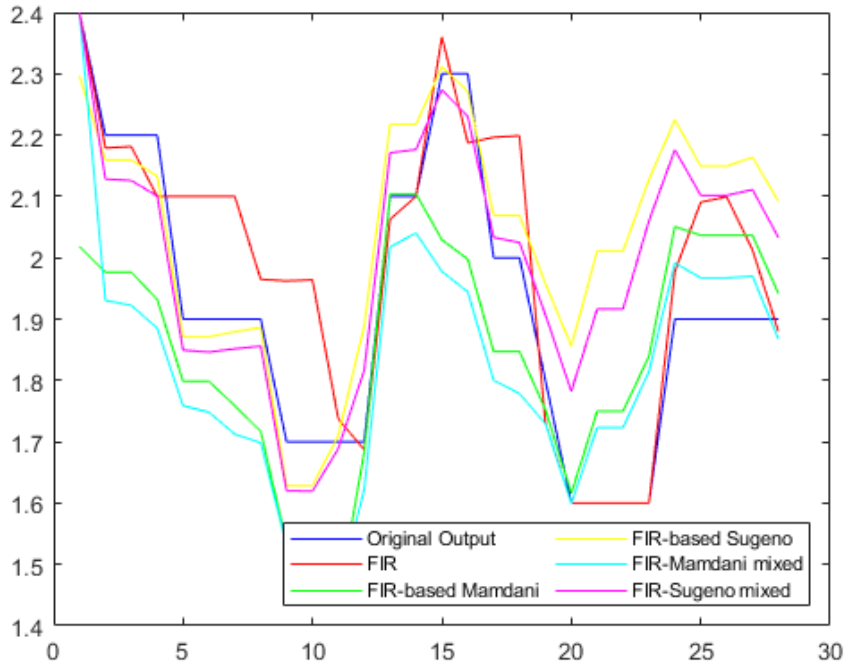

Figure 22: Predictions of all models on Sheffield anesthesia dataset.

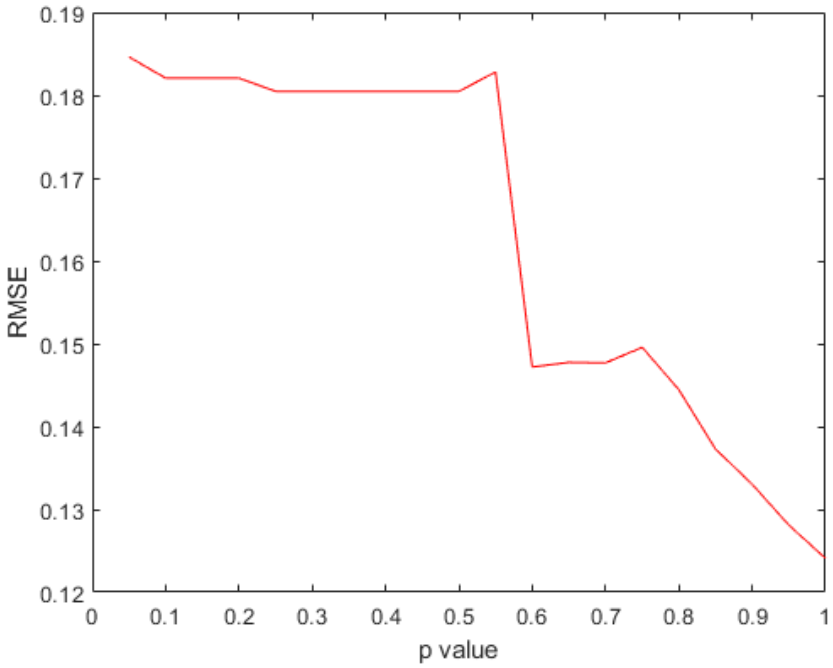

Figure 23: Effect of p value on Sheffield anesthesia dataset.

9.3 Mackey-Glass

9.3.1 Dataset Description

Mackey-Glass dataset is a time series dataset based on the Mackey-Glass equation (Eq. 12). A Mackey-Glass time series generator is available in MATLAB which is created by Marco Cococcioni [45]. A time series dataset is one where one data point is temporally dependent on previous ones, i.e. only one variable is available. Hence this is very suitable for evaluating this project. In [45], the Mackey-Glass series is generated by using the 4^{th} order Runge-Kutta method. The Mackey-Glass's nonlinear time delay equation is defined as:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}y} = \frac{ax(t-\tau)}{1+x(t-\tau)^{10}} - bx(t) \tag{12}$$

where a and b are constant parameters to be specified, τ is the delay constant and $x(0)$ is the first value at $t = 0$. This generates a single variable series dataset. The variable

defined in the dataset is:

1. **Input and Output variable**

   (a) *Y:* This is the time series data generated by solving Equation 12 using 4^{th} order Runge-Kutta method where each sample is dependent on the previous ones.

The generator is used to create a dataset comprising of 2500 samples, which is used as training set. Another set of 500 samples is used as test data. Since this dataset has only one variable which is the output itself, the only relations used here are temporal relations. To deal with a highly temporally dependent data a mask with bigger depth is necessary. So a mask with depth of 20 is chosen and the mask generated from the search process of *VisualFIR* is:

$$
\begin{matrix} Y \\ \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -2 \\ 1 \end{pmatrix} \end{matrix}
$$

This mask shows that the current data point (consequent) is only dependent on the immediate previous and the 19^{th} previous data points (antecedents). The single variable Y is divided into 9 classes and stored in the *VClass* variable. The *VFrom* holds the landmarks for these 9 classes. Using the mask and these variables the class value matrix (*ba*), membership value matrix (*mba*), and side value matrix (*sba*) are created, each having 2481 samples. The first 19 samples are not used as the depth of the mask is 20.

Since the output of the mask only depends on two inputs, i.e. $y(t-1)$ and $y(t-19)$, the uncertainty diagram in Figure 24 has only one plot. As the number of pattern rules is large the thickness of the data points is not as easily distinguishable as Figure 18 and Figure 21. But for values of Y between 0.5 and 1 in the input axes (x and y axis), we

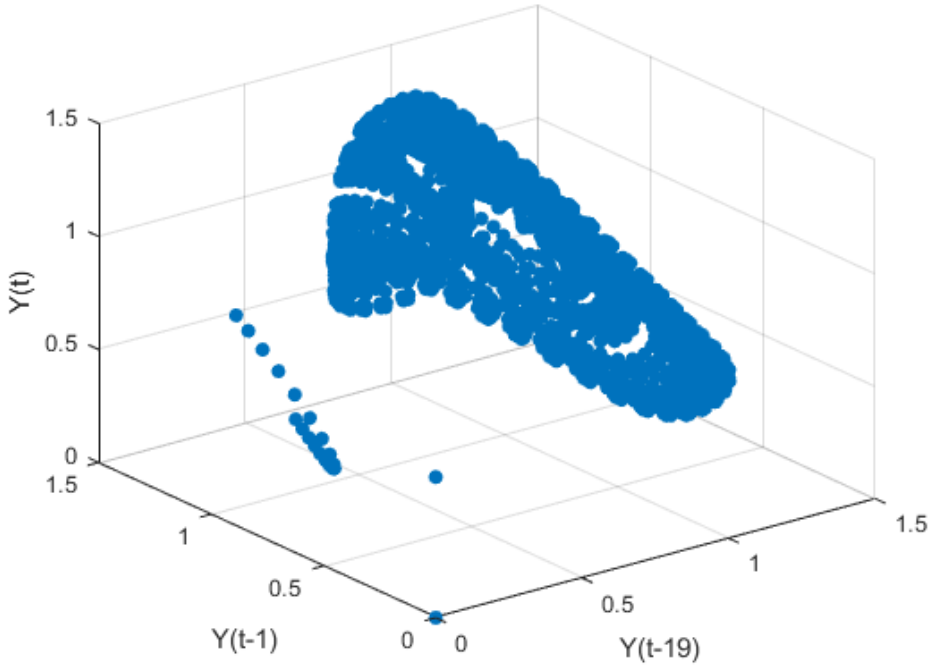

Figure 24: FIR pattern rules for the Mackey-Glass dataset.

can see some thickness along the points indicating the presence of uncertainty.

9.3.2 Dataset Results

Table 10 shows the results obtained from all the models on Mackey-Glass dataset. From this table we can see that the best tuning methods for all the models follows the same pattern as that of Sheffield anaesthesia, as both the Mamdani models perform well with Simulated Annealing (SA) approach and both Sugeno models with Pattern Search (PS) approach. Both the Mamdani models outperform all the other models significantly, except FIR, with much lesser training time even with iteration count as 5. The FIR model handles the data very well. The predictions of all five model options is shown in Figure 25.

Model	No. of fuzzy rules	No. of pattern rules	Best tuning method	Training Time (sec.)	RMSE	MAE
FIR	-	2481	-	76	0.00008	0.000003
FIR-based Mamdani	41	-	SA	1.33	0.03	0.02
FIR-based Sugeno	41	-	PS	33.67	0.24	0.18
FIR-Mamdani mixed	41	248	SA	1.57	0.02	0.01
FIR-Sugeno mixed	41	248	PS	34.08	0.09	0.05

Table 10: Results for Mackey-Glass dataset.

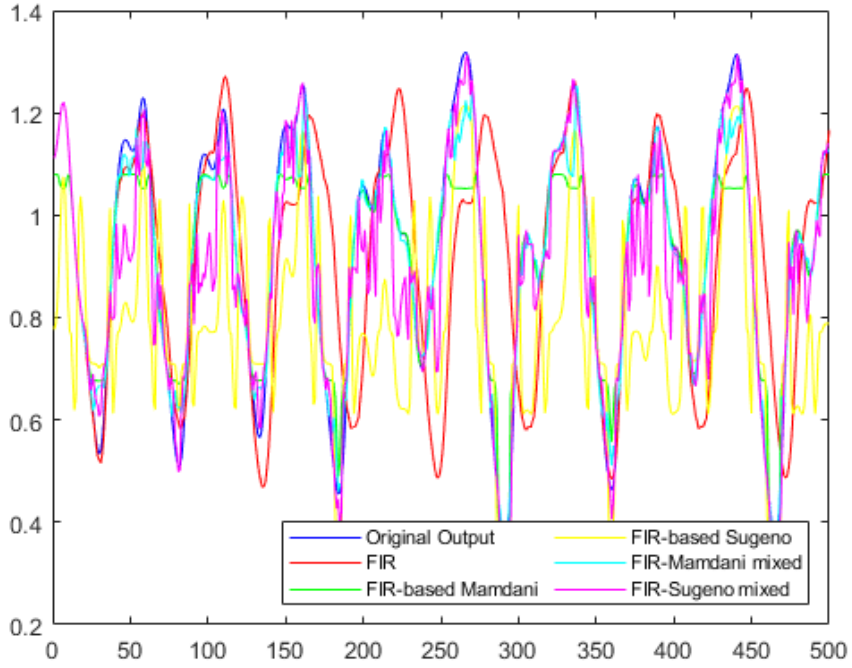

Figure 25: Predictions of all models on Mackey-Glass dataset.

Notice that all the modelling options get low prediction errors, except FIR-based Sugeno approach. FIR obtains almost a perfect match with the real data, but FIR model needs 2481 pattern rules to perform this prediction, being not useful when needed as decision making tool or when explainability is required. The FIR-based Mamdani does a good job, since the error is low, it is trained very quickly and the model complexity level is low (only 41 fuzzy rules). So, it is a good option if the goal is to explain the reasoning process or the model is needed to take decisions. Notice, however, that the piks of the

signal are not well predicted when the FIR-based Mamdani is used (see green signal in Figure 25).

The effect of p value for this dataset is shown in Figure 26. It can be seen that the RMSE is reduced gradually, without any plateau regions, as p slowly moves towards 1.

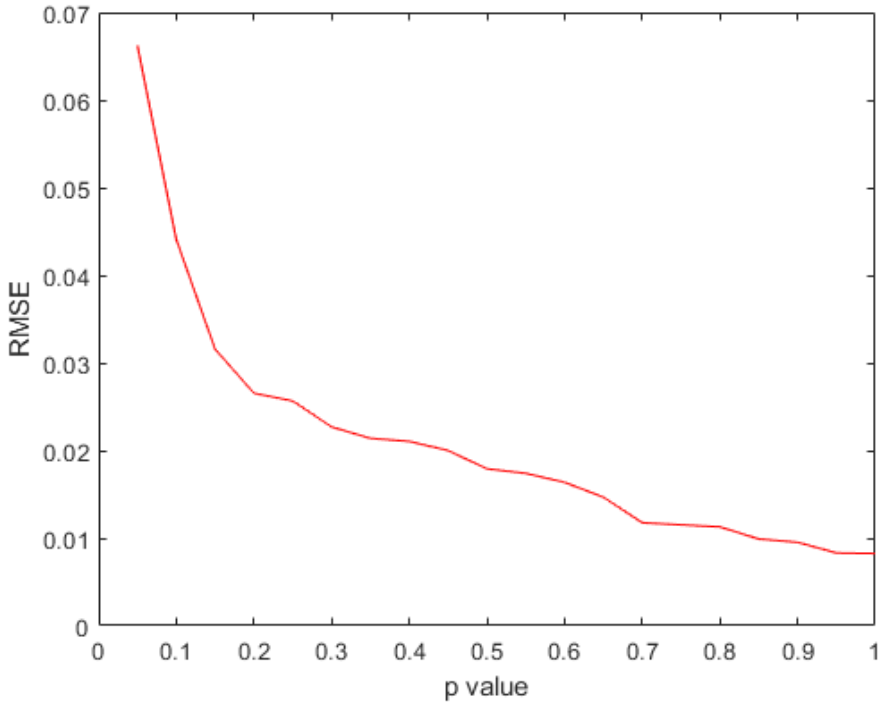

Figure 26: Effect of p value on Mackey-Glass dataset.

9.4 Combined Cycle Power Plant

9.4.1 Dataset Description

The Combined Cycle Power Plant dataset is one where a power plant was set to work with full load and the data was collected over a span of 6 years in hourly averages. This dataset was introduced in [46] where this was used to compare different regression methods on a real-time application. Since this dataset was directly obtained from a working power plant, it can be used to validate how this project works in real-time. This dataset has 4 input variables and one output variable and are defined as:

1. **Input variables**
   (a) *AT:* Temperature of the turbines running in the power plant. Ranges from 1.81°C and 37.11°C.
   (b) *AP:* Ambient pressure present in the power plant. Ranges from 992.89-1033.30 milibar.
   (c) *RH:* Relative humidity of the plant. Ranges from 25.56% to 100.16%.
   (d) *V:* Exhaust vaccum. Ranges from 25.36 to 81.56 cm Hg.
2. **Output Variable**
   (a) *EP:* Net hourly electrical energy output. Ranges from 420.26 to 495.76 MW.

This dataset consists of 9568 samples and is split into training and testing datasets, having 8000 and 1568 samples, respectively. The *VisualFIR* is used to generate the mask for this dataset by using a depth of 1. This depth is chosen because the energy output to be predicted is dependent on the power plant variables present at that hour. The mask obtained is:

$$\begin{matrix} AT & AP & RH & V & EP \end{matrix}$$
$$\begin{pmatrix} -1 & -2 & -3 & -4 & 1 \end{pmatrix}$$

From this mask, we can see that the output variable *EP* (consequent) is dependent on all the input variables (antecedant). All the variables, both input and output, are divided into 3 classes which is specified in *VClass*. The landmarks for these classes are stored in *VFrom*. These variables are used in *VisualFIR* to create the class value matrix (*ba*), the membership value matrix (*mba*), and the side value matrix (*sba*), each having a sample size of 8000 as depth is 1.

The sample size of this dataset is very large. So the regions with thickness present cannot be identified easily. But a close look at the 1^{st} row 1^{st} column and the 2^{nd} row plots show that there is uncertainty present in them. A small level of uncertainty can also be observed in the 3^{rd} row 1^{st} column plot where some thickness is observed in the

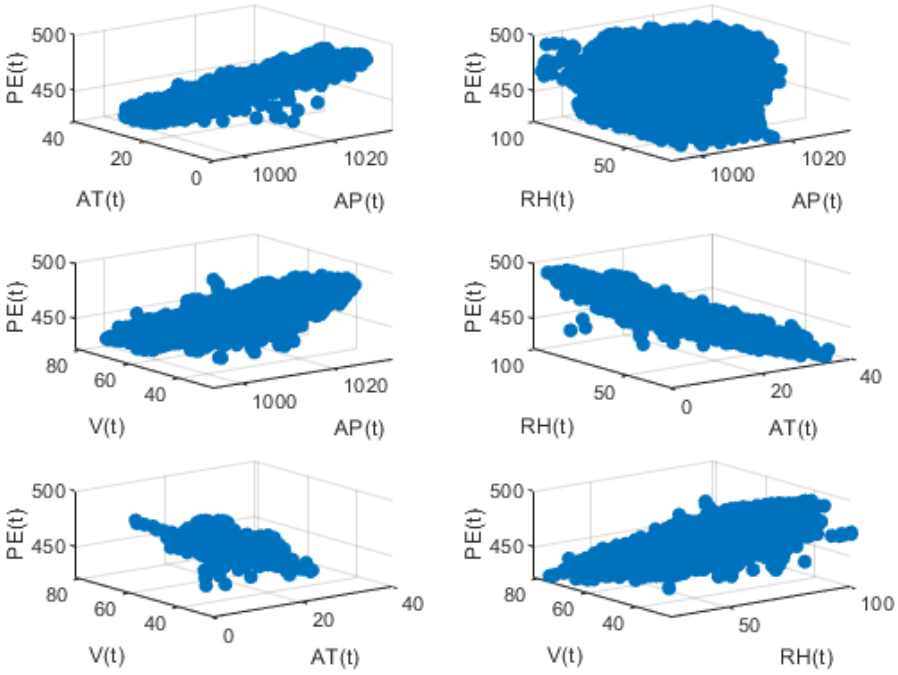

Figure 27: FIR pattern rules for the Combined Cycle Power Plant dataset.

middle.

9.4.2 Dataset Results

From Table 11 we can see that the FIR model produces great prediction results compared to other models, however it uses a rule base having 8000 rules. The FIR-Sugeno mixed scheme model uses 869 rules in total and with the help of Particle Swarm (PSW) tuning method is able to provide decent results with much better rule complexity and interpretability. However, with 869 rules is still a complex model when the objective is the interpretability. If this is the goal, FIR-based Sugeno is the best option for this dataset, since the number of rules decrease up to 69. The predictions obtained using the five approaches on Combined Cycle Power Plant dataset are shown in Figure 28.

Model textbf	No. of fuzzy rules	No. of pattern rules	Best tuning method	Training) Time (sec.)	RMSE	MAE
FIR	-	8000	-	14	3.97	2.75
FIR-based Mamdani	69	-	PS	37.46	17.27	15.07
FIR-based Sugeno	69	-	PSW	244.45	13.2	10.93
FIR-Mamdani mixed	69	800	PS	37.6	10.57	8.51
FIR-Sugeno mixed	69	800	PSW	245.37	8.42	6.51

Table 11: Results for Combined Cycle Power Plant dataset.

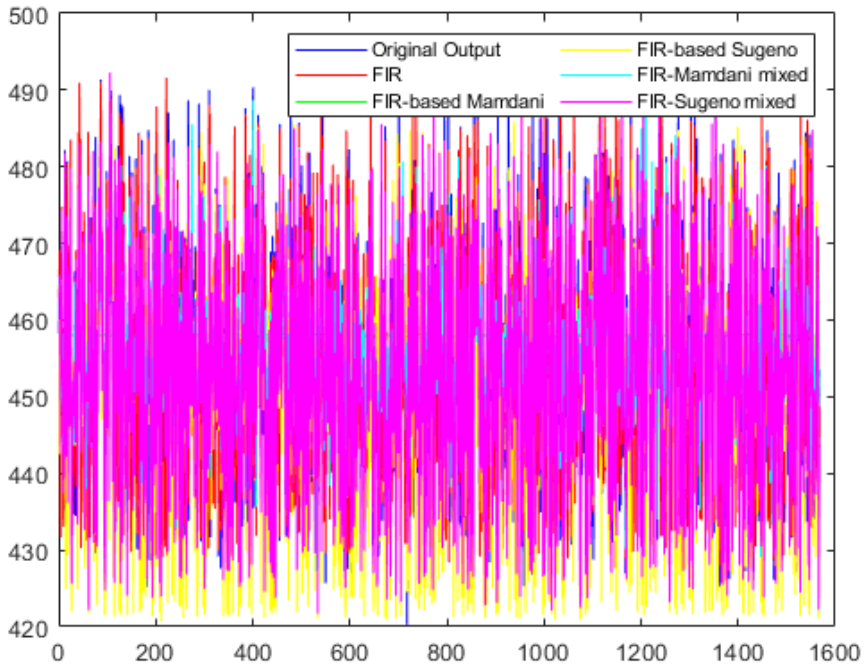

Figure 28: Predictions of all models on Combined Cycle Power Plant dataset.

The effect of p value for this dataset is shown in Figure 29. It can be seen that RMSE value decreases smoothly without any plateau regions, indicating that the uncertainty is spread evenly across the rules.

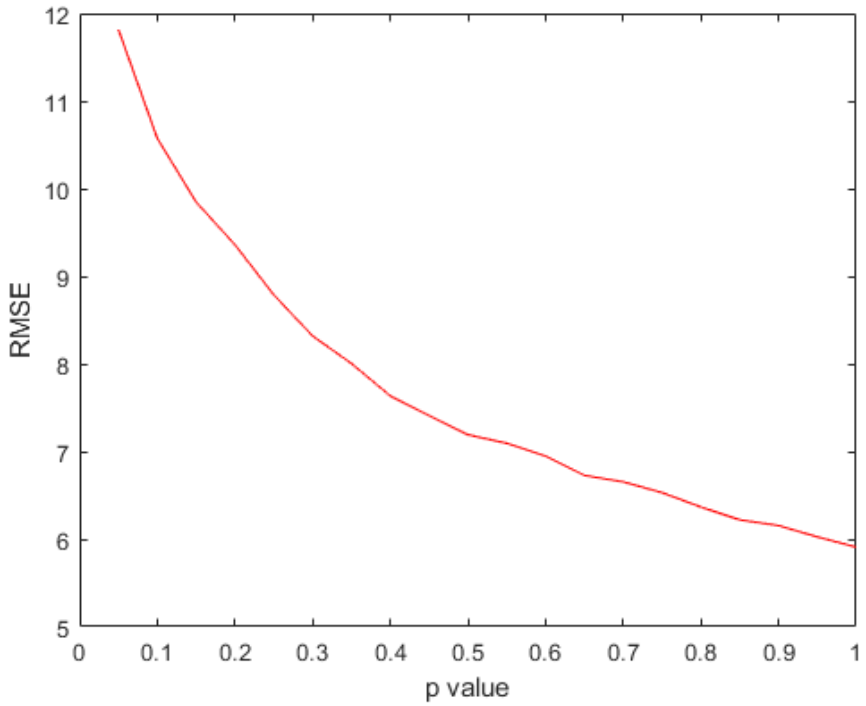

Figure 29: Effect of p value on Combined Cycle Power Plant dataset.

10 Result Comparison and Discussion

This section discusses the results produced by applying the different models discussed in Section 7 to the different datasets discussed in Section 9. All the options of this project are tested against all datasets in common ground and similar models are tested by varying the input parameters. The results are calculated using RMSE, as given in Equation 10 and MAE, as given in Equation 11.

10.1 Comparing the Modelling Options

This project as explained in the previous sections has five options: FIR (I), FIR-based Mamdani (II), FIR-based Sugeno (III), FIR-Mamdani mixed scheme (IV) and FIR-Sugeno mixed scheme (V). These options are evaluated with the datasets explained in Section 9. To compare all the options, a common ground is established so that the results can be compared effectively. So to achieve this the input parameters are defined as a constant for all the options. Option I has no specific input parameters except the FDadesModel.mat file. Options II and III depend on the tuning method and the number of tuning iterations, while Option IV and V depend on the p value given apart from the tuning method and number of tuning iterations. For options II, III, IV, V the tuning method is set as pattern search and the iteration count as 5. In addition, for options IV and V the p is set as 0.1, that is 10% of the FIR rules are taken into the mixed model to handle the uncertainty. The results obtained are shown in Table 12. Each result is shown in the format as: RMSE value/ MAE value.

Dataset/Option	I	II	III	IV	V
Energy Buildings	0.54/0.38	4.72/3.64	8.55/7.37	3.33/2.46	6.86/5.19
Sheffield	0.013/0.002	0.18/0.15	0.29/0.24	0.16/0.14	0.28/0.24
Mackey-Glass	0.00008/0.000003	0.09/0.05	0.24/0.18	0.05/0.03	0.09/0.05
Combined Cycle Power Plant	3.97/2.75	17.27/15.07	16.51/14.25	10.57/8.51	10.21/8.07

Table 12: Comparison of all options and datasets.

From Table 12, we can see some general trends. One is that FIR performs better than other options in all datasets. The reason that FIR performs better is because of its bigger

pattern rule base (that captures all system's uncertainty). The other general trend that can be observed is that the results of options IV and V lie in between that of option I and options II and III. This is due to the fact that adding the uncertain rule subset from the FIR rule base decreases the error of options IV and V but still maintaining the interpretability, up to some extend.

In Energy in buildings dataset, option II and option IV is better than options III and V. This shows that the Mamdani model works better for this dataset than the Sugeno model. The results of Sheffield anesthesia dataset shows that the Mamdani models (options II and IV) obtain much better results than the Sugeno models (options III and V). The same behavior can be seen in Mackey-Glass dataset where the Mamdani models outperform the Sugeno options. In the Combined Cycle Power Plant dataset, Sugeno approaches (options III and V) perform better than Mamdani approaches (options II and IV) but are nowhere close to option I. We can see a significant jump in prediction results from options II and III to options IV and V which is due to the addition of just 10 per cent of pattern rules.

10.2 Comparing FIR-based Models

The FIR-based Mamdani and Sugeno depend mostly on the tuning method and the iteration count. Hence to compare these two options on the datasets, we need to tune these input parameters. But to compare based on iteration count requires high compute capability to increase the count consecutively. So the two options are compared using different tuning methods: Genetic Algorithm (GA), Particle Swarm (PSW), Pattern Search (PTS), Simulated Annealing (SA). They are compared on Energy in buildings and Mackey-Glass datasets.

Model/Tuning method	**GA**	**PSW**	**PTS**	**SA**
FIR- based Mamdani	4.68/3.63	4.69/3.66	4.72/3.64	4.94/3.90
FIR- based Sugeno	4.43/3.39	6.20/4.82	8.55/7.37	8.88/7.48

Table 13: FIR-based Mamdani and Sugeno comparison on Energy in buildings dataset.

Table 13 shows the results from different tuning methods for FIR-based Mamdani and

Sugeno in Energy in buildings dataset. From this table we can see that the Mamdani model performs almost the same for all the tuning methods. The Sugeno model works comparatively well while using Genetic Algorithm rather than other tuning methods. The Sugeno model even outperforms Mamdani model slightly when using the Genetic Algorithm method hence making this tuning method the apt one for this dataset.

Model/Tuning method	**GA**	**PSW**	**PTS**	**SA**
FIR- based Mamdani	0.34/0.29	0.27/0.21	0.09/0.05	0.03/0.02
FIR- based Sugeno	0.33/0.26	0.41/0.32	0.24/0.18	0.35/0.28

Table 14: FIR-based Mamdani and Sugeno comparison on Mackey-Glass dataset.

The results from comparing the different tuning algorithms in Mackey-Glass dataset are shown in Table 14. We can see that the Mamdani model performs very well while using Simulated Annealing algorithm. The Sugeno model performs poorly while using the Particle Swarm algorithm. But again on using Pattern Search provides decent results.

11 Conclusion

The main idea of this project was to develop modelling hybrid methodologies that are able to handle the uncertainty of real live in such a way that the interpretability of the resulting models is a fact. The classical FIR methodology, even though it is a very efficient approach to obtain accurate models, can't easily get interpretable models useful for decision making. Therefore, from the *VisualFIR* module, 5 models were derived. These were combined into a single framework comprising of these models as 5 options: FIR, FIR-based Mamdani, FIR-based Sugeno, FIR-Mamdani mixed scheme and FIR-Sugeno mixed scheme. The large pattern rule base of the FIR model is processed using specific algorithms based on classical fuzzy model (Mamdani or Sugeno). The obtained fuzzy rule base is concise and more interpretable than the original one. This makes the resulting models more effective to use for complex real-time problems which require heavy computations. The uncertainty in data is also captured by identifying the rules which cause them and used in the models of the last two options.

From the results of this project in different datasets we can conclude that FIR generally performs well in all datasets. But this performance comes with the cost of less interpretability of the model, being less useful to draw future intuitions on the data. The FIR-based Mamdani and FIR-based Sugeno models provide almost similar results in the datasets used. The mixed models outperform the FIR-based models as the small set of uncertain pattern rules boosts the performance. It can also be postulated that the tuning methods vary according to the model used and the dataset on which it is applied. So the correct tuning method to be used can be determined by trial and error. The p value is also seen to change the results drastically but comes with the cost of losing interpretability of the system and, hence, is a trade-off.

With respect to my conclusions, this project has been a very interesting opportunity for me to learn about a branch of artificial intelligence which I did not have any knowledge. After learning about the foundations of Fuzzy Logic it made me realise that it could be applied to a wide variety of applications and could help in many decision-making processes. I also understood the drawbacks of the classical models which made me arrive at solutions that could alleviate these drawbacks. This project has also made me learn about new modules and functions in MATLAB which help in these decision

making processes. The use of certain realtime datasets and the results produced in this project left me enthralled and has increased my passion in this domain.

12 Future Work

Many ideas were written at the beginning of this project. But due to the limited time constraint, some were removed and could now be given an opportunity to be explored. One of them is to use pure classical models (pure Mamdani and Sugeno) without hybridizing with the FIR approach and allow these models to learn input and output membership functions and fuzzy rules by themselves. We postulate that the computation time of the pure Mamdani and Sugeno approaches will be very high and that the precision results will probably be worse than the FIR based Mamdani and Sugeno approaches, but we would like to test whether this hypothesis is correct. Yet another idea was to incorporate cross-validation and parameter search methods into this framework. These ideas could be addressed and studied in depth in the future. We would also have liked to do a more in-depth analysis of the comparison between the five modeling approaches developed in this project, performing a statistical analysis of the different results of the modeling parameters.

References

[1] L. A. Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

[2] Jui-Chuan Cheng, Chao-Yuan Chiu, and Te-Jen Su. Training and evaluation of human cardiorespiratory endurance based on a fuzzy algorithm. *Int. J. Environ. Res. Public Health*, 16(13):2390–665, 2019.

[3] George J. Klir and B. Yuan. *Fuzzy sets and fuzzy logic*. Prentice Hall, 1995.

[4] François Cellier, Àngela Nebot, and Alvaro de Alvornoz. Combined qualitative quantitative simulation models of continuous-time processes using fuzzy inductive reasoning techniques. *International Journal Of General System*, 24:95–116, 02 1996.

[5] Lotfi A Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[6] D. Singer and P. G. Singer. System identification based on linguistic variables. 1992.

[7] A.M. Luciano and M Savastano. Fuzzy identification of systems with unsupervised learning. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 27:138–41, 02 1997.

[8] Ken Nozaki, Hisao Ishibuchi, and Hideo Tanaka. A simple but powerful heuristic method for generating fuzzy rules from numerical data. *Fuzzy sets and systems*, 86(3):251–270, 1997.

[9] J-SR Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, 23(3):665–685, 1993.

[10] Hisao Ishibuchi, Ken Nozaki, Naohisa Yamamoto, and Hideo Tanaka. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on fuzzy systems*, 3(3):260–270, 1995.

[11] Shigeo Abe and Ming-Shong Lan. A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *IEEE transactions on fuzzy systems*, 3(1):18–28, 1995.

[12] Abdollah Homaifar and Ed McCormick. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE transactions on fuzzy systems*, 3(2):129–139, 1995.

[13] LiMin Fu. Incremental knowledge acquisition in supervised learning networks. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 26(6):801–809, 1996.

[14] Hisao Ishibuchi, Ryosuke Fujioka, and Hideo Tanaka. Neural networks that learn from fuzzy if-then rules. *IEEE Transactions on Fuzzy Systems*, 1(2):85–97, 1993.

[15] Ching-Chang Wong and Nine-Shen Lin. Rule extraction for fuzzy modeling. *Fuzzy sets and systems*, 88(1):23–30, 1997.

[16] Shan-Ben Chen, L Wu, and QL Wang. Self-learning fuzzy neural networks for control of uncertain systems with time delays. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 27(1):142–148, 1997.

[17] S Narendra Kumpati, Parthasarathy Kannan, et al. Identification and control of dynamical systems using neural networks. *IEEE Transactions on neural networks*, 1(1):4–27, 1990.

[18] Kumpati S Narendra and Kannan Parthasarathy. Neural networks and dynamical systems. *International Journal of Approximate Reasoning*, 6(2):109–131, 1992.

[19] John Theocharis and George Vachtsevanos. Recursive learning algorithms for training fuzzy recurrent models. *International journal of intelligent systems*, 11(12):1059–1098, 1996.

[20] Hoon Kang. Stability and control of fuzzy dynamic systems via cell-state transitions in fuzzy hypercubes. *IEEE Transactions on Fuzzy Systems*, 1(4):267–279, 1993.

[21] Ahmad Lotfi, Hans Christian Andersen, and Ah Chung Tsoi. Matrix formulation of fuzzy rule-based systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(2):332–340, 1996.

[22] M Ouassir and C Melin. Causal graphs and rule generation: application to fault diagnosis of dynamic processes. *IFAC Proceedings Volumes*, 30(18):1087–1092, 1997.

[23] Patrick K Simpson. Fuzzy min—max neural networks—part 1: Classification. *IEEE Trans. on Neural Networks*, 3(5):776–786, 1992.

[24] Ruck Thawonmas and Shigeo Abe. A novel approach to feature selection based on analysis of class regions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 27(2):196–207, 1997.

[25] Yi Lu and Tie Qi Chen. Fast rule generation and membership function optimization for a fuzzy diagnosis system. In *Proceedings of the 10th international conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 349–356, 1997.

[26] Ken Nozaki, Hisao Ishibuchi, and Hideo Tanaka. Adaptive fuzzy rule-based classification systems. *IEEE Transactions on fuzzy Systems*, 4(3):238–250, 1996.

[27] Riccardo Rovatti and Roberto Guerrieri. Fuzzy sets of rules for system identification. *IEEE Transactions on Fuzzy Systems*, 4(2):89–102, 1996.

[28] Michèle Sebag, WJ Maas, and Marc Schoenauer. Inductive learning of membership functions and fuzzy rules. 1993.

[29] Tomohiro Takagi and Michio Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE transactions on systems, man, and cybernetics*, (1):116–132, 1985.

[30] M. Sugeno and T. Yasukawa. A fuzzy-logic-based approach to qualitative modeling. *IEEE Transactions on Fuzzy Systems*, 1(1):7–, 1993.

[31] Hiroyoshi Nomura, Isao Hayashi, and Noboru Wakami. A learning method of fuzzy inference rules by descent method. In *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*, pages 203–210. IEEE, 1992.

[32] Agile- scrum process. `https://www.cprime.com/resources/what-is-agile-what-is-scrum/`.

[33] Average salary in spain. https://www.averagesalarysurvey.com/spain.

[34] George J. Klir. *Architecture of Systems Problem Solving*. Springer US, 1985.

[35] Àngela Nebot and Francisco Mugica. Fuzzy inductive reasoning: A consolidated approach to data-driven construction of complex dynamical systems. *Int. J. Gen. Syst.*, 41:645–665, 2012.

[36] Ebrahim H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. 1993.

[37] Timothy J Ross. *Fuzzy logic with engineering applications*. John Wiley & Sons, 2005.

[38] Sergio Jurado, Àngela Nebot, Francisco Mugica, and Narcís Avellana. Hybrid methodologies for electricity load forecasting: Entropy-based feature selection with machine learning and soft computing techniques. *Energy*, 86, 05 2015.

[39] Pilar Gómez, Àngela Nebot, Sabrine Ribeiro, Rene Alquezar, and Franz Wotawa. Local maximum ozone concentration prediction using soft computing methodologies. *Systems Analysis Modelling Simulation*, 43:1011–1031, 08 2003.

[40] Félix Castro Espinoza and Àngela Nebot. On the extraction of decision support rules from fuzzy predictive models. *Appl. Soft Comput.*, 11:3463–3475, 06 2011.

[41] Francisco Mugica and Àngela Nebot. Reasoning under uncertainty with fir methodology. *Int. J. Approx. Reasoning*, 41:287–313, 04 2006.

[42] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[43] Athanasios Tsanas and Angeliki Xifara. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. 2012.

[44] Àngela Nebot, François E. Cellier, and Derek A. Linkens. Synthesis of an anaesthetic agent administration system using fuzzy inductive reasoning. *Artificial intelligence in medicine*, 8 2:147–66, 1996.

[45] Mackey-glass time series generator. https://in.mathworks.com/matlabcentral/fileexchange/24390-mackey-glass-time-series-generator.

[46] Pınar Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014.