REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggesstions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA, 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any oenalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
10-03-2016	Ph.D. Dissertation	-

4. TITLE AND SUBTITLE
MULTIPLE INSTANCE FUZZY INFERENCE

5a. CONTRACT NUMBER
W911NF-13-1-0066

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS
Amine Ben Khalifa

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
University of Louisville
2301 S. Third Street
Jouett Hall
Louisville, KY 40208 -1838

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
63184-CS.12

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
A novel fuzzy learning framework that employs fuzzy inference to solve the problem of multiple instance learning (MIL) is presented. The framework introduces a new class of fuzzy inference systems called Multiple Instance Fuzzy Inference Systems (MI-FIS). Fuzzy inference is a powerful modeling framework that can handle computing with knowledge uncertainty and measurement imprecision e♂ectively. Fuzzy Inference performs a nonlinear mapping from an input space to an output space by deriving conclusions from a set

15. SUBJECT TERMS
Fuzzy Inference, Multiple Instance, Mamdani, Sugeno, ANFIS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Hichem Frigui
UU	UU	UU	UU		19b. TELEPHONE NUMBER 502-852-2009

Report Title

MULTIPLE INSTANCE FUZZY INFERENCE

ABSTRACT

A novel fuzzy learning framework that employs fuzzy inference to solve the problem of multiple instance learning (MIL) is presented. The framework introduces a new class of fuzzy inference systems called Multiple Instance Fuzzy Inference Systems (MI-FIS).

Fuzzy inference is a powerful modeling framework that can handle computing with knowledge uncertainty and measurement imprecision e♂ectively. Fuzzy Inference performs a nonlinear mapping from an input space to an output space by deriving conclusions from a set of fuzzy if-then rules and known facts. Rules can be identi♀ed from expert knowledge, or learned from data.

In multiple instance problems, the training data is ambiguously labeled. Instances are grouped into bags, labels of bags are known but not those of individual instances. MIL deals with learning a classi♀er at the bag level. Over the years, many solutions to this problem have been proposed. However, no MIL formulation employing fuzzy inference exists in the literature.

In this dissertation, we introduce multiple instance fuzzy logic that enables fuzzy reasoning with bags of instances. Accordingly, di♂erent multiple instance fuzzy inference styles are proposed. The Multiple Instance Mamdani style fuzzy inference (MI-Mamdani) extends the standard Mamdani style inference to compute with multiple instances. The Multiple Instance Sugeno style fuzzy inference (MI-Sugeno) is an extension of the standard Sugeno style inference to handle reasoning with multiple instances.

In addition to the MI-FIS inference styles, one of the main contributions of this work is an adaptive neuro-fuzzy architecture designed to handle bags of instances as input and capable of learning from ambiguously labeled data. The proposed architecture, called Multiple Instance-ANFIS (MI-ANFIS), extends the standard Adaptive Neuro Fuzzy Inference System (ANFIS).

We also propose di♂erent methods to identify and learn fuzzy if-then rules in the context of MIL. In particular, a novel learning algorithm for MI-ANFIS is derived. The learning is achieved by using the backpropagation algorithm to identify the premise parameters and consequent parameters of the network.

The proposed framework is tested and validated using synthetic and benchmark datasets suitable for MIL problems. Additionally, we apply the proposed Multiple Instance Inference to the problem of region-based image categorization as well as to fuse the output of multiple discrimination algorithms for the purpose of landmine detection using Ground Penetrating Radar.

MULTIPLE INSTANCE FUZZY INFERENCE

By

Amine Ben Khalifa
B.E., Telecommunications Engineering, Higher School of Communications of Tunis, 2009

A Dissertation
Submitted to the Faculty of the
J.B. Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy in Computer Science and Engineering

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

December 2015

Copyright 2015 by Amine Ben Khalifa

All rights reserved

MULTIPLE INSTANCE FUZZY INFERENCE

By

Amine Ben Khalifa
B.E., Telecommunications Engineering, Higher School of Communications of Tunis, 2009

A Dissertation Approved On

December 2, 2015

by the following Dissertation Committee:

Hichem Frigui, Ph.D., Dissertation Director

Olfa Nasraoui, Ph.D.

Jacek M. Zurada, Ph.D.

Adrian P. Lauf, Ph.D.

Tim Hardin, Ph.D.

ACKNOWLEDGEMENTS

I would like to gratefully and sincerely thank my advisor, Dr. Hichem Frigui, for giving me the opportunity to conduct research under his supervision. As a mentor he kept pushing me to challenge the status quo and advance the state of the art of our field. He worked with me side by side on a daily basis to explain difficult concepts early on in my studies and to make sure I was on the right track – all while giving me a great degree of independence to entertain new ideas and explore new avenues. Dr. Frigui, I truly appreciate all of the financial support you provided for travel to domestic and international conferences to present our research and represent the Multimedia Research Lab as well as the University of Louisville. It has been a pleasure working with you. I sincerely hope that we will remain both collaborators and friends for many years to come.

I would like to thank Dr. Olfa Nasraoui, Dr. Adrian Lauf, Dr. Jacek Zurada and Dr. Tim Hardin for agreeing to serve on my dissertation committee and being a part of this special milestone.

I would like to thank my colleagues in the Multimedia Research Laboratory, and the Computer Engineering and Computer Science Department for their support and friendship.

Finally, and most importantly, I would like to thank my family for their support, encouragement, and love.

ABSTRACT

MULTIPLE INSTANCE FUZZY INFERENCE

Amine Ben Khalifa

December 2, 2015

A novel fuzzy learning framework that employs fuzzy inference to solve the problem of multiple instance learning (MIL) is presented. The framework introduces a new class of fuzzy inference systems called Multiple Instance Fuzzy Inference Systems (MI-FIS).

Fuzzy inference is a powerful modeling framework that can handle computing with knowledge uncertainty and measurement imprecision effectively. Fuzzy Inference performs a nonlinear mapping from an input space to an output space by deriving conclusions from a set of fuzzy if-then rules and known facts. Rules can be identified from expert knowledge, or learned from data.

In multiple instance problems, the training data is ambiguously labeled. Instances are grouped into bags, labels of bags are known but not those of individual instances. MIL deals with learning a classifier at the bag level. Over the years, many solutions to this problem have been proposed. However, no MIL formulation employing fuzzy inference exists in the literature.

In this dissertation, we introduce multiple instance fuzzy logic that enables fuzzy reasoning with bags of instances. Accordingly, different multiple instance fuzzy inference styles are proposed. The Multiple Instance Mamdani style fuzzy inference (MI-Mamdani) extends the standard Mamdani style inference to compute with multiple instances. The Multiple Instance Sugeno style fuzzy inference (MI-Sugeno) is an extension of the standard Sugeno

style inference to handle reasoning with multiple instances.

In addition to the MI-FIS inference styles, one of the main contributions of this work is an adaptive neuro-fuzzy architecture designed to handle bags of instances as input and capable of learning from ambiguously labeled data. The proposed architecture, called Multiple Instance-ANFIS (MI-ANFIS), extends the standard Adaptive Neuro Fuzzy Inference System (ANFIS).

We also propose different methods to identify and learn fuzzy if-then rules in the context of MIL. In particular, a novel learning algorithm for MI-ANFIS is derived. The learning is achieved by using the backpropagation algorithm to identify the premise parameters and consequent parameters of the network.

The proposed framework is tested and validated using synthetic and benchmark datasets suitable for MIL problems. Additionally, we apply the proposed Multiple Instance Inference to the problem of region-based image categorization as well as to fuse the output of multiple discrimination algorithms for the purpose of landmine detection using Ground Penetrating Radar.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
LIST OF TABLES x
LIST OF FIGURES xii
LIST OF ALGORITHMS xv

CHAPTER

1 INTRODUCTION 1
1.1 Multiple Instance Learning 3
1.2 Fuzzy Logic and Fuzzy Inference Systems 4
1.3 Motivations and Contributions 6
1.3.1 Motivations 6
1.3.2 Contributions 7

2 BACKGROUND 9
2.1 Multiple Instance Learning 9
2.1.1 Diverse Density 11
2.1.2 Multiple Instance Regression 14
2.1.3 Multiple Instance Learning via Embedded Instance Selection (MILES) 15
2.1.4 Multiple Instance Neural Networks 16
2.1.5 Multiple Instance RBF Neural Networks 18
2.1.6 Citation K-Nearest Neighbors 20
2.2 Fuzzy Logic 21
2.2.1 Fuzzy Sets 22
2.2.2 Fuzzy Propositions 28

2.2.3 Fuzzy If-Then Rules 28
2.2.4 Fuzzy Reasoning 29
2.3 Fuzzy Inference 31
2.3.1 Mamdani Fuzzy Inference System 31
2.3.2 Sugeno Fuzzy Inference System 33
2.3.3 ANFIS: Adaptive Neuro-Fuzzy Inference System 34

3 MULTIPLE INSTANCE FUZZY LOGIC 42
3.1 Multiple Instance Fuzzy Propositions 42
3.2 Multiple Instance Fuzzy If-Then Rules 45
3.2.1 Multiple Instance Fuzzy Implication 45
3.3 Multiple Instance Fuzzy Reasoning 46
3.4 Illustrative Example 48
3.5 Discussion 48
3.6 Related Work 49
3.7 Chapter Summary 51

4 MULTIPLE INSTANCE FUZZY INFERENCE 52
4.1 Multiple Instance Mamdani Style Fuzzy Inference 52
4.2 Multiple Instance Sugeno Style Fuzzy Inference 55
4.3 Learning the Structure and Parameters of Multiple Instance Fuzzy Inference Systems 56
4.3.1 Illustrative Example 59
4.4 Chapter Summary 63

5 MI-ANFIS: A MULTIPLE INSTANCE ADAPTIVE NEURO-FUZZY ARCHITECTURE 64
5.1 MI-ANFIS Architecture 64
5.2 Basic Learning Algorithm 68
5.2.1 BackPropagation Learning Rule 68

5.3 Illustrative Example 74
5.4 Preventing Overfitting: Rule Dropout 77
5.5 Multi-Class MI-ANFIS 83
5.6 Complexity Analysis 87
5.7 Discussion 88

6 EXPERIMENTAL RESULTS 90
6.1 Benchmark Datasets 90
6.2 Evaluation of MI-MAMDANI and MI-ANFIS algorithms 92
6.3 MCMI-ANFIS 96

7 APPLICATION : LANDMINE DETECTION USING GROUND PENETRATING RADAR 102
7.1 Landmine Detection 102
7.1.1 GPR data 104
7.1.2 EHDDT and EHDCT algorithms 105
7.1.3 Fisher Vector discrimination algorithms 106
7.1.4 Auxiliary Feature Extraction 108
7.1.5 Data Collection 112
7.2 Fusion of Multiple Landmine Detection Algorithms Using Traditional Fuzzy Inference 113
7.2.1 Fusion of Multiple Landmine Detection Algorithms Using Mamdani Fuzzy Inference 113
7.2.2 Fusion of Multiple Landmine Detection Algorithms Using ANFIS 117
7.2.3 Results 120
7.3 Fusion of Multiple Landmine Detection Algorithms Using Multiple Instance Fuzzy Inference 122
7.3.1 Fusion of Multiple Landmine Detection Algorithms Using MI-Mamdani 123

7.3.2 Fusion of Multiple Landmine Detection Algorithms Using MI-ANFIS 124
7.3.3 Results 125

8 CONCLUSIONS AND POTENTIAL FUTURE WORK 132
8.1 Conclusions 132
8.1.1 Multiple Instance Fuzzy Logic (MI-FL) 132
8.1.2 Multiple Instance Fuzzy Inference Systems (MI-FIS) 133
8.1.3 Multiple Instance Adaptive Neuro-Fuzzy Inference System (MI-ANFIS) 133
8.1.4 Validation 135
8.2 Potential Future Work 136

REFERENCES 137

CURRICULUM VITAE 145

LIST OF TABLES

TABLE Page

2.1 Shape and parameters of commonly used parameterized MFs 24

2.2 Most frequently used T-norms and T-conorms operators 26

5.1 Benchmark data sets 83

6.1 MUSK, Fox, Tiger, and Elephant Datasets 92

6.2 20 Image Categories of the COREL dataset and the Corresponding Average Number of Instances (regions) 93

6.3 MI-ANFIS Training Parameters 94

6.4 Comparison of MI-ANFIS prediction accuracy (in percent) to Naive-ANFIS on the benchmark data sets (averaged over five runs) 94

6.5 Comparison of MI-ANFIS prediction accuracy (in percent) to other methods on the benchmark data sets. Results for 3 top performing methods are shown in bold font. We use reported results, N/A indicated that a given algorithm was not applied to that dataset 95

6.6 Comparison of MI-ANFIS running time (in Minutes) to other methods on the benchmark data sets. 96

6.7 MCMI-ANFIS Training Parameters 97

6.8 Confusion matrix of MCMI-ANFIS on the region-based image categorization experiment using Corel-1000 Dataset 98

6.9 Confusion matrix of MCMI-ANFIS on the region-based image categorization experiment using Corel-2000 Dataset 99

6.10 Comparison of MCMI-ANFIS classification accuracy (in percent) to other methods on the Corel-1000 and Corel-2000 benchmark datasets 100

7.1 MI-Mamdani Parameters 123
7.2 MI-ANFIS Training Parameters 125
7.3 Data Collections 130

LIST OF FIGURES

FIGURE Page

1.1 Example of an image represented as a bag of 12 instances 4

1.2 A graphical representation of a FIS and its components. 5

1.3 Linguistic terms of Color Intensity and Vertical Position features. 7

2.1 Difference between standard supervised learning and multiple instance learning 10

2.2 Illustration of a Ramon & Raedt's multiple instance neural network [1]. . . 18

2.3 Illustration of an RBF-MIP neural network with a single output. 19

2.4 An illustration of the crisp membership function "Young" 23

2.5 An illustration of the fuzzy membership function "Young" 23

2.6 Illustration of α-cut, core, and support of a bell-shaped membership function 25

2.7 Illustration of the fuzzy reasoning process 31

2.8 Illustration of Mamdani fuzzy inference with 2 rules and D inputs. 32

2.9 Illustration of Sugeno fuzzy inference with 2 rules and D inputs. 34

2.10 Architecture of an ANFIS system with two-input and two rules. 35

3.1 Illustration of the proposed multiple instance inference process using the "max" aggregation operator 48

4.1 Illustration of the proposed multiple instance Mamdani fuzzy inference system. 54

4.2 Illustration of the proposed multiple instance Sugeno fuzzy inference system 56

4.3 Instances from positive and negative bags drawn from data that have 2 concepts 60

4.4 Illustration of MI-Mamdani fuzzy inference system learned using FCMI . . 61

4.5 Instances from 2 positive and 1 negative bag. 61

4.6 Multiple instance fuzzy inference using the learned MI-Mamdani system. . . 62

5.1 Architecture of the proposed multiple instance Adaptive Neuro-Fuzzy Inference System 64
5.2 Root Mean Squared Error of 100 Epochs of MI-ANFIS training. 75
5.3 Input MFs before, during, and after MI-ANFIS training. 76
5.4 Dropout neural network model 78
5.5 Illustration of Dropout application. 78
5.6 Rule Dropout MI-ANFIS model. 80
5.7 Illustration of Rule Dropout application. Doted lines indicate a rule that has been dropped. 81
5.8 Training and testing errors for two MI-ANFIS networks with and without Rule Dropout. 83
5.9 Multi-Class MI-ANFIS with R rules and T classes (outputs). 85
6.1 Images randomly sampled from 20 categories and the corresponding segmentation results. 92
7.1 3-dimensional and 2-dimensional raw GPR data. 103
7.2 Vehicle mounted GPR system. 105
7.3 Sample GPR alarm with dense SIFT features (only first and last features are shown) 107
7.4 Target and Non-Target signatures. 110
7.5 Illustration of the identification of the SA and SD points. 111
7.6 Examples of SignatureWidthDT (gprDT) and SignatureWidthCT (gprCT) features for a target object. 112
7.7 Illustration of the generated Mamdani Fuzzy Rule Base (FRB), showing 4 of the 21 rules. 117
7.8 Illustration of the generated ANFIS Fuzzy Rule Base (FRB), showing 2 of the 16 rules. 120
7.9 Comparison of the performances of EHDDT and EHDCT discriminators. . 121

7.10 Comparison of the individual discriminators and the proposed fuzzy fusion method. 122
7.11 MI-Mamdami multiple instance fuzzy rules. 124
7.12 MI-ANIFS fusion rules before and after training (Dotted lines indicate the initial MFs). 126
7.13 A plot of MI-ANFIS RMSE during 150 training epochs. 127
7.14 Comparison of the individual discriminators and all proposed fuzzy fusion methods. 128
7.15 Comparison of the individual discriminators, the proposed MI-Mamdani , Mamdani, and NaiveMamdanifuzzy fusion methods. 128
7.16 Comparison of the individual discriminators, the proposed MI-ANFIS , ANFIS, and NaiveANFIS fuzzy fusion methods. 129
7.17 Comparison of all individual discriminators, ANFIS, and the proposed MI-ANFIS fuzzy fusion methods. 130
7.18 Comparison of the performances of all individual discriminators, ANFIS, and MI-ANFIS fuzzy fusion methods on the larger collection. 131

LIST OF ALGORITHMS

ALGORITHM Page

2.1 Multiple-Instance Regression Algorithm 16

2.2 First Layer's Clustering Algorithm of RBF-MIP 20

2.3 ANFIS Basic Learning Algorithm 41

5.1 MI-ANFIS Forward Pass Algorithm 67

5.2 MI-ANFIS Basic Learning Algorithm 74

CHAPTER 1

INTRODUCTION

Fuzzy inference is a powerful modeling framework that can handle computing with knowledge uncertainty and measurements imprecision effectively [2]. It is a process based on the concepts of fuzzy set theory and fuzzy reasoning. It performs a non-linear mapping from an input space to an output space by deriving conclusions from a set of fuzzy if-then rules and known facts [3]. Fuzzy inference has been successfully applied to a wide range of problems, mainly in system modeling and control [4–14]. Most of the proposed fuzzy inference methods gained success because of their ability to leverage expert knowledge to identify the model parameters [15]. This practice simplifies system design and ensures that the knowledge base (if-then rules) used by the system is easy to interpret [16].

More recently, fuzzy inference has increasingly been applied to more advanced applications, such as content-based information retrieval [17], image segmentation [18], image annotation [19], pattern recognition [20], recommender systems [21, 22], and multiple classifier fusion [23]. The aforementioned applications are more challenging as they require an extensive knowledge base to accommodate for various scenarios. Since this diverse knowledge base cannot be fully provided by domain experts, data-driven techniques are typically used to identify and learn the fuzzy inference system's parameters [24, 25]. In this later technique, supervised and unsupervised learning algorithms are devised to learn the parameters of the fuzzy inference system (i.e. learn the knowledge base) from a set of labeled training data. For instance, a clustering algorithm (unsupervised learning) can be used to identify local contexts of the input space, and a linear classifier (supervised learning) can be used to learn decisions within each of the contexts. Thus, substituting the traditional expert knowledge based system's identification methods, with more scalable, adaptive, and

broader learning methods.

Typically, in supervised learning problems, access to large labeled training datasets improves the performance of the devised algorithms by increasing their robustness and generalization capabilities. Nowadays, access to such large datasets is becoming more convenient. In fact, we generate about 2.5 quintillion bytes of data everyday [1] [26, 27]. This data is continuously collected from sensors that measure environmental information, posts to social media sites such as flickr [28], digital pictures and videos uploaded to advertisement websites such as Craigslist [29], etc. This trend is not expected to slowdown anytime soon and is fueled by the drastic decrease in the cost of data storage [30]. However, for a supervised leaning method to benefit from this data, it needs to be carefully preprocessed, filtered, and labeled. Unfortunately, this process can be too tedious as the vast portion of the collected data is unstructured with few tags that label the object at a high level (e.g. social media images). To overcome this lack of labeled data, many recent developments use crowdsourcing services such as Amazon Mechanical Turk [31] to hire an on-demand human workforce over the internet to assign labels to data points. For instance, a tool named "Labelme" by MIT [32] could be used for this purpose. Similarly, Google started using its Captcha service, reCaptcha [33], to label address' digits collected from Street View images for the purpose of a deep neural network training [34]. Despite the scalability of many recent machine learning algorithms, they still require the full engaged cognition of a human being to assign labels at a finer level (e.g. label regions within images). Unfortunately, this process is ambiguous, subjective, and prone to errors (e.g. difficulty to select an object of interest within an image).

To summarize, large amounts of data are available and could be used for learning. However, this data is typically labeled ambiguously and at a coarse level. In fact, labels, or tags, tend to be associated with collections of samples rather than single samples. For example, in image annotation, tags could be used as indicators of the existence of objects of interests within the images (sky, sea, beach,...). However, the exact location of those

[1] 90% of the data in the world today has been created in the last two years alone.

objects is not available and is too tedious to extract for large collection of images. An alternative and a relatively new framework of learning that tackles the inherent ambiguity better than supervised learning, is the Multiple Instance Learning (MIL) paradigm [35].

1.1 Multiple Instance Learning

Unlike standard supervised learning, in MIL, an object is not represented by a simple data point, but rather by a collection of instances, called a bag. Each bag can contain a different number of instances. In MIL, a bag is labeled negative if all of its instances are negative, and positive if at least one of its instances is positive (positive bags may also contain negative instances). Positive bags can encode ambiguity since the instances themselves are not labeled. Given a training set of labeled bags, the goal of MIL is to learn a concept that predicts the labels of training data at the instance level and generalizes to predict the labels of testing bags and their instances [36].

The MIL problem was first formalized by Dietterich et al. [37] providing a solution to drug activity prediction. Ever since, it has increasingly been applied to a wide variety of tasks. Some of the applications include content-based information retrieval [38], drug discovery [39], pattern recognition [40], image classification [41], text classification [42], region-based image categorization [43], image annotation [44], object tracking [45] and time-series prediction [35], to name a few. To illustrate the need for MIL, in the following we analyse how a multiple instance (MI) representation can be applied to image classification.

Consider the simple example of classifying images that contain "sky". In this problem, for an input image we want to determine whether a region that contains sky is present in the image. Using an MIL approach, each training image is represented by a bag of instances where each instance corresponds to a segmented region of interest. These regions could be obtained by dividing images into patches. A multiple instance representation is well suited for this purpose because only few regions may contain the object of interest (sky), that is the positive class. Other patches will be from background or other classes. This representation is illustrated in Figure 1.1. Traditional, single instance learning methods

are based on instance level (patch-level) labels and would require the image to be correctly segmented and labeled prior to learning.

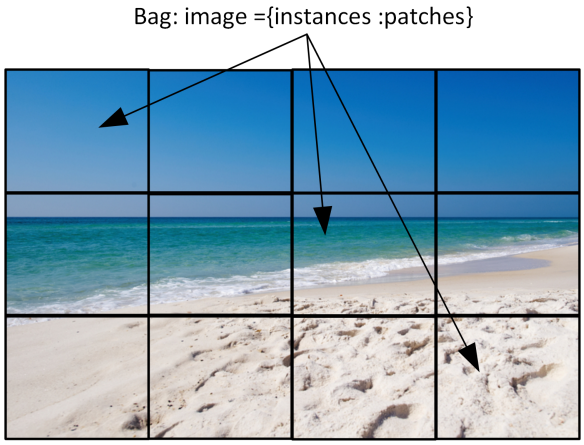


Figure 1.1: Example of an image represented as a bag of 12 instances. Each instance could be a feature vector extracted from one patch. The bag is labeled "sky" because at least one of its instances is sky. However, many other instances are not "sky". Labels at the instance level are not available.

1.2 Fuzzy Logic and Fuzzy Inference Systems

Fuzzy logic [4] is a computational framework that makes use of fuzzy set theory and fuzzy assignment of elements to sets. In classical set theory, also known as crisp sets, an element is either a member of a set or not. Whereas, in fuzzy set theory, an element is characterized by a degree of membership, usually a real number between 0 and 1. Fuzzy logic, in contrast to traditional two-valued (boolean) logic, uses the elements' membership degrees to evaluate the degree of truth of logical propositions. Hence, the degree of truth is non-crisp, or soft. This enables fuzzy logic to be characterized by linguistic terms rather than by numbers. For example, in fuzzy logic, a fuzzy proposition can have the following expression: "patch is blue", in which the linguistic term "blue" is a fuzzy set that describes color intensity. Fuzzy logic simulates human imprecise understanding of the world, and can be viewed as a framework for computing with words [46].

A Fuzzy Inference System (FIS) is a paradigm in soft computing which provides a means of approximate reasoning [47]. A FIS is capable of handling computing with

knowledge uncertainty and measurements imprecision effectively [2]. It performs a non-linear mapping from an input space to an output space by deriving conclusions from a set of fuzzy if-then rules and known facts. Fuzzy rules are condition/action (if-then) rules composed of a set of linguistic variables (e.g. patch) which can each take on linguistic terms (e.g. red, green, blue). For example, the following rules could be used to identify patches from the image in Figure 1.1:

- If *patch* is *blue* then *region* is *sky*.
- If *patch* is *blue* and *patch position* is *upper half* then *region* is *sky*.
- If *patch* is *yellow* and *patch position* is *lower half* then *region* is *beach*.

Typically, a FIS is composed of 5 components. First, a *Fuzzification* unit that assigns a membership degree to each crisp input dimension in the input fuzzy sets. Second, a *Knowledge Base* characterized by fuzzy sets of linguistic terms. Third, a *Rule Base* containing a set of fuzzy if-then rules. Fourth, an *Inference unit* that performs fuzzy reasoning. Finally, a *Deffuzification unit* that generates crisp output values. Mamdani [48] and Sugeno [49] are the two commonly used fuzzy inference systems. A graphical representation of a generic FIS is shown in Figure 1.2.

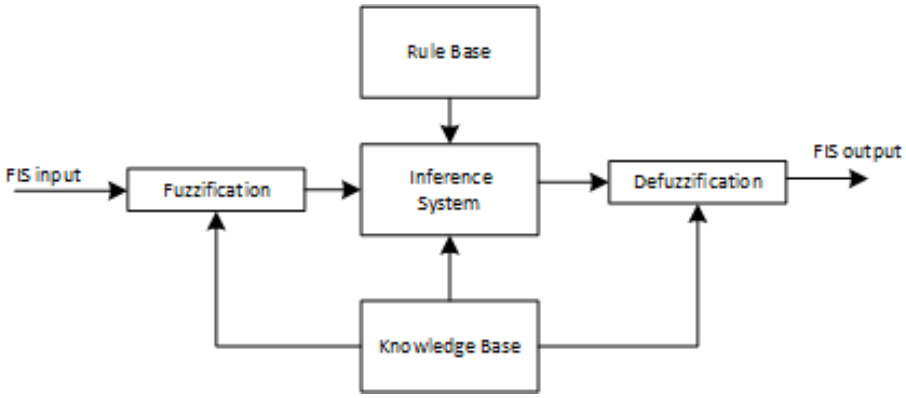


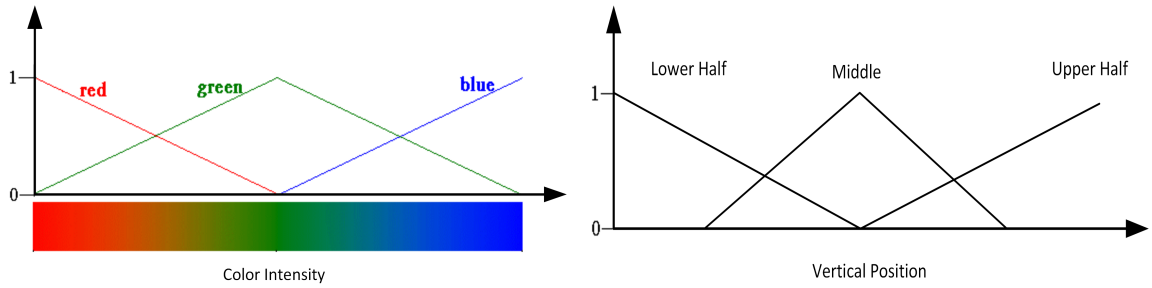
Figure 1.2: A graphical representation of a FIS and its components.

1.3 Motivations and Contributions

1.3.1 Motivations

Consider the example of sky image classification presented earlier. Let us suppose we have assembled a training dataset with images labeled as positive if they contain sky, negative otherwise. Clearly, the data is ambiguously labeled (i.e. labels are available only at the bag level, and individual patches are not labeled). We want to train a FIS capable of recognizing images containing sky (i.e. produce a high output when the test image contains a sky region). To do so, the FIS needs to learn rules capable of describing the concept of sky. For the human perception, the sky concept can be described as a blue region that is located in the upper half of the frame. Thus, one possibility is to extract 2 features: Color Intensity, and Vertical Position of a region. Doing so, features need to be extracted locally at the patch level. Hence, turning the problem into a multiple instance problem, an image is a bag of instances (with label only at the bag level). On the other hand, extracting one global feature vector covering the whole image will lead to confusions and will not be able to describe concepts effectively because the features will describe non homogenous regions and will be based on averages. Because of the uncertainty and subjectivity of describing the color of a patch and its vertical position within the image, the two features are better represented as fuzzy sets. Color intensity feature can be described by means of 3 linguistic terms: Red, Green, and Blue. While vertical position can be described with linguistic terms, Upper Half, Middle, and Lower Half. Figure 1.3 shows a graphical representation of membership degrees of the 2 features in the different linguistic terms (fuzzy sets).

Clearly, this representation is close to the way humans perceive the patches of the image in Figure 1.1. Due to the absence of labels at the patch level, and therefore absence of feedback, FIS training could not be achieved. Nonetheless, in this particular example the concept of the sky can be considered trivial. Thus, the FIS can be designed by leveraging expert knowledge that can lead to using the following rule for the purpose of patch classification.



(a) A graphical representation of 3 fuzzy sets describing the Color Intensity feature.

(b) A graphical representation of 3 fuzzy sets describing the Vertical Position feature.

Figure 1.3: Linguistic terms of Color Intensity and Vertical Position features.

- If *patch* is *blue* and *patch position* is *upper half* then *region* is *sky*.

To classify the image correctly, the results of patches' classification need to be aggregates to produce a final output.

There are two major limitations that prevent using standard FIS methods with multiple instance data. First, due to the absence of labels at the instance level, we cannot use standard FIS learning methods to construct the knowledge base. Second, we need an effective mechanism to aggregate instances' confidences and infer at the bag level.
The limitations are due mainly to the inherent architecture of fuzzy inference systems. The generic inference system shown in Figure 1.2 reasons with individual instances. First, the system's input is an individual instance. Second, the rules describe fuzzy regions within the instances space. Third, the output of the system corresponds to the fuzzy inference using a single instance. Fourth, labels of the individual instances are required when using learning techniques to identify the parameters of the system. In summary, traditional fuzzy inference systems cannot be used effectively within the MIL framework.

To address the above limitations we propose to generalize fuzzy inference to extend it to reason with bags of instances.

1.3.2 Contributions

In this dissertation, we propose developing a Multiple Instance Fuzzy Inference framework. In particular, we propose:

1. Developing Multiple Instance Fuzzy Logic that generalizes traditional fuzzy logic to compute with bags of instances. Under this work, we propose multiple instance generalization of fuzzy propositions, fuzzy if-then rules, fuzzy implication, and fuzzy reasoning.

2. Extending Mamdani and Sugeno fuzzy inference systems to reason with bags instead of individual instances using the developed Multiple Instance Fuzzy Logic. We call the new inference systems Multiple Instance-Mamdani (MI-Mamdani) and Multiple Instance-Sugeno (MI-Sugeno).

3. Developing methods to identify and learn multiple instance fuzzy if-then rules from ambiguously labeled data.

4. Extending the standard Adaptive Neuro-Fuzzy Inference System (ANFIS) [50] to reason with bags of instances as input and to learn from ambiguously labeled data. We call the new neuro-fuzzy architecture Multiple Instance-ANFIS (MI-ANFIS).

5. Developing a learning algorithm to learn the parameters of the proposed MI-ANFIS neuro-fuzzy inference system.

The remainder of this dissertation is organized as follows. Chapter 2 provides a review of multiple instance learning, fuzzy logic, and common fuzzy inference systems. Chapter 3 introduces our proposed multiple instance fuzzy logic framework. Chapter 4 introduces our proposed MI-Mamdani and MI-Sugeno inference systems. Chapter 5 introduces our proposed MI-ANIFS neuro-fuzzy architecture. Chapter 6 provides experimental results and analysis of the proposed methods. Finally, chapter 7 provides conclusions and future work.

CHAPTER 2

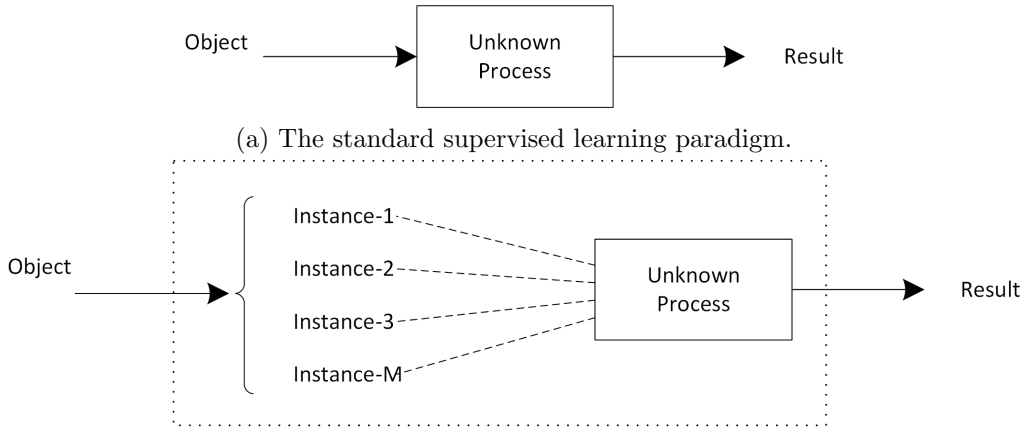
BACKGROUND

In this chapter, we provide background material that is relevant to our research. We start with a review of the Multiple Instance Learning problem and give brief examples to motivate the need for this learning paradigm. Next, we provide an overview of fuzzy logic. Finally, we provide an overview of common fuzzy inference systems.

2.1 Multiple Instance Learning

Multiple Instance Learning (MIL) is a supervised learning paradigm that aims at solving classification and regression problems by devising algorithms capable of learning from ambiguously labeled data [51]. In standard supervised learning, each example is represented by a fixed-length vector of features. In MIL, an example is a collection of feature vectors (instances), called a bag. Each bag can contain a different number of instances. Labels of bags are known but not those of individual instances. A bag is labeled negative if all of its instances are negative, and positive if at least one of its instances is positive. Positive bags can encode ambiguity since the instances themselves are not labeled. Given a training set of labeled bags, the goal of MIL is to learn a concept that predicts the labels of training data and generalizes to predict the labels of testing bags [36]. The difference between standard supervised learning and multi-instance learning is illustrated in Figure 2.1.

The problem of MIL arises naturally in many scenarios. It was first applied by Dietterich et al. to provide a solution to drug activity prediction [37]. Ever since, it has increasingly been applied to a wide variety of tasks such as content-based information retrieval [38], pattern recognition [40], image classification [41], text classification [42], region-



(a) The standard supervised learning paradigm.

(b) The multiple instance learning paradigm.

Figure 2.1: Difference between standard supervised learning and multiple instance learning.[1]

based image categorization [43], image annotation [44], object tracking [45] and timeseries prediction [35], to name a few. MIL has a broader domain of application beyond those few examples. Maron et al. [35] presented a methodology to transform difficult learning problems into Multiple-Instance learning problems.

In general, MIL can be applied in two contexts of ambiguity: "polymorphism ambiguity" and "part-whole ambiguity" [52]. In polymorphism ambiguity, an object can have multiple forms of expression in the input space and it is not known which form is responsible for the object label. Whereas, in part-whole ambiguity, an object can be broken into several parts represented by different feature vectors in the input space. However, only few parts are responsible for the object label [53]. In the following we briefly describe two application domains related to the two distinct ambiguity concepts.

1. **Polymorphism Ambiguity** arise more often in applications related to chemistry and bioscience. The original MIL application of drug discovery [35, 36] is a case of polymorphism ambiguity. In this type of applications, typically, the goal is to classify molecules by looking at their shapes. Each molecule can appear in several distinct shapes because of binding and twisting that might occur during interactions. Thus, a molecule can have different forms of expression. However, it is a tedious process to identify which form is responsible for the molecule behaviour (label). Hence,

[1]Figure based on [36].

the problem is better represented as a multiple instance problem. A more recent application that presents polymorphism ambiguity is genomic data analyses [54]. In this type of applications a gene is represented by multiple isoforms, the goal is to predict the gene-level function. Typically, this problem is a multiple instance problem.

2. **Part-whole Ambiguity:** This type of ambiguity is more common in pattern recognition problems. For example, for an image annotation application such as presented in Section 1.1, usually features are extracted locally (from patches) with labels, or tags, available only at the image level, making the problem a multiple instance problem. Another closely related application is object detection. In this application objects of interest cover only a limited region of the image, the rest could be other objects or background. For the task of training a classifier to detect the object, traditionally, tedious human labor is required to extract patches containing the object and labeling them. As indicated by Viola et al. [55], placing bounding boxes around objects is an inherently ambiguous task. Thus, it is more convenient to solve the problem of object detection using the MIL paradigm, which in turn encodes ambiguity effectively. The part-whole ambiguity also arise in other applications such as computer audition [56] and text document classification [57]. These applications are similar to object detection: features are extracted from audio segments or text paragraphs, and labels are only available at the audio clip level or text document level, respectively.

We now review some of the common algorithms that have been proposed to solve the multiple instance problem and are related to our research.

2.1.1 Diverse Density

The most commonly referenced MIL algorithm found in the literature is Diverse Density (DD). It was first introduced by Maron et al. [39]. The objective of DD is to find a "soft" set that describes the intersection of the positive bags minus the union of the negative bags. To achieve this, DD attempts to find a concept in the feature space (instance space) that is close to at least one instance from every positive bag but far away from instances in

the negative bags.

If we define diverse density as a measure of how many different positive bags have instances near a given point of the input space, and how far the negative instances are from that point, then a concept as defined by Maron et al. [39] as a point with maximum diverse density.

Formally, if the training data is presented as positive bags, denoted $B_1^+, B_2^+, \ldots, B_n^+$, and negative bags, denoted $B_1^-, B_2^-, \ldots, B_m^-$, the diverse density of a given concept t is defined as the probability that t is the correct concept.

$$DD(t) = Pr(t \mid B_1^+, B_2^+, \ldots, B_n^+, B_1^-, B_2^-, \ldots, B_m^-). \tag{2.1}$$

Using Bayes' rule and under the assumption that all bags are conditionally independent given the true target concept, it was shown that (2.1) can be decomposed into:

$$DD(t) = \prod_{1 \le i \le n} Pr(B_i^+ \mid t) \prod_{1 \le i \le m} Pr(B_i^- \mid t). \tag{2.2}$$

Using Bayes' rule further and under the assumption of constant priors, Maron showed that optimizing DD is equivalent to optimizing $\widehat{DD}$, defined as

$$\widehat{DD}(t) = \prod_{1 \le i \le n} Pr(t \mid B_i^+) \prod_{1 \le i \le m} Pr(t \mid B_i^-). \tag{2.3}$$

Instead of maximizing $\widehat{DD}$, a common practice is to minimize the negative log-likelihood given by:

$$-log\widehat{DD}(t) = -\left[\sum_{1 \le i \le n} log(Pr(t \mid B_i^+)) + \sum_{1 \le i \le m} log(Pr(t \mid B_i^-)) \right]. \tag{2.4}$$

This formulation is more robust against very small probabilities.

To compute $Pr(t \mid B_i)$ for a given bag B_i, a conjunction measure of all its instances B_{ij}, $j = 1, \ldots, M$ is computed using the noisy-or operator

$$Pr(t \mid B_i) = 1 - \prod_{1 \le j \le M} (1 - Pr(B_{ij} \in t)), \tag{2.5}$$

where $Pr(B_{ij} \in t)$ is computed from a Gaussian distribution centred at the concept point t.

To optimize the above cost function (2.4), gradient descent can be used to find an optimal target concept t. Another optimization technique that can be used to find the most likely concept is EM-DD [58]. The basic idea behind EM-DD is to view "the knowledge of which instance corresponds to the label of the bag as a missing attribute which can be estimated using the Expectation Maximization (EM) approach". The EM-DD starts by taking an initial guess from positive instances as a target concept, then alternates between two steps: In the first step, the current concept is used to pick one instance from each bag which is most likely responsible for the bag label, and in the second step, find a new target concept t' by maximizing the likelihood over all negative instances and the positive instances identified by the first step.

Once concepts are identified using DD or EM-DD, the label for an unseen bag B_{new} (with M instances) in a given concept t is estimated as following:

$$Label(B_{new} \mid t) = \max_{k} \Big\{ exp(-(B_{new,k} - t)^2) \Big\},\ k = 1, \ldots, M. \tag{2.6}$$

- **Multi-target concept Diverse Density (MDD)**

  The MDD is a new metric developed by Karem and Frigui [59] for the purpose of fuzzy clustering of multiple instance data (FCMI). This approach extends the standard Diverse Density (DD) metric established by Maron to accommodate more than one positive target concept. The governing assumption behind this extension is that there exist MIL problems for which a single target concept inadequately represents the feature space.

  In MDD, there are multiple target concepts $\{C_1, \ldots, C_r\}$, and each bag is assigned memberships to multiple target concepts. This membership assignment is conducted by selecting the concept that maximizes the noisy-or measure (2.5). Once memberships are assigned to each bag, the target concepts are optimized separately following the pattern of the general DD methodology. The MDD metric is given by the following:

$$MDD(\mathbf{T}, \mathbf{U}) = \prod_{n=1}^{N} \prod_{i=1}^{r} \left[Pr(C_i|B_n)\right]^{u_{in}^m}. \tag{2.7}$$

In (4.8), $\mathbf{U} = [u_{in}]$ is a membership matrix such that each bag B_n is assigned to target concept C_i with membership degree u_{in}, and m is a fuzzifier that controls the fuzziness of the partitions as in the FCM [60]. $Pr(C_i|B_n)$ is the probability that C_i is a target concept given B_n, and defined as

$$Pr(C_i|B_n) = \begin{cases} 1 - \prod_{k=1}^{M}(1 - Pr(x_{nk} \in C_i)) & if\ label(B_n) = 1, \\ \prod_{k=1}^{M}(1 - Pr(x_{nk} \in C_i)) & if\ label(B_n) = 0 \end{cases} \tag{2.8}$$

where $label(B_n)$ is the label of bag B_n and x_{nk} is the kth instance of bag B_n.

$Pr(X_{nk} \in C_i)$ is regarded as the similarity of instance X_{nk} to target concept C_i, and its computed using

$$Pr(X_{nk} \in C_i) = e^{-(\sum_{j=1}^{D} s_{ij}(x_{nkj} - c_{ij})^2)} \tag{2.9}$$

In (4.5), s_{ij} is a scaling parameter that weights the role of feature j in target concept i [39].

2.1.2 Multiple Instance Regression

Multiple instance regression (MI-regression) was first introduced by Ray and Page [61]. In MI-regression, bags are associated with real-valued labels instead of the usual binary class labels (positive/negative). Similarly to the standard regression, the task of MI-regression is to predict a real-valued bag label.

Ray and Page assumed that every bag has a primary instance responsible for the bag label. Under this assumption an ideal regression model is a hyperplane $\mathbf{Y} = \mathbf{Xb}$ such that

$$\hat{\mathbf{b}} = \underset{\mathbf{b}}{argmin} \sum_{i=1}^{N} \mathrm{L}(y_i, X_{ip}, \mathbf{b}), \tag{2.10}$$

where N is the number of bags, y_i is the real-valued label of bag B_i, X_{ip} is the primary instance of the ith bag, and L is a loss function defined by

$$\mathrm{L}(y_i, X_{ij}, \mathbf{b}) = (y_i - X_{ij}\mathbf{b})^2. \tag{2.11}$$

Equation (2.10) assumes that the primary instance X_{ip} is known during training. However, this is not the case for most MIL problems. To overcome this issue, Ray and Page proposed to use the "best fit" hyperplane instead:

$$\hat{\mathbf{b}} = \underset{\mathbf{b}}{argmin} \sum_{i=1}^{N} \underset{j}{min} \mathrm{L}(y_i, X_{ij}, \mathbf{b}),\ j = 1, \ldots, M_i \tag{2.12}$$

where M_i is the number of instances of the ith bag.

To find the optimal set of parameters $\hat{\mathbf{b}}$. Ray and Page proposed an algorithm based on an EM approach. First, a hypothesis hyperplane $\hat{\mathbf{b}}$ is initialized. Then the algorithm alternately iterates between two steps: (1) in the expectation step, from each bag the instance with the least L-error w.r.t. to $\hat{\mathbf{b}}$ is selected, and (2) in the maximization step, ordinary regression is performed to find a new hyperplane that best fits the selected instances. The process continues until convergence. The MI-regression solution is summarized in Algorithm 2.1

2.1.3 Multiple Instance Learning via Embedded Instance Selection (MILES)

MILES was proposed by Chen et al. [43]. The framework converted the MIL problem into a standard supervised learning problem by mapping each bag into a feature space defined by the similarity between its instances and a set of target concepts. Formally, for a given bag B_i of instances X_{ij}, $j = 1, \ldots, M_i$, the similarity to a given target concept t_k, $k = 1, \ldots, C$ (C number of target concepts) is given by:

$$s(t_k, B_i) = \underset{j}{max}\Big\{exp\Big(-\frac{||X_{ij} - t_k||^2}{\sigma^2}\Big)\Big\}, \tag{2.13}$$

where σ is a scaling factor.

Using (2.13), a bag is mapped into the space induced by the similarity values. i.e. a bag is represented by the coordinates $m(B_i)$ as following,

$$m(B_i) = [s(t_1, B_i), s(t_2, B_i), \ldots, s(t_C, B_i)]. \tag{2.14}$$

Algorithm 2.1 Multiple-Instance Regression Algorithm

Inputs: $\mathcal{B}$: the set of training bags.
$\mathcal{T}$: the set of training labels.
b: random initial hyperplane.
M_i: the number of instances in bag i.
N: the number of training bags.
Outputs: A hyperplane $\mathbf{Y} = \mathbf{Xb}$.

$E = \infty$
$Done = flase$
repeat
  $I = \varnothing$
  $Error = 0$
  for each bag B_i **do**
    for each instance X_{ij} in B_i **do**
      $\mathrm{L}(y_i, X_{ij}, \mathbf{b}) = (y_i, X_{ij}\mathbf{b})^2$, [Calculate the error of the instance with respect to the hyperplane]
    end for
    $I = I \cup \{the\ instance\ with\ the\ lowest\ error\}$, Let this error be L_{min}
    $Error = Error + \mathrm{L}_{min}$
  end for
  if $Error \geq E$ **then**
    $Done = true$
  else
    $E = Error$
    solve (2.12) for **b**
  end if
until Done
return **b**.

Considering a binary MIL classification problem, with bag labels of $+1$ and -1, MILES uses 1-Norm SVM [62] to learn a linear classifier on the mapped space. i.e.,

$$y_i = sign(\sum_{k=1}^{C} w_k s(t_k, B_i) + b). \tag{2.15}$$

where w_k is a weight associated with $s(t_k, B_i)$ and b a bias parameter.

2.1.4 Multiple Instance Neural Networks

In this approach Zhou and Zhang [63] proposed to adapt the BackPropagation algorithm [64] for multiple instance learning through employing a modified loss function. For a

given neural network of one or more hidden layers, a bag B_i of instances X_{ij}, $j = 1, \ldots, M_i$, is fed to the network one instance at a time, and for a given instance a partial network error E_{ij} is computed as following:

$$E_{ij} = \begin{cases} 0 & if\ (B_i\ is\ positive)\ and\ (0.5 \leq O_{ij}) \\ 0 & if\ (B_i\ is\ negative)\ and\ (O_{ij} < 0.5) \\ \frac{1}{2}(O_{ij} - 0.5)^2 & otherwise, \end{cases} \tag{2.16}$$

where O_{ij} is the network's computed output when presented with instance X_{ij}.
Given (2.16), the overall network error, E_i, for a given bag B_i is computed using the following heuristic

$$E_i = \begin{cases} \min\limits_{1 \leq j \leq M_i} E_{ij} & if\ (B_i\ is\ positive) \\ \max\limits_{1 \leq j \leq M_i} E_{ij} & if\ (B_i\ is\ negative) \end{cases} \tag{2.17}$$

Using the error defined in (2.17), and given the neural network architecture it is straightforward to derive a backpropagtion update rule for the network's weights. To speedup the training process Zhou suggested that when the partial error, E_{ij}, for a given instance X_{ij} of bag B_i is equal to zero, the rest of instances should be skipped and not fed to the network.

Even though this solution of MIL is supposed to extend neural networks to reason with bags, it is still relying on computing with single instances. Another Multiple instance neural network approach was proposed by Ramon and Raedt [1]. In this work, the authors proposed a neural network architecture composed of two stages:

1. A first stage composed of an ensemble of subnetworks (multilayered perceptrons), $\{Net_j\}_{j=1}^{M_i}$, with count equals to the number of instances of the input bag B_i. All subnetworks of the first stage are identical and share the same weights (Hence, also share the same weight update).

2. A second stage that aggregates the outputs $\{O_j\}_{j=1}^{M_i}$, of all subnetworks. For the purpose of aggregation, this stage uses a differentiable version of the "max" function,

$dmax$, defined as:

$$dmax_{\alpha}(O_1, O_2, \ldots, O_{M_i}) = \frac{1}{\alpha} ln\Big(\sum_{j=1}^{M_i} e^{\alpha O_j}\Big), \tag{2.18}$$

where α is a real-valued parameter that controls the accuracy of the max function approximation.

To optimize the weights of the network, the authors derived update equations using the commonly used BackPropagation algorithm [64]. A multiple instance neural network graphical representation is shown in Figure 2.2.

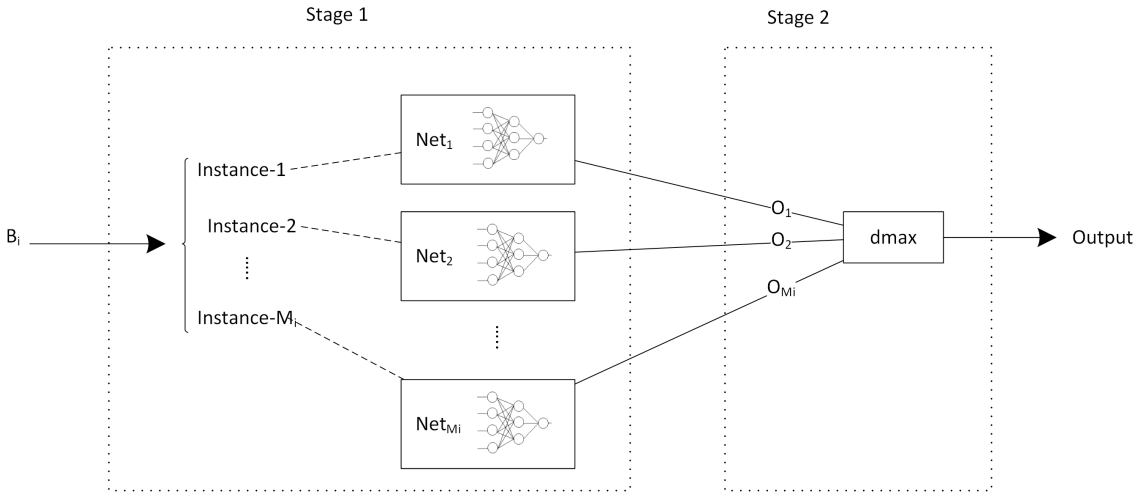


Figure 2.2: Illustration of a Ramon & Raedt's multiple instance neural network [1].

2.1.5 Multiple Instance RBF Neural Networks

Multiple Instance RBF (Radial Basis Function) Neural Networks (RBF-MIP) is an adaptation of the standard RBF neural network [65] for the problem of multiple instance learning. This approach was introduced by Zhou and Zhang [66]. Similarly to the standard RBF network, the RBF-MIP is composed of two layers. However, as opposed to the standard RBF neural networks where the first layer's nodes are prototype vectors indicating the centers of basis functions, the first layer of RBF-MIP corresponds to clusters of training bags, i.e., each input node of RBF-MIP is a cluster C_k, $k = 1, \ldots, K$, of training bags. The second layer of the RBF-MIP network is the same as the standard RBF neural network. A graphical representation of a typical RBF-MIP neural network is shown in Figure 2.3.

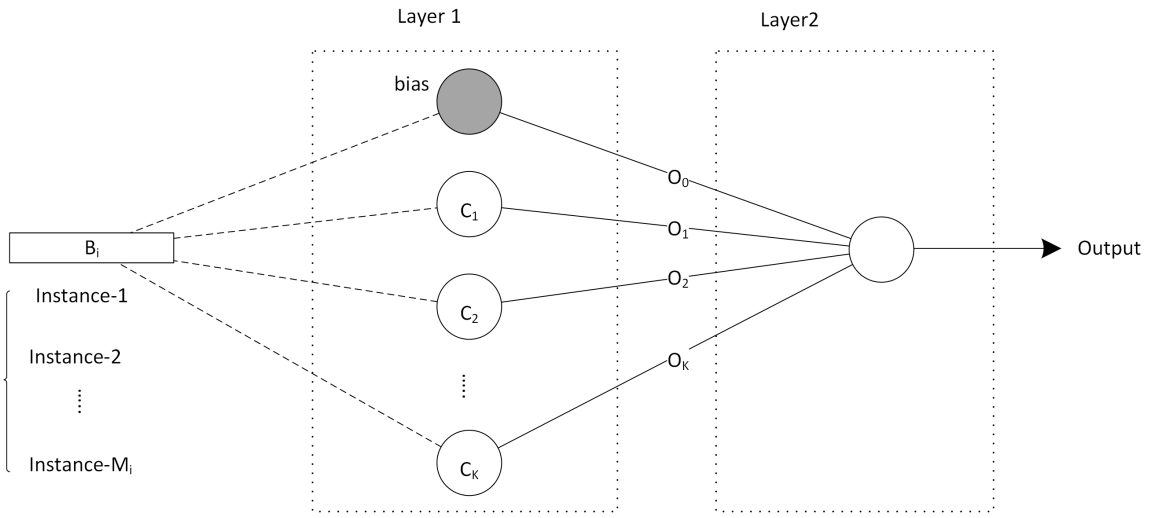


Figure 2.3: Illustration of an RBF-MIP neural network with a single output.

In the first layer of a given RBF-MIP network, the clustering of bags is achieved by merging training bags agglomeratively using the Hausdorff metric to measure distances between bags and between clusters [67]. Formally, given two bags B_1 and B_2 of instances $\{X_{1j}\}_{j=1}^{M_1}$ and $\{X_{2j}\}_{j=1}^{M_2}$, respectively, the Hausdorff metric between B_1 and B_2 is defined as

$$\mathbf{H}(B_1, B_2) = \min_{X_{1j} \in B_1, X_{2l} \in B_2} \big\{ dist(X_{1j}, X_{2l}) \big\}. \tag{2.19}$$

where $dist$ is a distance measure of the instance space (e.g. Euclidian distance). To compute the Hausdorff metric between a bag and a cluster of bags, first the instances from all bags in the cluster are merged into a new bag and (2.19) is used to compute the metric. The clustering process is summarized in Algorithm 2.2.

For a given input bag B_i, the first layer's outputs are computed as follows:

$$O_k = \begin{cases} exp\left(- \frac{\mathbf{H}(B_i, C_k)^2}{2\sigma_k^2} \right) & if\ k \neq 0 \\ 1 & if\ k = 0 \end{cases} \tag{2.20}$$

where σ_k is a standard deviation parameter whose value controls the smoothness of the kth input node function. σ_k is fixed to the same value σ that is the same for all input nodes

Algorithm 2.2 First Layer's Clustering Algorithm of RBF-MIP

Inputs: $\mathcal{B}$: the set of training bags.
N: the number of training bags.
K: number of remaining clusters in the first layer.
H: Hausdorff metric.
Outputs: $\{C_k\}_{k=1}^{K}$ clusters of training bags.

Begin with one cluster per training bag ($C_1 = \{B_1\}, \ldots, C_N = \{B_N\}$)
while there are more than K clusters **do**
Merge the two clusters C_i, C_j which minimize $\mathbf{H}(C_i, C_j)$
end while
return $\{C_k\}_{k=1}^{K}$.

and is computed by taking the average distance between every pair of clusters. i.e.,

$$\sigma = \mu \times \left(\frac{\sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \mathbf{H}(C_i, C_j)}{K(K-1)/2} \right). \tag{2.21}$$

In (2.21), μ is a scaling factor.

To optimize the weights of the second layer of the RBF-MIP neural network a sum-of-squared error loss function is minimized similarly to the standard RBF networks [65].

2.1.6 Citation K-Nearest Neighbors

In the standard K-NN classifier (K-Nearest Neighbors), to classify a given instance, "K" nearest instances are retrieved using a distance measure on the instance space (e.g. Euclidian distance), then an output label is computed from the labels of the "K" nearest instances. Using the same approach, Wang and Zucker [67] adapted K-NN for the case of multiple instances. To determine the nearest neighbors for a given bag, the Hausdorff metric (defined at (2.19)) is used instead of the Euclidian distance. Then the K-NN algorithm can be applied directly. Wang and Zucker found that the majority vote method, used by standard K-NN, often produced sub-optimal results in the multiple instance setting [68]. To improve the multiple instance K-NN, they proposed a variation called Citation-KNN [68]. Citation-KNN is motivated by the notion of citation from library and information science. Under this view the authors defined a "C-nearest citers" measure for a given bag. This measure is defined as following:

- For two given bags, B and B', let $Rank(B', B)$ equals n if B is the nth nearest neighbor of B'.

- Then, the C-nearest citers of B are the C bags that return the lowest neighbor ranking for B. i.e.,

$$Citers(B, C) = \{B_i \mid Rank(B_i, B) \leq C,\ B_i \in \mathcal{B}\}, \tag{2.22}$$

  where $\mathcal{B}$ is the set of all training bags.

The decision of Citation-KNN relies on the K-nearest bags as well as the C-nearest citers. Specifically, a bag is classified as positive if and only if there are strictly more positive bags than negative bags in the combined K-nearest bags and C-nearest citers. C is usually set to K+2.

2.2 Fuzzy Logic

Research on fuzzy set theory goes back to 1965 [69]. The first main development started with Zadeh [69] introducing fuzzy sets to extend classical set theory, and offering an intuitive approach to model and manipulate data with imprecision and uncertainty. Few years later, fuzzy logic was introduced by the same author [4]. Fuzzy logic is a computational framework that makes use of fuzzy set theory and fuzzy assignment of elements to sets. In classical set theory, also known as crisp sets, an element is either a member of a set or not. Whereas, in fuzzy set theory, an element is characterized by a degree of membership, usually a real number between 0 and 1. Fuzzy logic, in contrast to traditional two-valued (boolean) logic, uses the elements' membership degrees to evaluate the degree of truth of logical propositions. Hence, the degree of truth is non-crisp, or soft. This enables fuzzy logic to be characterized by linguistic terms rather than by numbers. For example, in fuzzy logic, a fuzzy proposition can have the following expression: " The temperature is high", in which the linguistic term "high" is a fuzzy set that describes the temperature. Fuzzy logic simulates human imprecise understanding of the world, and can be viewed as a framework for computing with words [46].

2.2.1 Fuzzy Sets

A fuzzy set expresses the degree to which an element belongs to a set. It has a characteristic function that describes the membership degree of an element in the set and takes values between 0 and 1.

Let X represent a collection of objects, referred to as the universe of discourse. Formally a fuzzy set A in X is defined as:

$$A = \{(x, \mu_A(x)) \mid x \in X\}, \tag{2.23}$$

where $\mu_A(x)$ is called the membership function (MF) for fuzzy set A. The MF maps every element of X to a membership degree, $\mu_A(x) \in [0, 1]$.

The difference between a crisp set and a fuzzy set, is that the MF is allowed to take any value in the interval $[0, 1]$ rather than $\{0, 1\}$. A simple interpretation of the degree of membership is given by:

- $\mu_A(x) = 1$ if x is totally in A
- $\mu_A(x) = 0$ if x is not in A
- $0 < \mu_A(x) < 1$ if x is partly in A

To clarify this definition, let us consider the subjective example of a person's age. Clearly, there is no crisp boundary beyond which a person can be considered "young" or not. If we model this statement by means of a crisp set, we need to use an expression of the following form:

$$Young = \{x \mid age(x) \leq 25,\ x \in X\}, \tag{2.24}$$

where X is the set of all people. An illustration of the crisp membership function of this example is shown in Figure 2.4. It should be clear that this crisp representation is not appropriate to model the concept of age. In fact, using this representation, a person who is 24.9 years old is considered young, while a person who is 25.1 years old is not young.

A fuzzy set representation, however, does not define any hard boundaries. Instead, it gradually assigns older people to the fuzzy set $Young$ in an ordered manner. It can be

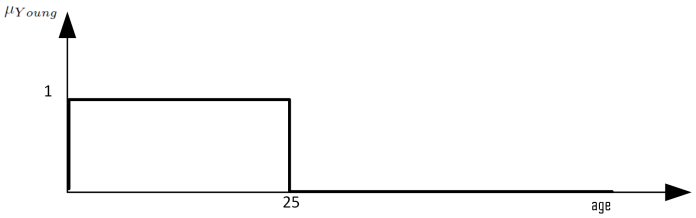


Figure 2.4: An illustration of the crisp membership function "Young"

described by:

$$Young = \{(x, \mu_{Young}(x)) \mid x \in X\}, \tag{2.25}$$

where μ_{Young} is the membership function of the fuzzy set $Young$, and is illustrated in Figure 2.5. In Figure 2.5, people of age between 0 and 25 are considered young, whereas people

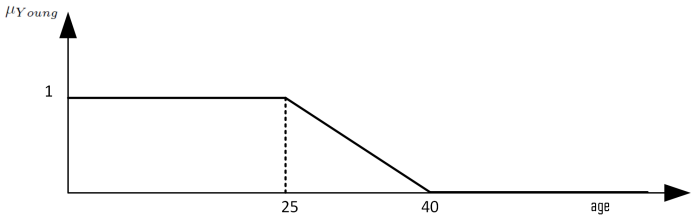


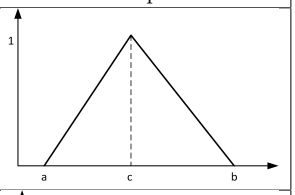
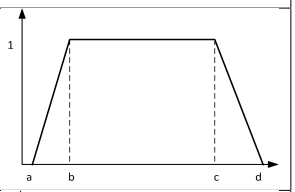
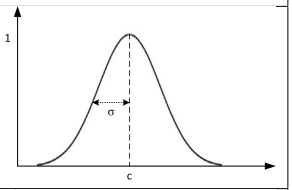
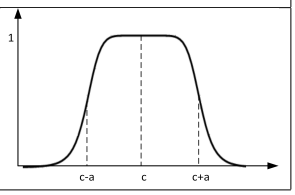
Figure 2.5: An illustration of the fuzzy membership function "Young"

older than 40 are not considered young. Between the ages of 25 and 40, the membership degree gradually decreases to 0. This representation is close to the way humans perceive the statement of a "person is young".

The construction of a fuzzy set depends on two main factors: the identification of a suitable universe of discourse, and the specification of an appropriate membership function [70]. In some applications, such as control, the fuzzy sets are typically designed by experts using domain knowledge. For other applications, such as pattern recognition, fuzzy sets can be learned from training data. In this case, the membership functions are parameterized functions and training data is used to learn the optimal set of parameters that best fit the data. Some of the common parameterized MFs include triangular MF, trapezoidal MF, Gaussian MF, and the generalized bell MF. The Shape and parameters of these MFs are shown in table 2.1.

TABLE 2.1

Shape and parameters of commonly used parameterized MFs

MF	Equation for $\mu(x)$	Parameters	Shape
Triangular MF	$max\Big(min\Big(\frac{x-a}{b-a}, \frac{c-x}{c-b}\Big), 0\Big)$	a, b, c	
Trapezoidal MF	$max\Big(min\Big(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\Big), 0\Big)$	a, b, d, c	
Gaussian MF	$exp\Big(-\frac{(x-c)^2}{2\sigma^2}\Big)$	c, σ	
Generalized bell MF	$\frac{1}{1+\left\lvert\frac{x-c}{b}\right\rvert^{2b}}$	a, b,c	

A fuzzy set is uniquely identified through its membership function. The α-cut of fuzzy set A, A^α is usually used to describe membership functions in more details. A^α is defined as:

$$A^\alpha = \{x \in X \; A(x) \geq \alpha\}, \tag{2.26}$$

where $\alpha \in [0, 1]$, A^1 is called the **core** of A, and A^0 is called the **support** of A. Figure 2.6 illustrates the α-cut, core, and support of a bell-shaped membership function.

As in classical crisp sets, the most basic operations for fuzzy sets are: union, intersection, and complement. In the following, let A and B be two fuzzy sets with membership functions $\mu_A(x)$ and $\mu_B(x)$.

- The **union** of two fuzzy sets A and B, often called "join", is a fuzzy set C characterized by a membership function μ_C defined as:

$$\mu_C(x) = \mu_{A \vee B}(x) = max(\mu_A(x), \mu_B(x)). \tag{2.27}$$

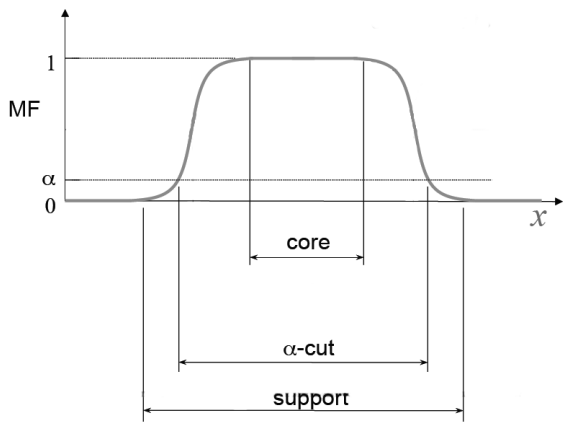


Figure 2.6: Illustration of α-cut, core, and support of a bell-shaped membership function

- The **intersection** of two fuzzy sets A and B, also known as "meet", is a fuzzy set C characterized by a membership function μ_C defined as:

$$\mu_C(x) = \mu_{A \wedge B}(x) = min(\mu_A(x), \mu_B(x)). \tag{2.28}$$

- The **complement** of fuzzy set A, denoted by $\neg A$ is defined as:

$$\mu_{\neg A}(x) = 1 - \mu_A(x). \tag{2.29}$$

The physical interpretation of the above fuzzy set operators relates to the linguistic concepts of OR, AND, and NOT. For instance, if the fuzzy sets A and B describe the $Youthfulness$ and $Tallness$ of a person respectively, then applying the set operators leads to the following statements:

- $\mu_{A \vee B}(x)$ = the degree to which x is **either** "Young" or "Tall";
- $\mu_{A \wedge B}(x)$ = the degree to which x is **both** "Young" and "Tall";
- $\mu_{\neg A}(x)$ = the degree to which x is **not** "Young".

We should emphasize here that, in addition to the definitions in (2.27), (2.28), and (2.29), there are multiple ways to define fuzzy union, intersection, and complement. Most of the

TABLE 2.2

Most frequently used T-norms and T-conorms operators

T-norms	
Minimum	$T_{min}(a,b) = min(a,b) = a \wedge b$
Algebraic product	$T_{ap}(a,b) = ab$
Bounded product	$T_{bp}(a,b) = 0 \vee (a+b-1)$
Drastic product	$T_{dp}(a,b) = \begin{cases} a & if\ b = 1 \\ b & if\ a = 1 \\ 0 & if\ a,b < 1 \end{cases}$
T-conorms	
Maximum	$Tc_{max}(a,b) = max(a,b) = a \vee b$
Algebraic sum	$Tc_{as}(a,b) = a + b - ab$
Bounded sum	$Tc_{bs}(a,b) = 1 \wedge (a+b)$
Drastic sum	$Tc_{ds}(a,b) = \begin{cases} a & if\ b = 0 \\ b & if\ a = 0 \\ 0 & if\ a,b > 0 \end{cases}$

operators, except complement, fall under two categories. The first one, called **T-norms**, is a class of fuzzy intersection operators [70] suitable to carry intersection, cartesian product, and as we will see later, fuzzy implication. The second category of operators, called **T-conorms**, is a class of fuzzy union operators [70] suitable to carry union and other aggregation operations.

The most frequently used T-norms operators include Minimum, Algebraic product, Bounded product, and Drastic product. Similarly, T-conorms operators include, Maximum, Algebraic sum, Bounded sum, and Drastic sum. These operators are defined in table 2.2.

In addition to modeling union, intersection, and complement, fuzzy set theory embeds mechanisms to model compensatory operations, i.e., aggregation operators where a high value in matching one criterion can compensate to some extent for a low value for another criterion. In the following we list five examples of such operators.

1. **Generalized mean:** Let $a_1, a_2, \ldots, a_n$ be the degrees of satisfaction of n criteria. The generalized mean is defined as:

$$h_\alpha(a_1, a_2, \ldots, a_n) = \Big(\frac{a_1^\alpha + a_2^\alpha + \ldots + a_n^\alpha}{n}\Big)^{\frac{1}{\alpha}}, \tag{2.30}$$

where α is a fixed real number. For $\alpha = 1$, h_α implements the arithmetic average.

Similarly, when $\alpha = -1$, h_α is the harmonic average, and as α approaches 0, the generalized mean converges to the geometric mean. All instantiations of the generalized mean produce values between the minimum and maximum of the degrees of satisfaction of the individual criteria.

2. **Fuzzy hybrid operators:** Fuzzy hybrid operators, combine different types of fuzzy set operators into a single equation.

   - Arithmetic hybrid operators

   $$A \bigoplus_{\gamma} B = (1-\gamma)(A \cap B) + \gamma(A \cup B), \tag{2.31}$$

   - Multiplicative hybrid operators

   $$A \bigotimes_{\gamma} B = (A \cap B)^{1-\gamma}(A \cup B)^{\gamma}. \tag{2.32}$$

   In (2.31) and (2.32), $\gamma \in [0,1]$ controls the amount of "mixing" of the union and intersection components.

3. **Zimmermann hybrid operator:** is a hybrid operator for multi-criteria aggregation that was modeled after the compensatory nature of human aggregation. For $a_1, a_2, \ldots, a_n$ degrees of satisfaction of n criteria, the Zimmermann hybrid operator is defined as:

   $$h_\gamma(a_1, a_2, \ldots, a_n) = \left(\prod_{i=1}^{n} (a_i)^{\delta_i} \right)^{1-\gamma} \left(1 - \prod_{i=1}^{n} (1-a_i)^{\delta_i} \right)^{\gamma}, \tag{2.33}$$

   where $\gamma \in [0,1]$ is a mixing coefficient, and δ_i are weights associated with each criterion a_i, such that $\sum_{i=1}^{n} \delta_i = n$.

4. **Ordered Weighted Averaging Operator (OWA) [71]:** Let $\{a_1, a_2 \ldots, a_n\}$ be n degrees of satisfaction of a given criteria, and $w = (w_1, w_2, \ldots, w_n)^T$ be a weight vector such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$. Let $a_{(j)}$ indicate the sorted a_j from largest degree of satisfaction to the minmum (i.e., $a_{(1)} = max\{a_1, a_2 \ldots, a_n\}$). The OWA operator is a mapping function, $OWA : R^n \to R$ defined as:

   $$OWA(a_1, a_2 \ldots, a_n) = \sum_{j=1}^{n} w_j a_{(j)}. \tag{2.34}$$

5. **Ordered Weighted Geometric Averaging Operator (OWGA) [72]:** OWGA is of similar form to OWA and is defined as:

$$OWGA(a_1, a_2 \ldots, a_n) = \prod_{j=1}^{n} w_j a_{(j)}. \tag{2.35}$$

2.2.2 Fuzzy Propositions

In fuzzy logic, a fuzzy proposition is defined as

$$p: \ X \ is \ A \tag{2.36}$$

Where X receives values x from a universal set U and A is a fuzzy set on U. For example, a proposition can be, "*temperature is high*", or "*patch is blue*". Each fuzzy proposition has a degree of truth $T(p)$ that is the membership degree of $X = x$ in A, denoted by $\mu_A(x)$.

2.2.3 Fuzzy If-Then Rules

A fuzzy if-then rule is expressed as

$$if \ x \ is \ A \ then \ y \ is \ B \tag{2.37}$$

where A and B are fuzzy sets on universes of discourse X and Y, respectively. The phrase "x *is* A" is often called **premise** (or, antecedent), and the phrase "y *is* B", is called **consequence**. Fuzzy rules can have multiple antecedents and multiple consequences connected with fuzzy operators. Examples of fuzzy if-then rules include:

- If a person is young then income level is low.
- If temperature is high then turn AC on.
- If time is day and sky is blue then weather is good. (notice the multiple antecedents)
- If pressure is high then volume is small and temperature is high. (notice the multiple consequences)

Interpreting a fuzzy if-then rule involves two main steps. First, the antecedent part of the rule is evaluated. This involves fuzzifying the input. The second step consists of applying

the results of the antecedent expression to the consequence using fuzzy implication [73]. In particular, as defined by Ramot et al. [74], a rule represents a fuzzy implication relation between unconditional fuzzy propositions p and q, where proposition p is the phrase "x *is* A", and q is the phrase "y *is* B". For instance, rule (2.37) combines the fuzzy propositions (p, q) into a logical implication denoted by $p \rightarrow q$, which is sometimes abbreviated as $A \rightarrow B$. The implication is in essence a fuzzy relation R between p and q on the product space $X \times Y$. Formally,

$$R = A \rightarrow B = A \times B = \int_{X \times Y} \mu_A(x) \star \mu_B(y)/(x,y)^1 \tag{2.38}$$

where $\star$ is a T-norm operator and $A \times B$ is used to represent the fuzzy relation R. R has a membership function denoted $\mu_{A \rightarrow B}(x,y)$ that represents the degree of truth of the implication $p \rightarrow q$ when $X = s$ and $Y = y$. In the literature, the most commonly used implication operators are the T-norms "min" and "algebraic product". In this case, (2.38) can be written as:

$$\mu_{A \rightarrow B}(x,y) = \int_{X \times Y} \mu_A(x) \wedge \mu_B(y)/(x,y) = min[\mu_A(x),\ \mu_B(y)] \tag{2.39}$$

or,

$$\mu_{A \rightarrow B}(x,y) = \int_{X \times Y} \mu_A(x) \cdot \mu_B(y)/(x,y) = \mu_A(x) \cdot \mu_B(y) \tag{2.40}$$

2.2.4 Fuzzy Reasoning

Fuzzy Reasoning is "an inference procedure that derives conclusions from a set of fuzzy if-then rules and known facts" [2, 70]. The inference is carried using the Generalized Modus Ponens rule [4, 70], which is given by the following scheme

premise	if x is A then y is B
fact	x is A'
consequence	y is B'

[1]The notation $\int_X \mu(x)/x$ stands for the union of membership grades, and "/" stands for a marker and does not imply division.

The premise part is a fuzzy rule as defined in (2.37), A and B are fuzzy sets on the universes of discourse X and Y. The fact is a fuzzy proposition and A' is in turn a fuzzy set on X. The consequence part B' can be derived using the compositional rule of inference introduced by Zadeh in 1973 [4]. B' is determined as a composition of the fact and the fuzzy implication operator. Specifically,

$$B' = A' \circ (A \rightarrow B) \tag{2.41}$$

or, equivalently,

$$\mu_{B'}(y) = max_x(min[\mu_{A'}(x), \mu_{A \rightarrow B}(x, y)]) \tag{2.42}$$

Using (2.39), (2.42) can be rewritten as,

$$\mu_{B'}(y) = max_x(min[\mu_{A'}(x), min[\mu_A(x), \mu_B(y)]) \tag{2.43}$$

Further simplification of (2.43) yields:

$$\mu_{B'}(y) = min(max_x(min[\mu_{A'}(x), \mu_A(x)]), \mu_B(y)) \tag{2.44}$$

The quantity $max_x(min[\mu_{A'}(x), \mu_A(x)])$ is known in the literture as *rule firing strength.*

To summarize, fuzzy reasoning involves the following 3 main steps:

1. Start by computing the proposition degree of truth, i.e. evaluate $min[\mu_{A'}(x), \mu_A(x)]$;
2. Compute the rule firing strength, or as pointed by Jang [70], the degree of belief for the antecedent part;
3. Compute the degree of belief of the consequent part by applying the "min" operator.

To better understand the fuzzy reasoning process, we analyze a simple generic example that is based on the following fuzzy if-then rule

$$if\ x\ is\ A\ then\ y\ is\ B \tag{2.45}$$

In (2.45), A and B are fuzzy sets on the universes of discourse X, and Y. Given the fact x is A', we want to evaluate rule (2.45) using fuzzy reasoning process defined by equation (2.43). This process is illustrated in Figure 2.7. First we compute the rule firing strength, $max_x(min(\mu_{A'_1}(x), \mu_A(x)))$. Then we infer the fuzzy set B' as B clipped by the rule firing strength.

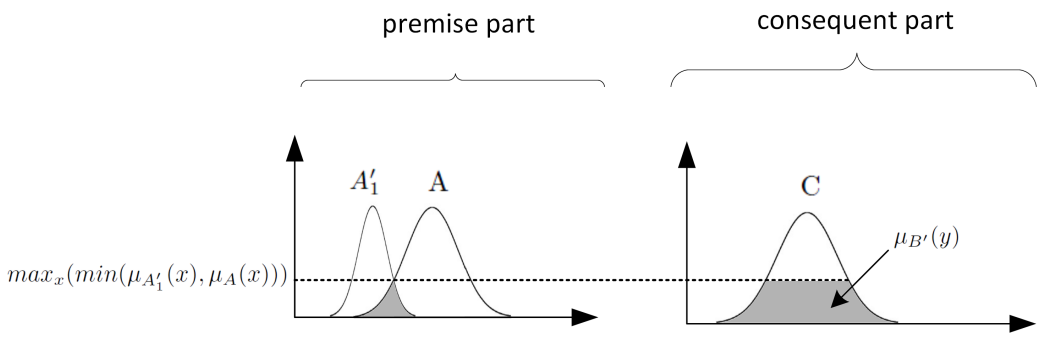


Figure 2.7: Illustration of the fuzzy reasoning process

2.3 Fuzzy Inference

Fuzzy inference is a powerful modeling framework that can handle computing with knowledge uncertainty and measurements imprecision effectively [2]. Fuzzy Inference is based on the concepts of fuzzy set theory, fuzzy if-then rules, and fuzzy reasoning. It performs a non-linear mapping from an input space to an output space by deriving conclusions from a set of fuzzy if-then rules and known facts [3]. Fuzzy Inference has been successfully applied to a wide range of problems, such as control [4–14], time series prediction [75], pattern recognition [20], and more recently classifier fusion [23]. Mamdani [48] and Sugeno [49] are the two commonly used fuzzy inference systems.

2.3.1 Mamdani Fuzzy Inference System

A Mamdani fuzzy inference system is an effective computing framework [48,76] based on fuzzy reasoning. This type of inference systems can be totally defined by means of a fuzzy rule base (FRB) composed of a union of if-then fuzzy rules.

For an input vector $\mathbf{x} = \{x_j \mid j = 1, \ldots, D\}$, a typical Mamdani-style fuzzy rule, $\mathcal{R}^i$, has the following form:

$$\mathcal{R}^i: \text{ If } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \ldots, \text{and } x_D \text{ is } A_D^i, \text{ then } o^i \text{ is } C^i. \tag{2.46}$$

In (2.46) $\mathcal{R}^i, i = 1, 2, \ldots, r$, is the ith fuzzy rule of the FRB, A_j^i is a fuzzy set associated with the jth input x_j, and C^i is the fuzzy set describing the output of the ith rule. These fuzzy sets consist of linguistic labels characterized by parameterized membership functions.

The FRB is the union of all rules, i.e.,

$$FRB = \bigcup_{i=1}^{r} \mathcal{R}^i. \tag{2.47}$$

Figure 2.8 is a graphical representation of a two-rule Mamdani fuzzy inference system and how it derives the output z when subject to a crisp input $\mathbf{x} = \{x_j | j = 1, \ldots, D\}$. The inference starts by fuzzification of $\mathbf{x}$. Fuzzifcation assigns a membership degree to each input dimension in the rules input fuzzy sets. As shown in in Figure 2.8, $\mathbf{x}$ activates the ith input fuzzy set of the jth rule by a degree of truth w_{ij} . Next, an implication process is executed resulting in the activation of the rules' output with different degrees. In this example, we use a simple min operator, and the output of rule $\mathcal{R}^j$ will be activated by a degree $w_j = min_{k=1,\ldots,D} w_{kj}$. Next, using a simple max operator, the 2 output fuzzy sets are aggregated to generate one output fuzzy set. Finally, the output set is defuzzified (e.g. using its centroid) to generate a final crisp output value.

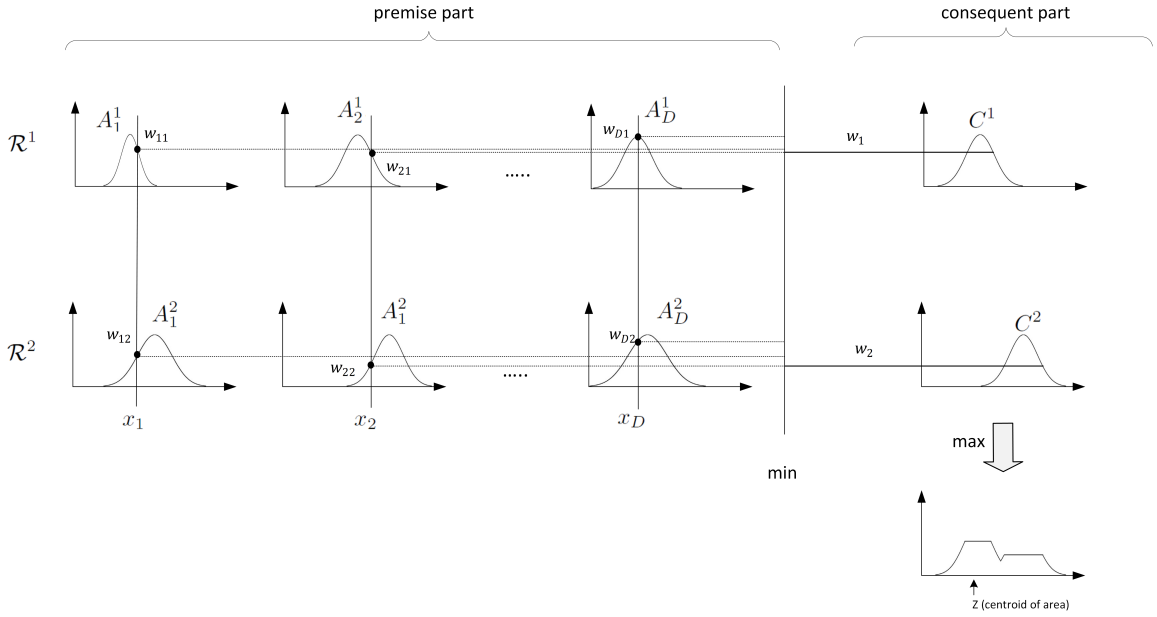


Figure 2.8: Illustration of Mamdani fuzzy inference with 2 rules and D inputs.

The system in Figure 2.8 implements a nonlinear mapping from its input space to an output space. Each fuzzy rule describes a local context in which the mapping is achieved. The input and output membership functions can be designed by leveraging expert knowledge (this practice is common in control problems), or can be learned directly from the data.

Specifically, labeled training data can be used to learn the FRB and the parameters of their membership functions. Typically, grid-based or clustering-based algorithms are used to partition the input space [70]. Each cluster will be represented by one fuzzy rule that describes a local context. Input membership functions will be generated based on the statistics of the input features within each context. Output membership functions can be generated by considering the distributions of labels within each context [23].

2.3.2 Sugeno Fuzzy Inference System

The Sugeno fuzzy model [49] was the first attempt at learning fuzzy rules from the training data. Similar to Mamdani system, the Sugeno fuzzy inference system is defined by means of a fuzzy rule base. However, unlike Mamdani rules, a Sugeno rule does not use fuzzy sets to describe the consequent part. Instead, it uses a crisp function $f()$ to compute the output. A typical Sugeno rule is defined as following

$$\mathcal{R}^i: \text{ If } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \ldots, \text{and } x_D \text{ is } A_D^i, \text{ then } o^i = f(x_1, x_2, \ldots, x_D). \tag{2.48}$$

where $\mathcal{R}^i, i = 1, 2, \ldots, r$, is the ith Sugeno fuzzy rule, A_j^i is a fuzzy set associated with the jth input x_j. Typically, $f()$ is polynomial in the input variables $x_1, \ldots, x_D$. In this case (2.48) can be rewritten as:

$$\mathcal{R}^i: \text{ If } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \ldots, \text{and } x_D \text{ is } A_D^i, \text{ then } o^i = b_0^i + \sum_{k=1}^{D} b_k^i \cdot x_k. \tag{2.49}$$

where $b_0^i, ..., b_D^i$ are the polynomial coefficients. When the polynomial coefficients $\mathbf{b^i}$ are first order, The Sugeno fuzzy model is called **first order**, and **zero order** when the polynomial coefficients are zero order.

The choice of a polynomial function makes the Sugeno method computationally effective and works well with optimization and adaptive techniques. This made Sugeno style inference very attractive in control problems, particularly for dynamic nonlinear systems [77].

Figure 2.9 illustrates the Sugeno fuzzy inference procedure with 2 rules. The premise part is evaluated as in the the Mamdani system. Every rule R^i is activated with a degree

w_i, firing strength. The output of every rule is a crisp value, o^1 and o^2, the overall output of the system is obtained by taking the weighted average of rules' outputs.

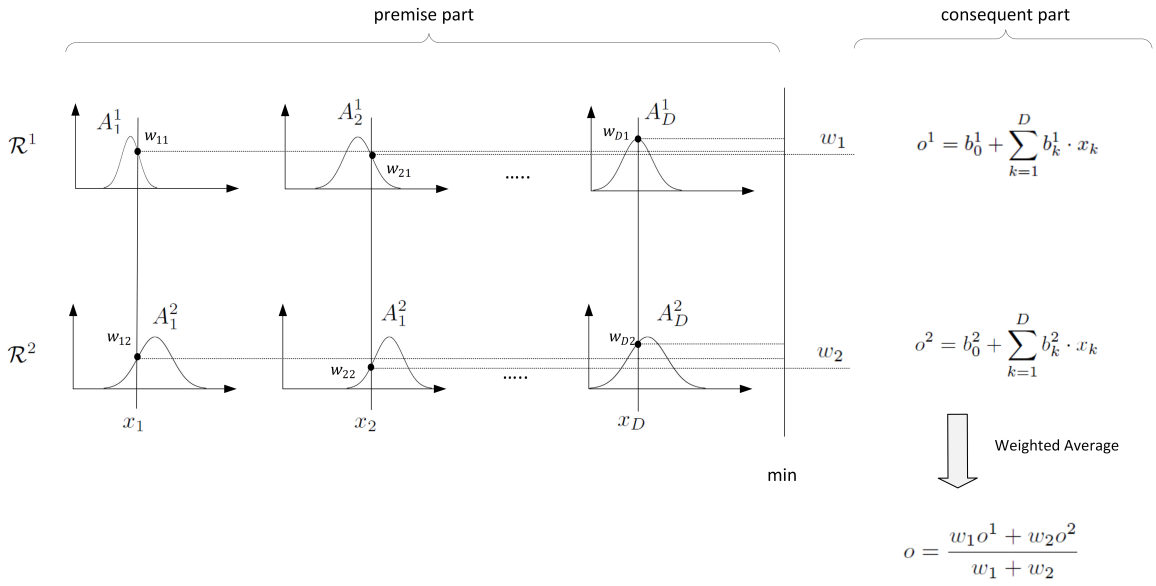


Figure 2.9: Illustration of Sugeno fuzzy inference with 2 rules and D inputs.

2.3.3 ANFIS: Adaptive Neuro-Fuzzy Inference System

The Adaptive Neuro-Fuzzy Inference System (ANFIS) [50] is a universal approximator that combines the learning and modeling power of neural networks and fuzzy logic into an adaptive inference system. Neural network deals with imprecise data by training, while fuzzy logic can deal with the uncertainty of human cognition. ANFIS offers an alternative to rule identification. The Mamdani and Sugeno fuzzy system identify rules based on intuition. ANFIS, in contrast, can jointly learn the optimal input space partition and the optimal output parameters through optimization. ANFIS is a hybrid intelligent system which implements a Sugeno fuzzy inference system and provides a systematic approach to generate fuzzy rules from a given input-output dataset. Typically, ANFIS is structred in a feedforward neural network that contains five layers. Figure 2.10 is a graphical representation of an ANFIS system with two Sugeno style rules and 2 inputs, given by

$$\begin{aligned} \mathcal{R}^1 &: \text{ If } x_1 \text{ is } M_1^1 \text{ and } x_2 \text{ is } M_1^2 \text{ then } f_1 = p_1 x_1 + q_1 x_2 + r_1. \\ \mathcal{R}^2 &: \text{ If } x_1 \text{ is } M_2^1 \text{ and } x_2 \text{ is } M_2^2 \text{ then } f_2 = p_2 x_1 + q_2 x_2 + r_2. \end{aligned} \tag{2.50}$$

where M_j^k is a fuzzy set associated with the jth input of rule k, and $\{p_k, q_k, r_k\}$ are the consequent parameters of the kth fuzzy rule. Nodes of same layers have similar functions.

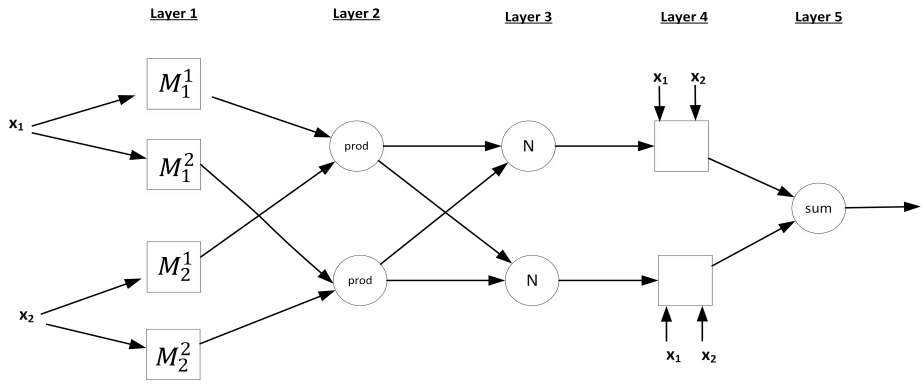


Figure 2.10: Architecture of an ANFIS system with two-input and two rules.

We denote the output of the ith node in layer l as $O_{l,i}$

Layer 1 known as the fuzzification layer, and is adaptive. It calculates the degree to which a given input satisfies a fuzzy set M. Every node evaluates the membership degree of an input in the fuzzy set M_j^k of membership function $\mu_{M_j^k}$. Generally, $\mu_{M_j^k}$ is a parameterized membership function (MF), for example Gaussian MF, where

$$\mu_{M_j^k}(x) = exp(\frac{-(x - c_{kj})^2}{2{\sigma_{kj}}^2}), \tag{2.51}$$

In (2.51) c_{kj} and σ_{kj} are the mean and variance of the Gaussian function, and are referred to as the **premise parameters**.

Layer 2 is a fixed layer where every node computes the firing strength of a rule. The output is the product of all incoming inputs.

$$O_{2,i} = w_i = \prod_{j=1}^{2} \mu_{M_j^i}(x_j), \tag{2.52}$$

Layer 3 is called "normalized firing strength". It calculates the ratio of a rule's firing strength to the sum of all rules' firing strengths.

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum_{j=1}^{r} w_j}, \tag{2.53}$$

where r is the number rules.

Layer 4 is an adaptive layer, it calculates each rule's output according to (2.50).

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i(p_i x_1 + q_i x_2 + r_i), \tag{2.54}$$

Layer 5 is a fixed layer, it computes the overall output which is the summation of all incoming signals.

$$O_{5,i} = \sum_i \overline{w}_i f_i = \sum_i \overline{w}_i(p_i x_1 + q_i x_2 + r_i), \tag{2.55}$$

In the following, we assume that we have N D dimensional training observations $\{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$ with desired output $\mathcal{T} = \{t_j | j = 1, \ldots, N\}$. ANFIS is devised to learn its parameters from training data. This process typically involves two step. 1) A structure identification and initialization step, and 2) a parameters optimization step. These two steps are described below.

1. **Model structure identification and initialization:** This step involves finding an optimal partition of the input space and initializing the fuzzy if-then rules. This task can be achieved using input space partitioning method as in the Mamdani FIS. However, unlike Mamdani inference, ANFIS optimizes the parameters of the fuzzy sets M_k^j. Thus, we need to use parameterized membership functions that are differentiable. Typically, Gaussian membership function is used. This MF can be completely determined by two scalar parameters (center c and width σ):

$$\mu_{M_j^k}(x) = exp(\frac{-(x - c_{kj})^2}{2{\sigma_{kj}}^2}), \tag{2.56}$$

   Thus, identifying the structure of the ANFIS network is equivalent to:

   (a) Partitioning the N D-dimensional training data into r clusters (i.e. rules). For this step, standard clustering algorithms such as the FCM algorithm [60] can be used.

   (b) Initializing the premise parameters:

$$\mathcal{P} = \left\{c_{ij}, \sigma_j^i \mid i = 1, 2, \cdots r; j = 1, \cdots, D\right\}$$

Typically, c_i is set to the center of the i^{th} cluster, and

$$\sigma_j^i = \frac{1}{N} \times \sum_{k=1,\ k \not\subseteq \mathbb{K}}^{N} \sqrt{\frac{-(x_{kj} - c_{ij})^2}{2 \times \log u_k^i}}, \mathbb{K} = \left\{ k | u_k^i = 1, k = 1, \ldots, N \right\}. \quad (2.57)$$

In (2.57) x_{kj} is the jth component of the kth observation, u_k^i indicates the membership of observation x_k in cluster i.

(c) Initializing the consequent parameters:

$$\mathcal{C} = \{p_i, q_i, r_i |\ i = 1, 2, \cdots r\}$$

Typically, a least squares estimator is used to initialize $\mathcal{C}^i$ as follows:

$$\mathcal{C}^i = (\mathcal{X}^T \mathcal{X})^{-1} \mathcal{X}^T \mathcal{T}. \quad (2.58)$$

where $\mathcal{X}$ is the $N \times (D+1)$ matrix of input training data right-padded with a column vector of all 1's. $\mathcal{T}$ is a column vector of the desired outputs. $\mathcal{X}^T$ is the matrix transpose of $\mathcal{X}$.

2. **Parameter Optimization:** Once the structure of the network is defined and initialized, an optimization and fine-tuning step of the system parameters is executed. the hybrid learning rule [50] based on alternating optimization to learn the optimal premise and consequent parameters. During the network forward pass, premise parameters are fixed and consequent parameters are updated using a least square estimator (LSE). Then, the consequent parameter are fixed and Gradient descent is used during back-propagation to optimize the premise parameters. These two steps are alternated until the network converges to a target training error or a maximum number of epochs is reached. A detailed description of the two main steps of the hybrid learning is provided below.

   - **BackPropagation Learning Rule:** In order to determine the update rule for premise parameters, first, for the pth training pattern, we compute a squared error measure commonly used in the backpropagation algorithm and defined as

$$E_p = (t_p - O_p)^2, \quad (2.59)$$

where t_p is the desired output, and O_p is the computed output of the network when presented with training sample p. Before we continue with the derivation, we want to point the reader's attention that during the backward pass, the consequents parameters are fixed and only the premise parameters are subject to optimization.

The overall error measure of the network is given by

$$E = \sum_{p=1}^{N} E_p. \tag{2.60}$$

To develop the gradient descent optimization on E, we compute the error rate for the pth training and for each node output $O_{l,i}$. This error rate $\varepsilon_{l,i}$ ($1 \leq l \leq 5$ indicates an ANFIS layer) is defined as follows

$$\varepsilon_{l,i} = \frac{\partial E_p}{\partial O_{l,i}},\ l = 1, \ldots, 4. \tag{2.61}$$

The error rate at the output node is given as following

$$\varepsilon_{5,1} = \frac{\partial E_p}{\partial O_{5,1}} = \frac{\partial E_p}{\partial O_p} = -2(t_p - O_p). \tag{2.62}$$

For non-output nodes (i.e. internal nodes, $l < 5$), we use the chain rule to derive the error rate

$$\varepsilon_{l,i} = \frac{\partial E_p}{\partial O_{l,i}} = \sum_{h=1}^{Card(l+1)} \frac{\partial E_p}{\partial O_{l+1,h}} \frac{\partial O_{l+1,h}}{\partial O_{l,i}}, \tag{2.63}$$

where $Card(l+1)$ refers the number of nodes at layer $l+1$.

Next, we need to minimize the network error with respect to the premise parameters $\{c_{kj}, \sigma_{kj} \mid 1 \leq k \leq r, 1 \leq j \leq D\}$. First, we compute the error rate with respect to a generic parameter θ using

$$\frac{\partial E_p}{\partial \theta} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \frac{\partial O^*}{\partial \theta}, \tag{2.64}$$

where S is the set of nodes whose outputs depend on θ.

Given (2.60), we have

$$\frac{\partial E}{\partial \theta} = \sum_{p=1}^{N} \frac{\partial E_p}{\partial \theta}. \tag{2.65}$$

The error rate for the premise parameters c_{kj} and σ_{kj} can be computed using

$$\frac{\partial E_p}{\partial c_{kj}} = \frac{\partial E_p}{\partial O_5}\frac{\partial O_5}{\partial O_4}\frac{\partial O_4}{\partial O_3}\frac{\partial O_3}{\partial O_2}\frac{\partial O_2}{\partial O_1}\frac{\partial O_1}{\partial c_{kj}}. \tag{2.66}$$

and,

$$\frac{\partial E_p}{\partial \sigma_{kj}} = \frac{\partial E_p}{\partial O_5}\frac{\partial O_5}{\partial O_4}\frac{\partial O_4}{\partial O_3}\frac{\partial O_3}{\partial O_2}\frac{\partial O_2}{\partial O_1}\frac{\partial O_1}{\partial \sigma_{kj}}. \tag{2.67}$$

From (2.62), we have

$$\frac{\partial E_p}{\partial O_5} = -2(t_p - O_p). \tag{2.68}$$

It is also straightforward to show that

$$\frac{\partial O_5}{\partial O_4} = \frac{\partial(\sum_{i=1}^{r} O_4)}{\partial O_4} = 1, \tag{2.69}$$

and,

$$\frac{\partial O_4}{\partial O_3} = \frac{\partial(f_i\overline{w}_i)}{\partial(\overline{w}_i)} = f_i = p_i x_1 + q_i x_2 + r_i. \tag{2.70}$$

Continuing the derivation, we have

$$\frac{\partial O_3}{\partial O_2} = \frac{\partial \overline{w}_i}{\partial w_i} = \frac{\partial\left(\frac{w_i}{\sum_{l=1}^{r} w_l}\right)}{\partial w_k} = \frac{\sum_{l=1}^{r} w_l - w_k}{\left(\sum_{l=1}^{r} w_l\right)^2}. \tag{2.71}$$

Next we compute the derivative from layer 2 to layer 1

$$\frac{\partial O_2}{\partial O_1} = \frac{\partial\left(\prod_{d=1}^{D} \mu_{M_d^i}(x_{pd})\right)}{\partial\left(\mu_{M_j^i}(x_{pj})\right)} = \prod_{d=1, d\neq j}^{D} \mu_{M_d^i(x_p d)}. \tag{2.72}$$

Finally, we have

$$\frac{\partial O_1}{\partial c_{kj}} = \frac{(x_{p,j} - c_{kj})}{\sigma_{kj}^2} \times exp(-\frac{(x_{p,j)} - c_{kj})^2}{2\sigma_{kj}^2}). \tag{2.73}$$

and,

$$\frac{\partial O_1}{\partial \sigma_{kj}} = \frac{(x_{p,j} - c_{kj})^2}{\sigma_{kj}^3} \times exp(-\frac{(x_{p,j)} - c_{kj})^2}{2\sigma_{kj}^2}). \tag{2.74}$$

Thus, the update equations for the parameters c_{kj} and σ_{kj} are given by

$$\triangle c_{kj} = -\eta\frac{\partial E}{\partial c_{kj}}, \tag{2.75}$$

and,

$$\triangle\sigma_{kj} = -\eta\frac{\partial E}{\partial\sigma_{kj}}, \tag{2.76}$$

where η is a learning rate determined in a similar manner to that of standard backpropagation algorithm [50].

Equations (2.75) and (2.76) can be used to update c_{kj} and σ_{kj} parameters either on-line, or in a batch mode. Next we develop the update rules for the consequents parameters.

- **LSE:** The Least Squares Estimator (LSE) is used to minimise the squared error $||AB-\mathcal{T}||^2$, where A has the outputs of Layer 3, and B has the set of consequent parameters subject of optimization. Initially, the parameters are identified using (2.58). Then, in the subsequent forward passes the consequent parameters are obtained using the pseudo-inverse of B, i.e.,

$$\hat{B} = (A^T A)^{-1} A^T \mathcal{T}, \tag{2.77}$$

In this type of LSE problems, it may happen that $(A^T A)$ is a singular matrix. To overcome this problem a recursive version of LSE can be used [70].

The derived update equations are used in an iterative algorithm that involves successive updates of the premise and consequent parameters. The ANFIS learning algorithm is summarized in Algorithm 2.3.

Algorithm 2.3 ANFIS Basic Learning Algorithm

Inputs: $\mathcal{X}$: the set of training pattern.
$\mathcal{T}$: the set of training labels.
η: the learning rate.
e: number of epochs.
Outputs: $\mathbf{b}_{ij}$: the sets of consequent parameters.
$\mathbf{c}_i j$: the set of membership functions' centers (premise parameters).
σ_{ij}: the set of membership functions' widths (premise parameters).

Initialize $\mathbf{b}_{ij}$ using (2.58),
Initialize $\mathbf{c}_{ij}$ using FCM.
Initialize σ_{ij} using (2.57).
repeat
Update $\mathbf{b}_{ij}$ using (2.77).
Update $\mathbf{c}_{ij}$ using (2.73).
Update σ_{ij} using (2.74).
until parameters do not change significatively or number of epochs is exceeded
return $\mathbf{b}_{ij}$, $\mathbf{c}_{ij}$, σ_{ij}

CHAPTER 3

MULTIPLE INSTANCE FUZZY LOGIC

In this chapter, we formalize Multiple Instance Fuzzy Logic (MIFL). MIFL is different from traditional fuzzy logic in that it allows for an additional dimension of ambiguity and it enables fuzzy reasoning with bags of instances instead of a single instance at a time. We introduce multiple instance variations of fuzzy propositions, fuzzy if-then rules, and fuzzy reasoning, which are the building blocks of our proposed framework. The following formulation is inspired by the work of Jang et al. [70] on traditional[1] fuzzy logic.

3.1 Multiple Instance Fuzzy Propositions

Recall that in traditional fuzzy logic, a fuzzy proposition can be written as

$$p: \ X \ is \ A \tag{3.1}$$

where X receives values x from a universal set U and A is a fuzzy set on U. For example, a proposition can be, "*temperature is high*". In traditional fuzzy logic, to evaluate the proposition p in (3.1), X is assigned a single value, say "$temperature = 90$", this will lead to "$p : temperature = 90$ *is high*". This will work in most cases even if X is a vector in $\mathbb{R}^n$. In fact, proposition (3.1) is valid as long as X is expressed by a single instance. However, for multiple instance (MI) data, the universe of discourse consists of bags of instances rather than single instances and the proposition needs to be generalized to a set of instances. Let B_i be a bag of M_i instances. The jth instance, $\mathbf{x}_{ij}$, is a D dimensional vector with elements

[1] In the remaining of this proposal previously presented fuzzy logic and fuzzy inference material will be referred to as "traditional".

x_{ijk} corresponding to features, i.e.,

$$B_i = \begin{pmatrix} x_{i11} & x_{i12} & \dots & x_{i1D} \\ x_{i21} & x_{i22} & \dots & x_{i2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{iM_i1} & x_{iM_i2} & \dots & x_{iM_iD} \end{pmatrix}. \tag{3.2}$$

Note that the number of instances can vary between bags (M_i depends on B_i). A bag is labeled positive if at least one of its instances is positive, and negative if all of its instances are negative.

Definition 3.1.1. *Let $\mathcal{B} = \{B_i | i = 1, \dots, N\}$ be the set of all bags. The universe of discourse U is the set of all bags of a given problem ($U = \mathcal{B}$). For a given instance $\mathbf{x}_{ij}$ of a given bag B_i, we define a "proposition instance" as:*

$$p_j : \mathbf{x}_{ij} \; is \; A, \tag{3.3}$$

Definition 3.1.2. *We define a multiple instance fuzzy proposition as the disjunction of proposition instances, i.e.,*

$$q : \; B_i \; is \; A \Longleftrightarrow q : \bigvee_{j=1}^{M_i} p_j \equiv \bigvee_{j=1}^{M_i} (\mathbf{x}_{ij} \; is \; A) \tag{3.4}$$

In (3.4) "$\bigvee$" is a T-conorm (maximum, algebraic sum, bounded sum, etc.), as defined in [78].

The proposition instance ("$\mathbf{x}_{ij}$ *is* A") in Definition 3.1.1 is evaluated as in (2.36), and represents the degree of truth of the proposition on a single instance. Not only the bag has different forms of expression (or instances), the proposition it self has different instances of truth. It follows that the degree of truth of a multiple instance fuzzy proposition is a combination of degrees of truth associated with the proposition instances. (3.4) is analogues to fuzzy information fusion [79, 80]. Fuzzy information fusion deals with merging uncertain observations that are possibly generated by heterogeneous sources. Thus, it is possible to

view the combination of degrees of truth of multiple instances as a a fuzzy information fusion process. In the following, we formalize our truth instances fusion process.

Let $\tilde{\mu}_A(B_i)$ denote the degree of truth of a multiple instance fuzzy proposition. $\tilde{\mu}_A(B_i)$ indicates the "membership degree" of B_i in A . The expression in (3.4) can be simplified further using the following theorem.

Theorem 3.1.3. *Let B be a collection of M instances drawn form an instance space X, and let A be a fuzzy set on X. The multiple instance proposition "q: B is A" is equivalent to the following*

$$q: \ B \text{ is } A \Rightarrow \exists \mathbf{x} \in X \mid \tilde{\mu}_A(B) = \mu_A(\mathbf{x}) \tag{3.5}$$

Note that $\mathbf{x}$ *is not necessary an instance of* B*.*

Proof. From (3.4), we have $\tilde{\mu}_A(B) = \bigvee_{j=1}^{M} \mu_A(\mathbf{x}_j)$, and we know that the T-conorm "$\bigvee$", aggregation operator, is closed in $[0, 1]$. Thus, the aggregation of a given set of membership grades remains in $[0, 1]$. It follows that $\tilde{\mu}_A(B) = \bigvee_{j=1}^{M} \mu_A(\mathbf{x}_j) \in [0, 1]$. Assuming that the fuzzy set A is normal and its membership function $\mu_A(x)$ is continuous, there exists $\mathbf{x} \in X$ such that

$$\bigvee_{j=1}^{M} \mu_A(\mathbf{x}_j) = \mu_A(\mathbf{x}) \tag{3.6}$$

Hence,

$$\bigvee_{j=1}^{M} \mu_A(\mathbf{x}_j) = \mu_A(\mathbf{x}) = \tilde{\mu}_A(B) \tag{3.7}$$

□

If the T-conorm is carried using a max operator, then $\tilde{\mu}_A(B_i)$ reduces to

$$\tilde{\mu}_A(B_i) = max\{\mu_A(\mathbf{x}_{ij}), j = 1 \ldots M_i\} \tag{3.8}$$

In (3.8), $\tilde{\mu}_A(B_i)$ is the highest degree of truth associated with the proposition's instances. This formulation is inline with the standard MIL assumption [36, 39], which states that a bag is positive if and only if one or more of its instances are positive. This relation will be covered in more details when we introduce multiple instance fuzzy inference in chapter 4.

If we consider the example of the image classification task described in Section 1.1. In this case, B_i is the image shown in Figure 1.1, and instances are the 12 patches (marked by black squares). In this case, an example of a multiple instance proposition can be

$$q: \textit{ image is blue} \Longleftrightarrow q: \bigvee_{j=1}^{12} \textit{patch}^j \textit{ is blue}. \tag{3.9}$$

3.2 Multiple Instance Fuzzy If-Then Rules

Recall that in traditional fuzzy logic a fuzzy if-then rule is expressed as

$$if\ x\ is\ A\ then\ y\ is\ B \tag{3.10}$$

where A and B are fuzzy sets on universes of discourse X and Y, respectively. As presented in chapter 2, rule (3.10) combines the fuzzy propositions (p, q) into a logical implication abbreviated as $A \rightarrow B$ with membership function $\mu_{A\rightarrow B}(x, y)$. The rule in (3.10) is defined using a premise part that is a single instance traditional fuzzy proposition. Thus, it is not suitable to carry implications on multiple instance problems. In the following, we introduce our approach to multiple instance implication that will lead to the development of multiple instance fuzzy if-then rules and multiple instance fuzzy reasoning.

3.2.1 Multiple Instance Fuzzy Implication

Definition 3.2.1. *Similarly to traditional fuzzy if-then rules, we define a multiple instance fuzzy rule as:*

$$if\ B_i\ is\ A\ then\ y\ is\ C \Longleftrightarrow if \bigvee_{j=1}^{M_i} (\mathbf{x}_{ij}\ is\ A)\ then\ y\ is\ C \tag{3.11}$$

where A and C are fuzzy sets on the universes of discourse X and Y, respectively. B_i is a bag of instances $\mathbf{x}_{ij}$, and M_i is the number of instances.

The premise part of a multiple instance fuzzy rule (i.e. $\bigvee_{j=1}^{M_i}(\mathbf{x}_{ij}\ is\ A)$) is a multiple instance proposition, whereas the consequence part is a traditional proposition. An example rule is given as follows,

$$\textbf{if } \textit{image } \textbf{is } \textit{blue } \textbf{then } \textit{class } \textbf{is } \textit{sky} \Longleftrightarrow \textbf{if} \bigvee_{j=1}^{12} (\textit{patch}^j \textbf{ is } \textit{blue}) \textbf{ then } \textit{class } \textbf{is } \textit{sky} \tag{3.12}$$

As before, the multiple instance rule combines the premise and consequence parts into a logical implication. However, since the premise part is a multiple instance proposition we will refer to this new logical implication as *multiple instance implication*. It is a fuzzy relation on the product space $\mathcal{B} \times C$ ($\mathcal{B}$: bags' space). Formally,

$$R = A \rightarrow C = A \times C = \int_{\mathcal{B} \times Y} \tilde{\mu}_A(B_i) \star \mu_C(y)/(B_i, y) \tag{3.13}$$

where $\star$ is a T-norm and $A \times C$ is used to represent the fuzzy relation R.

Lemma 3.2.2. *There exists a transformation that transforms a multiple instance fuzzy implication to a traditional fuzzy implication.*

Proof. Using theorem (3.1.3) we replace $\tilde{\mu}_A(B_i)$ by $\mu_A(x)$ and rewrite (3.13) as

$$R = A \rightarrow C = A \times C = \int_{X \times Y} \mu_A(x) \star \mu_C(y)/(x, y) \tag{3.14}$$

which is the expression of a traditional fuzzy relation. □

Thus, multiple instance fuzzy implication can be carried using traditional fuzzy implication. This result will be used when we develop multiple instance fuzzy reasoning in the next section.

In (3.13), R has a membership function denoted $\mu_{A \rightarrow C}(B_i, y)$ that represents the degree of truth of the implication when $\mathcal{B}$ is equal to B_i and Y is equal to y. Using *min* and *product* as implication operators, we have:

$$\mu_{A \rightarrow C}(B_i, y) = \int_{\mathcal{B} \times Y} \tilde{\mu}_A(B_i) \wedge \mu_B(y)/(x, y) = min[\tilde{\mu}_A(B_i),\ \mu_C(y)] \tag{3.15}$$

and,

$$\mu_{A \rightarrow C}(B_i, y) = \int_{\mathcal{B} \times Y} \tilde{\mu}_A(B_i) \cdot \mu_C(y)/(x, y) = \tilde{\mu}_A(B_i) \cdot \mu_C(y) \tag{3.16}$$

3.3 Multiple Instance Fuzzy Reasoning

Multiple instance fuzzy reasoning is needed when the universe of discourse U is a "bag-space" (i.e. $U = \mathcal{B}$), i.e., every element is a bag of instances rather than a single instance. In this case, we define the Generalized Modus Ponens as

premise	if B_i is A then y is $C \Longleftrightarrow$ if $\bigvee_{j=1}^{M_i}(X_{ij}$ is $A)$ then y is C
fact	$B_i = \{X_{ij}\}_{j=1}^{M_i}$ and X_{i1} *is* A'_1, X_{i2} *is* A'_2,...,X_{iM} *is* A'_{M_i}
consequence	y is C'

A and $\{A'_j\}_{i=j}^{M_i}$ are fuzzy sets on X (the instances space), and C is a fuzzy set on Y. Using the composition rule of inference, we determine C' using

$$C' = (\bigvee_{j=1}^{M_i} A'_j) \circ (A \to C) = \bigvee_{j=1}^{M_i} (A'_j \circ (A \to C)) \tag{3.17}$$

and we have,

$$\mu_{C'}(y) = \bigvee_{j=1}^{M_i} \{max_x(min[\mu_{A'_j}(x), \mu_{A\to C}(x,y)])\} \tag{3.18}$$

Using min as implication operator, (3.18) is equivalent to

$$\mu_{C'}(y) = \bigvee_{j=1}^{M_i} \{max_x(min[\mu_{A'_j}(x), min(\mu_A(x), \mu_C(y))])\} \tag{3.19}$$

further simplification yields

$$\mu_{C'}(y) = \bigvee_{j=1}^{M_i} \{min[max_x(min[\mu_{A'_j}(x), \mu_A(x)]), \mu_C(y)]\} \tag{3.20}$$

which is equivalent to

$$\mu_{C'}(y) = min\Big[\bigvee_{j=1}^{M_i} \{max_x(min[\mu_{A'_j}(x), \mu_A(x)])\}, \mu_C(y)\Big] \tag{3.21}$$

For instance, if the "max" aggregation operator is used, we have

$$\mu_{C'}(y) = min\Big[max\{max_x(min[\mu_{A'_j}(x), \mu_A(x)])\}_{j=1}^{M_i}, \mu_C(y)\Big] \tag{3.22}$$

The term "$max\{max_x(min[\mu_{A'_j}(x), \mu_A(x)])\}_{j=1}^{M_i}$" in (3.22) can be interpreted as the rule firing strength [70].

To summarize, the proposed multiple instance fuzzy reasoning involves the following 3 main steps:

1. Compute the multiple instance proposition degree of truth, i.e. evaluate $max\{\mu_{A'}(\mathbf{x}_{ij}), j = 1 \dots M_i\}$;
2. Compute the rule firing strength, or the degree of belief for the antecedent part;
3. Compute the degree of belief of the consequent part by applying the "min" operator.

3.4 Illustrative Example

Let B be a bag of three instances $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$. Let A'_1, A'_2, A'_3 be the fuzzy sets associated with the instances. Given this fact we want to evaluate the following multiple instance rule

$$if\ B\ is\ A\ then\ y\ is\ C \Longleftrightarrow if \bigvee_{j=1}^{3} (\mathbf{x}_i\ is\ A)\ then\ y\ is\ C \tag{3.23}$$

where A and C are fuzzy sets, defined as before. Figure 3.1 illustrates the proposed multiple instance inference process. To compute the rule firing strength we need to evaluate

$$\mu_{C'}(y) = min[max\{max_x(min[\mu_{A'_j}(x), \mu_A(x)])\}_{j=1}^{3}, \mu_C(y)] \tag{3.24}$$

First, we compute the truth instances (the shaded area in the premise part of Figure 3.1).

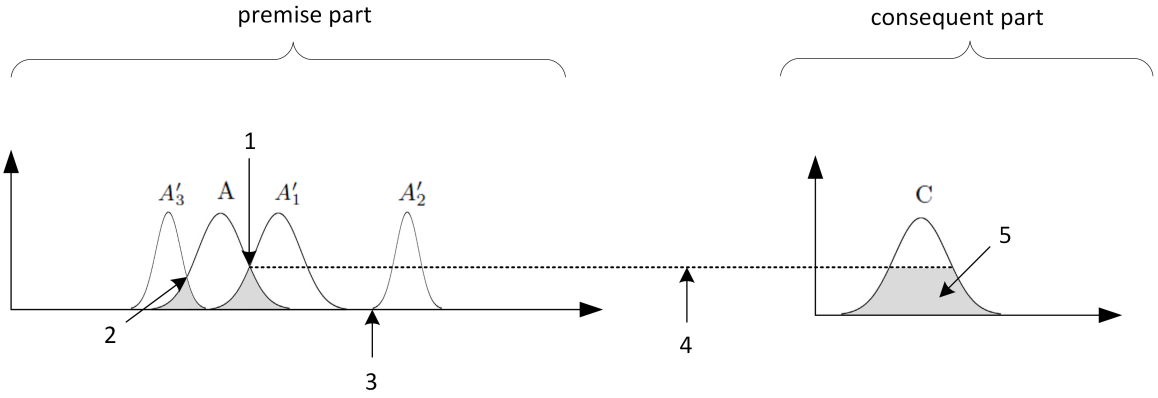


Figure 3.1: Illustration of the multiple instance inference process using the "max" aggregation operator. Legend: (1) $= max_x(min(\mu_{A'_1}(x), \mu_A(x)))$, (2) $= max_x(min(\mu_{A'_2}(x), \mu_A(x)))$, (3) $= max_x(min(\mu_{A'_3}(x), \mu_A(x)))$, (4) $= max\{max_x(min[\mu_{A'_j}(x), \mu_A(x)])\}_{j=1}^{3}$, (5) $= \mu_{C'}(y)$

Then all truth instances are aggregated using the "max" operator, i.e. we select the highest truth instance as the rule firing strength. Finally the membership function (MF), $\mu_{C'}(y)$, for the consequent part is computed as the MF of C clipped by the rule firing strength.

3.5 Discussion

Equation (3.21) defines fuzzy reasoning with bags of instances. To reach this goal, we have proposed multiple instance variations of fuzzy logic building blocks; i.e. propositions,

if-then rules, implications, and Generalized Modus Ponens. Our generalization was derived using a thoroughly and abstract mathematical formulation. The new findings will be used to build more advanced and complex fuzzy inference systems as will be shown in the remaining chapters. It is also worth noting that multiple instance fuzzy logic is a generalization of fuzzy logic, in fact if we set the number of instances in each bag to 1, all presented approaches will reduce to those of traditional fuzzy logic.

The difference between our multiple instance framework and fuzzy logic may seem subtle, but we think there is an important contribution to point out. In his short abstract published in 2008, titled "Is there a need for fuzzy logic?" [81], Zadeh wrote: "Fuzzy logic is not fuzzy. Basically, fuzzy logic is a precise logic of imprecision and approximate reasoning". We think that fuzzy logic is powerful at modeling knowledge uncertainty and measurements imprecision. More generally, it is one of the best frameworks to model vagueness. However, in addition to uncertainty and imprecision, there is a third vagueness concept that fuzzy logic does not address quiet well, yet. This vagueness concept is due to the ambiguity that arises when the data have multiple forms of expression, this is the case for multiple instance problems. Our framework deals with ambiguity by introducing the novel concept of truth instances: when carrying reasoning using multiple instance fuzzy logic, a proposition will not only have one degree of truth, it will have multiple degrees of truth, we call truth instances. Thus, effectively encoding the third vagueness component of ambiguity and increasing the expressive power of traditional fuzzy logic.

3.6 Related Work

Zadeh introduced fuzzy sets in 1965 [69] and fuzzy logic in 1973 [4]. After that, Mamdani and Sugeno followed with substantial additions [48, 49, 76]. Since then, many other developments and extensions to the fuzzy theory have been proposed. Most of the contributions can be classified under three categories: 1) contributions that propose variations and generalizations of fuzzy sets, 2) contributions that develop new fuzzy logic frameworks, and 3) contributions that propose new fuzzy inference schemes. For instance, Yager introduced

a new type of fuzzy sets known as fuzzy multisets (fuzzy bags) [82], Atanassov proposed intuitionistic fuzzy sets [83], and more recently Torra proposed hesitant fuzzy sets [84]. These approaches can be classified under the first category. Work that can be classified under the second category, include complex fuzzy logic [74, 85, 86] and complex fuzzy reasoning [87]. Under the third category, we can cite the contribution of Kaburlasos et al. [88] that consisted of an extension of fuzzy inference systems based on lattice theory.

To the best of our knowledge, there have been no proposed variations that aimed at reformulating fuzzy logic to support reasoning with multiple instances at the same time. The only previous work, that have a mention of fuzzy and MIL in the same framework, was presented by Mahnot et al. at [89]. They used fuzzy operators to compute diverse density [39]. This is by no means a reformulation of fuzzy logic to solve the multiple instance problem.

While there are no directly related approaches to our work, most methods have something in common as they aim to extend fuzzy logic and broader its domain of applicability. For instance, fuzzy multisets [82] may seem to be related to our approach because it utilizes bags of elements to represent objects. A fuzzy multiset can be defined as a fuzzy set where multiple occurrences of an element are permitted. Within our framework it can be used to represent the results of bags' fuzzification; i.e., the membership degrees of each instance in a given fuzzy set. Also aggregations operators proposed for fuzzy multisets [90] could be used in our proposed extension.

In fact, other proposed extensions of fuzzy sets could be adapted to the context of multiple instance fuzzy logic. For example, complex fuzzy sets [85] or complex fuzzy classes [91] are based on fuzzy sets characterized by complex-valued membership functions. Because of the two dimensionality nature of a complex fuzzy set, one can think of using it to carry reasoning with bags containing two instances at most. This later formulation is not necessary obvious and is worth investigating in future research projects.

3.7 Chapter Summary

In this chapter, we have introduced a new approach for multiple instance fuzzy logic. This approach extends traditional fuzzy logic to enable reasoning with bags rather than single instances. In particular, we have introduced multiple instance variations of fuzzy propositions, fuzzy implication, fuzzy if-then rules, and fuzzy reasoning. We have also discussed the novel concept of truth instances. In the next chapter, we will use the presented building blocks to derive new styles of fuzzy inference systems.

CHAPTER 4

MULTIPLE INSTANCE FUZZY INFERENCE

In this chapter, we introduce our approach to perform fuzzy inference with multiple instances. More specifically, we introduce two multiple instance fuzzy inference styles. The first one, the Multiple Instance Mamdani style fuzzy inference (MI-Mamdani), extends the traditional Mamdani style inference to account for multiple instances. The second one, the Multiple Instance Sugeno style fuzzy inference (MI-Sugeno), extends the Sugeno type inference.

4.1 Multiple Instance Mamdani Style Fuzzy Inference

The traditional Mamdani inference system outlined in chapter 2 is limited to reason with individual instances. First, the system's input is an individual instance. Second, the rules describe fuzzy regions within the instances's space. Third, the output of the system corresponds to the fuzzy inference using the D dimensions of a single instance. Fourth, labels of the individual instances are required to learn the parameters of the system.

In MIL, as outlined in Section 2.1, objects are described by bags of instances, and labels are available only at the bag level. Thus, the standard Mamdani style fuzzy inference system cannot be used within the MIL framework.

In the following, we propose a generalization of Mamdani fuzzy inference to extend it to reason with bags of instances. Similar to the traditional Mamdani system, we formulate the proposed multiple instance Mamdani system (MI-Mamdani) by means of multiple instance fuzzy if-then rules that can evaluate bags. As introduced in chapter 3, multiple instance

fuzzy rules can be expressed using:

$$\mathcal{R}^i(\mathbf{B}_p): \bigvee_{j=1}^{M_p} (\text{ If } x_{pj1} \text{ is } A_1^i \text{ and } x_{pj2} \text{ is } A_2^i, \ldots, \text{and } x_{pjD} \text{ is } A_D^i), \text{ then } o^i \text{ is } C^i. \tag{4.1}$$

where $\mathbf{B}_p$ is a bag of M_p instances as defined in (3.2). In (4.1), A_k^i is a fuzzy set associated with the kth instance feature, and "$\bigvee$" is a T-conorm. The output of the rule is described by the fuzzy set C^i.

In multiple instance fuzzy reasoning, the antecedent part,$\bigvee_{j=1}^{M_p}(\text{ If } x_{pj1} \text{ is } A_1^i \ldots, \text{and } x_{pjD} \text{ is } A_D^i)$, evaluates the degree to which the antecedent fuzzy sets describe each instance separately, then all responses are combined into a rule firing strength using a T-conorm. Using this inference style, the rule will be fired if and only if there exist at least one instance in the bag that can be described by means of the antecedent fuzzy sets.

The reason behind using a T-conorm for combining individual instances' responses, goes back to the standard MIL assumption [36, 39] which states that each instance has a hidden class label, and under this assumption, an example is positive if and only if one or more of its instances are positive. Thus, the bag-level class label is determined by the disjunction of the instance-level class labels. In the context of multiple instance inference, if a fuzzy rule describes a local region of the instances space that happens to be a positive MIL concept, and if the rule's output is high, the multiple instance fuzzy rule will be capable of classifying positive bags correctly. This is because at least one instance from each positive bag will activate the rule, leading to a high output (positive label). On the other hand, negative bags will not be able to significantly activate any rule.

Figure 4.1 illustrates the proposed MI-Mamdani system and its fuzzy inference mechanism to derive the output z in response to a bag of instances for the simple case of two rules. As it can be seen, the premise part of the rules evaluates all the bag's instances simultaneously. The inference starts by the fuzzification of instances x_{pm} of a given bag B_p. Fuzzification assigns a membership degree to each input instance dimension in the rules input fuzzy sets. In Figure 4.1, instance x_{pm} activates the ith input fuzzy set of the jth rule by a degree of truth w_{mij} . Next, an implication process is executed to combine the activations of the instances within the bag resulting in the activation of the rules' output with different

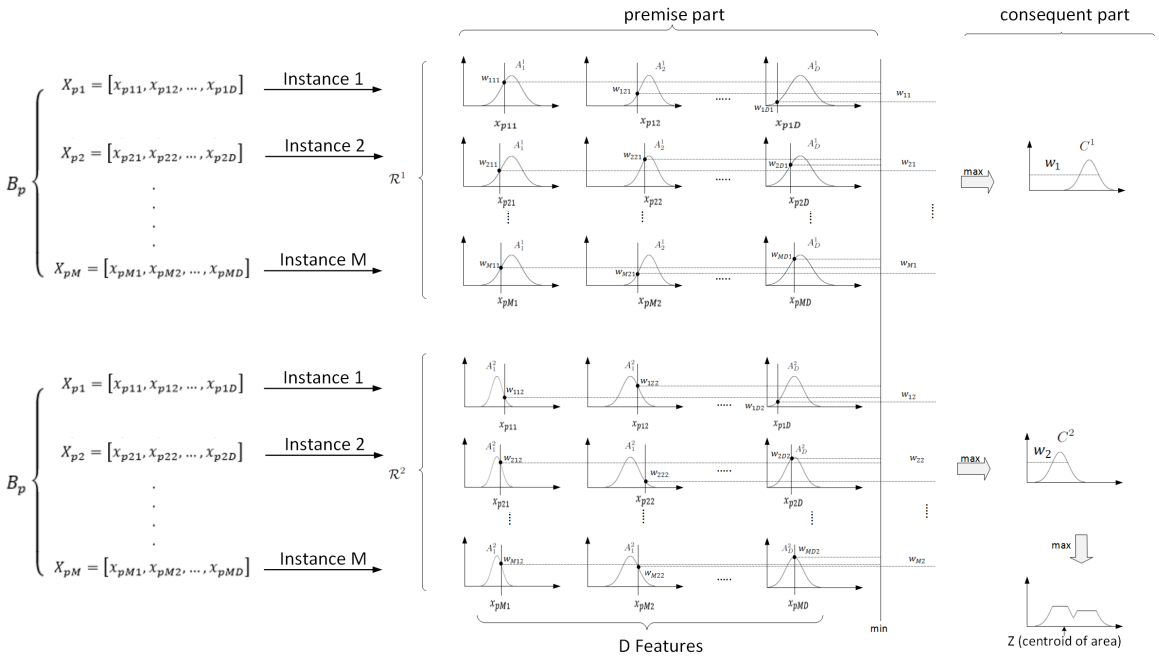


Figure 4.1: Illustration of the proposed multiple instance Mamdani fuzzy inference system.

degrees. In this example, we use a simple min operator, and the output of rule R^j will be partially activated by a degree $w_{mj} = min_{k=1,...,D} w_{mkj}$. The w_{mj} (truth instances) are combined in the premise part using the max T-conorm, resulting in the activation of rule R^j by a degree $w_j = max_{m=1,...,M} w_{mj}$. Next, using a simple max operator, the 2 output fuzzy sets are aggregated to generate one output fuzzy set. Finally, the output set is defuzziffied (e.g., using its centroid) to generate a final crisp output value.

The MI-Mamdani inference system allows the use of different T-conorms on different rules. The choice of the appropriate function should depend on the application and the purpose of the rule. More specifically: should the rule be activated if at least one instance of the bag is within the target concept? Or should it be activated only if at least a fixed subset of the instances are within the target concept?

Finally, we should note here that if we set M_p to 1 (i.e., constraint all bags to include only one instance), (4.1) reduces to a traditional fuzzy if-then rule commonly used in Mamdani FIS. Thus, the proposed MI-Mamdani fuzzy inference system, can be considered as a generalization of the traditional Mamdani system.

4.2 Multiple Instance Sugeno Style Fuzzy Inference

The rule in (2.49) is a traditional fuzzy if-then rule. As we showed in chapter 3, this type of rules is not suitable to solve multiple instance problems. To take advantage of the Sugeno inference and apply it to problems where objects are described by multiple instances, we propose the multiple instance Sugeno inference (MI-Sugeno) system.

Similar to the MI-Mamdani system introduced in Section 4.1, the MI-Sugeno system uses multiple instance fuzzy if-then rules where the consequent part is described by means of a function C that maps a bag of instances to a crisp numerical value. Specificaly, we define a multiple instance sugeno rule as:

$$\mathcal{R}^i(\mathbf{B}_p): \bigvee_{j=1}^{M_p} (\text{ If } x_{pj1} \text{ is } A_1^i, \ldots, \text{and } x_{pjD} \text{ is } A_D^i), \text{ then } o^i = C(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i) \quad (4.2)$$

In (4.2), $\mathbf{b}^i = b_0^i, ..., b_D^i$ is a set of polynomial coefficients. Similar to the traditional Sugeno fuzzy model, when the polynomial coefficients $\mathbf{b^i}$ are first order, our MI-Sugeno fuzzy model is called **first order**, and **zero order** when the polynomial coefficients are zero order.

The premise part of the rule is evaluated as in the MI-Mamdani case. To evaluate the consequent part, first the linear response of each instance is computed, i.e. $\mathbf{x}_{pj} \cdot \mathbf{b}^i$. Then a function C is used to compute the final output by combining the instances' response. Many functions could be used and the choice should be domain-specfic. For instance, the "max" function has been used in many applications.

The consequent part of the proposed MI-Sugeno style inference system is inspired by the work of Ray and Page on multiple instance regression [61]. In their work, the authors proposed a regression framework for predicting bags' labels. This formulation allows the linear coefficients $\mathbf{b}^i$ and the parameters of the combining function C to be learned using optimazation techniques.

Figure 4.2 illustrates the proposed MI-Sugeno system with 2 rules. The premise part of this system is equivalent to MI-Mandani (Figure 4.1). Its task is to evaluate each multiple instance rule firing strength. In the consequent part, the output of each rule, o^1 and o^2, are crisp values obtained as output of the combining function C. As in the traditional Sugeno fuzzy inference system, the overall output of the system is obtained by taking the weighted

average of the rules' outputs.

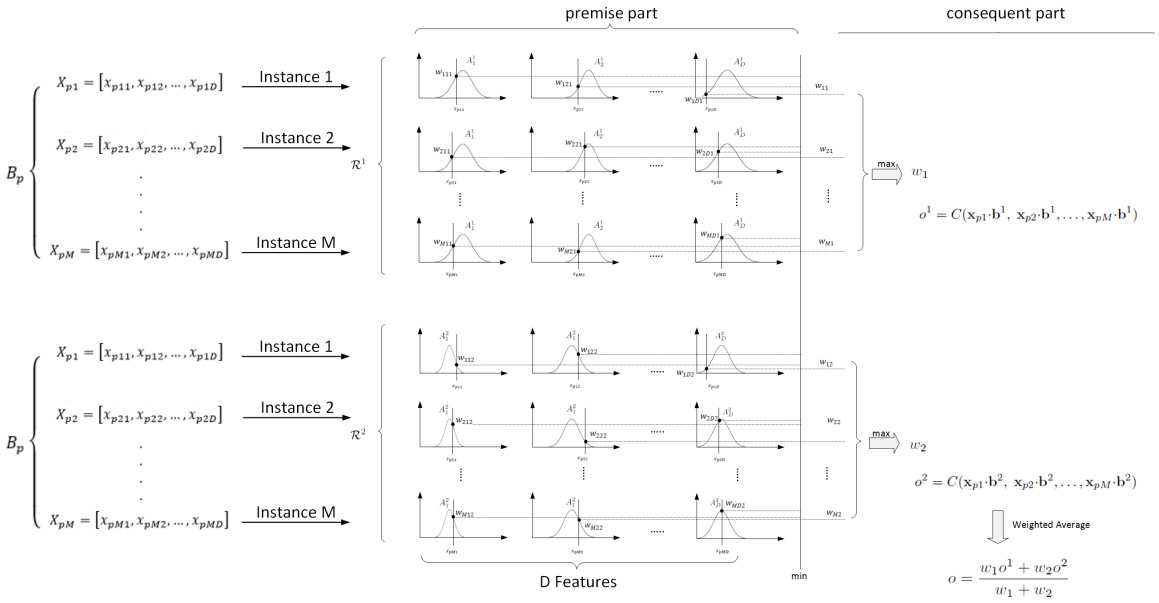


Figure 4.2: Illustration of the proposed multiple instance Sugeno fuzzy inference system

Similar to traditional fuzzy inference, the premise part of a multiple instance rule defines a local fuzzy region within the instance space, and the consequent part describes the characteristics of the system's output within that region. More specifically, in multiple instance learning (MIL) problems, a local region describes a positive concept, and the output of a rule represents the degree of "positivity" of the instances in that local region.

4.3 Learning the Structure and Parameters of Multiple Instance Fuzzy Inference Systems

The most important task in fuzzy inference with both MI-Mamdani and MI-Sugeno systems is the identification and learning of the system's structure and its parameters. Structure identification consist of identifying the number of multiple instance if-then fuzzy rules, identifying the membership functions (MFs) of the premise and consequent parts (i.e. Gaussian MFs or Trapezoidal MFs?), and also the T-conorms (min, max, product ...) involved in the multiple instance fuzzy reasoning. After structure, the parameters of the membership functions need to be learned. For example, for a Gaussian MF we need

to specify the mean and the standard deviation. In addition, in the case of an MI-Sugeno system we need to initialize the polynomial coefficients.

The system's structure and parameters identification rely mainly on determining the characteristics of the local regions within the instances' space that characterize positive bags. In traditional (i.e., single instance representation) fuzzy modeling, this task is achieved through input space partitioning, typically using grid partitioning and clustering [70]. In multiple instance inference systems, we need to identify regions that are defined by positive instances, referred to as positive concepts. Since in MIL, data is labeled at the bag level and not at the instance level, traditional space partitioning methods could not be used to learn the multiple instance fuzzy rules.

In the following, we describe our proposed approach to identify multiple instance fuzzy rules based on a fuzzy clustering algorithm of multiple instance (FCMI) data [59]. FCMI identifies target concepts that correspond to dense regions in the instance space that include as many positive instances as possible and as few negative instances as possible. In particular, we define the permise parts of the MI-FIS rules as local contexts within the input space (instances' space) that coincide with the identified target concepts.

Assume that we have N training bags, $\mathcal{B} = \{B_i | i = 1, \ldots, N\}$, and the set of their corresponding labels, $\mathcal{T} = \{t_i | i = 1, \ldots, N\}$. Let $\mathbf{T} = \{C_1, \ldots, C_r\}$, be r target concept points. Each target concept C_i is characterized by a center $c_i \in \mathbb{R}^D$ and a feature relevance scale vector $s_i \in \mathbb{R}^D$. The FCMI algorithm maximizes a fuzzy Multiple Concept Diverse Density (MDD) measure [59] defined as:

$$MDD(\mathbf{T}, \mathbf{U}) = \prod_{n=1}^{N} \prod_{i=1}^{r} \left[Pr(C_i | B_n) \right]^{u_{in}^m}. \tag{4.3}$$

In (4.8), $\mathbf{U} = [u_{in}]$ is a membership matrix such that each bag B_n is assigned to target concept C_i with membership degree u_{in}, and m is a fuzzifier that controls the fuzziness of the partitions as in the FCM [60]. $Pr(C_i|B_n)$ is the probability that C_i is a target concept given B_n, and defined as

$$Pr(C_i|B_n) = \begin{cases} 1 - \prod_{k=1}^{M} (1 - Pr(x_{nk} \in C_i)) & if\ lable(B_n) = 1, \\ \prod_{k=1}^{M} (1 - Pr(x_{nk} \in C_i)) & if\ lable(B_n) = 0 \end{cases} \tag{4.4}$$

where $label(B_n)$ is the label of bag and x_{nk} is the kth instance of bag B_n. $Pr(X_{nk} \in C_i)$ is regarded as the similarity of instance X_{nk} to target concept C_i, and its computed using

$$Pr(X_{nk} \in C_i) = e^{-(\sum_{j=1}^{D} s_{ij}(x_{nkj}-c_{ij})^2)} \tag{4.5}$$

In (4.5), s_{ij} is a scaling parameter that weighs the role of feature j in target concept i [39]. Let $\{C_i^{opt} = \{c_i^{opt}, s_i^{opt}\}\}_{i=1}^{r}$ be the optimal target concepts identified by FCMI that maximizes (4.8). Let $\mathbf{T} = \{C_1, \ldots, C_r\}$, be the r target concept points. For simplicity, we assume that the MFs of the r multiple instance rules are Gaussian MFs, with centers c_{ij}, $i = 1, \ldots, r$, and $j = 1, \ldots, D$. For a given multiple instance rule $\mathcal{R}^i$, the centers of the premise part's MFs are the centers of the target concepts, i.e.,

$$c_{ij} = C_{ij},\ for\ j = 1, \ldots, D. \tag{4.6}$$

The diverse density of each concept decreases gradually as we move away from C_i. Intuitively, the width σ_{ij} of a given concept C_i along dimension j can be set to the radius beyond which MDD is lower than a diverse density threshold τ_i. Formally, the standard deviations, $\{\sigma_{ij}\}$, can be computed as following:

$$\sigma_{ij} = \min_{Z \in \mathcal{I}} \big\{ |C_{ij} - Z_j| \ s.t.\ MDD_i(Z) < \tau_i \big\}, \tag{4.7}$$

In (4.7), $\mathcal{I}$ is the set of all instances, Z is a D dimensional vector and τ_i is constant, typically

$$\tau_i = \frac{1}{2} MDD_i(C_i) = \prod_{n=1}^{N} \big[Pr(C_i|B_n) \big]^{u_{in}^m}. \tag{4.8}$$

To identify the rules' consequent parts we can employ one of the following two strategies:

1. The consequents parts of multiple instance fuzzy rules are set to the singleton fuzzy set $\{1\}$. Using this strategy, positive bags that activate a rule, lead to rule's output of 1. This is inline with standard MIL assumption given that rules describe positive concepts.

2. Treat concepts as regular contexts. For each multiple instance fuzzy rule, its consequents fuzzy sets' parameters are identified by considering the ratio of positive to

negative instances within the context described by the multiple instance fuzzy rule. For example, if a context has 90% instances from positive bags, then a consequent MF can be set to a Gaussian with center equals to 0.9 and a predefined standard deviation ε.

4.3.1 Illustrative Example

To illustrate the proposed multiple instance fuzzy rules and the ability to learn from data without instance-level labels, we use a simple synthetic dataset. The data were generated from a distribution of two positive contexts, marked with squares in Figure 4.3. From each positive concept we generated 50 bags. We also generated 50 negative bags randomly from non concept regions. The number of instances within each bag is a random number between 2 and 10 instances. The data is shown in Figure 4.3. Instances from negative bags are shown as ".", and instances from positive bags are shown as "+" or "△" depending on the underlying concept. In Figure 4.3, we highlight one bag from Concept 1 by circling all of its instances. As it can be seen, one instance is close to a dense region of a positive concept while the other instances are scattered around. We note here that the centers of concepts in Figure 4.3 are shown just for the purpose of explanation and validation. We do not use this information as it is not available.

First, we run FCMI [59] to identify target concepts. These points are then used to identify the parameters of the fuzzy rules. Next, for the rules' consequents identification, we set the output MFs to the singleton fuzzy set {1}. This will ensure that bags that have instances within the positive concepts will get assigned high output. Finally, all the rules' parameters are used within an MI-Mamdani fuzzy inference system composed of two multiple instance fuzzy rules, each with two inputs and one output. A graphical representation of this system is shown in Figure 4.4. It can be seen that the centers of MFs identified using FCMI match the centers of positive concepts shown in Figure 4.3. To test the system, we generate 3 bags of instances: 2 positive bags and 1 negative bag. The instances of these bags are displayed in Figure 4.5. The multiple instance fuzzy inference using the MI-Mamdani system of

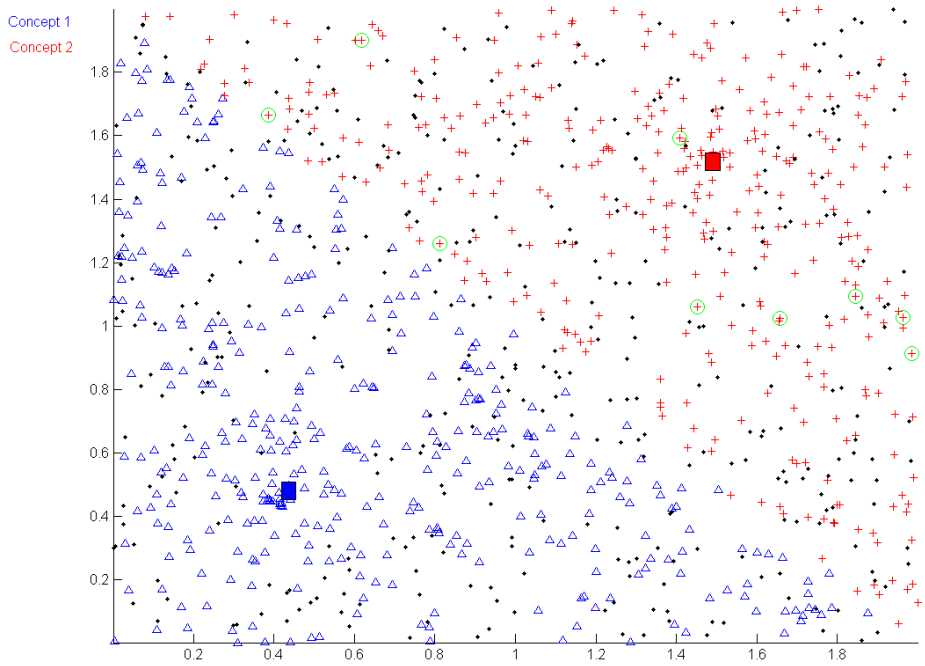


Figure 4.3: Instances from positive and negative bags drawn from data that have 2 concepts. Instances from negative bags are shown as ".", and instances from positive bags are shown as "+" or "△". Instances from one sample positive bag are circled.

the 3 test bags is summarized in Figure 4.6. The inference starts by fuzzification of all the instances at the same time, as illustrated in Figure 4.6a, then multiple instance fuzzy implication process is executed resulting in the activation of the rules' output with different degrees (each degree of activation is a firing strength as defined in Section 3.3). Next, using a simple max operator, the 2 output fuzzy sets are aggregated to generate one output fuzzy set. Finally, the output set is defuzzified using its centroid weighted by the maximum rule firing strength. The weighting ensures that negative bags that do not activate any rule will always have a low output.

In addition, we notice that while both first and second bags are positive, the inference process assigned a lower degree of belief to the second bag and as a consequence a lower output value. This will not impact classification's results as negative bags will not be able to activate any of the rules with a significant degree. But it will rather give applications an

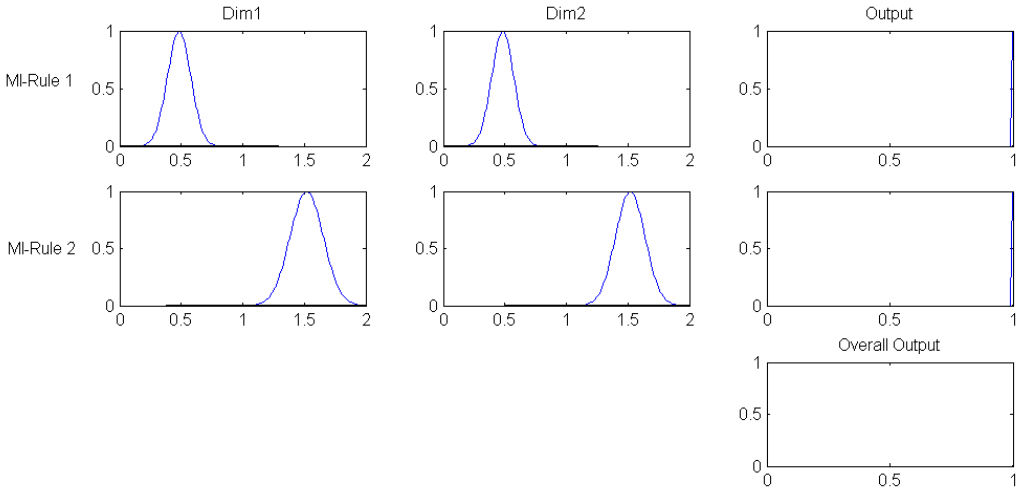


Figure 4.4: Illustration of MI-Mamdani fuzzy inference system learned using FCMI

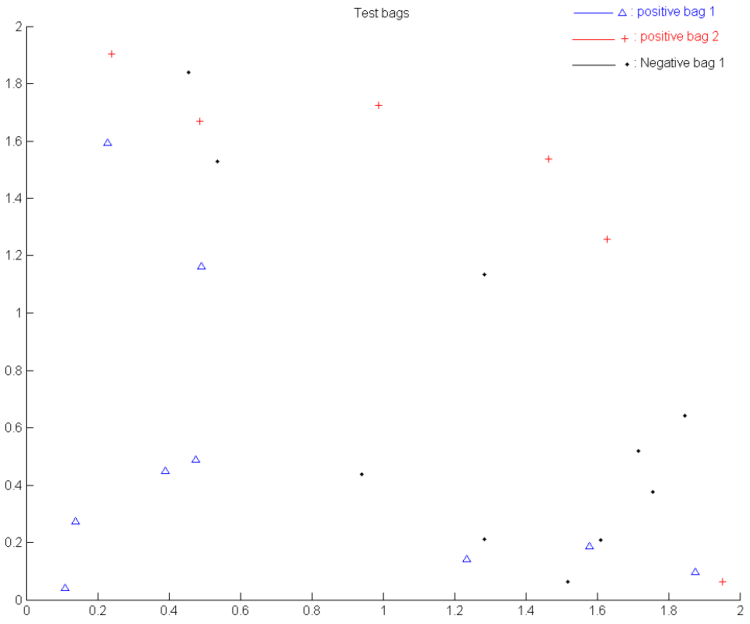
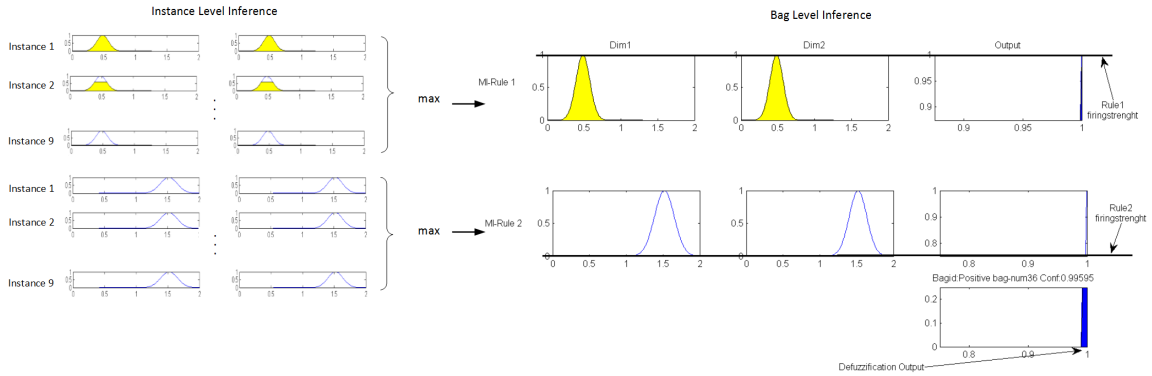
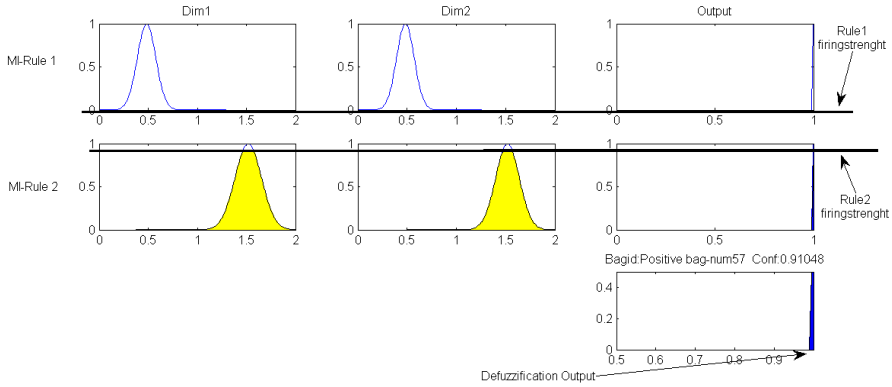


Figure 4.5: Instances from 2 positive and 1 negative bag.

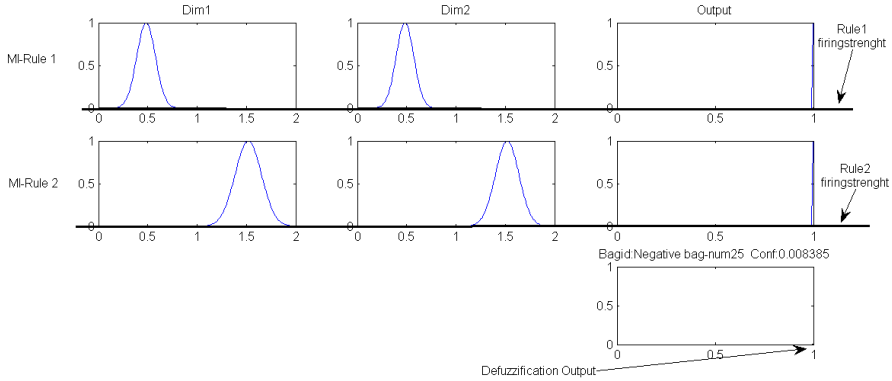
assessment about the confidence of the prediction.



(a) Inference process with the first positive bag .



(b) Inference process with the second positive bag.



(c) Inference process with the negative bag.

Figure 4.6: Multiple instance fuzzy inference using the learned MI-Mamdani system. The level of the activation indicates the membership degree of a bag in a given concept (i.e., rule). The system defuzzified output is the final confidence value.

4.4 Chapter Summary

In this chapter, we used our multiple instance fuzzy logic framework to: 1) derive a multiple instance Mamdani fuzzy inference style, and 2) derive a multiple instance Sugeno fuzzy inference style. We have also presented a method to learn multiple instance rules from data to solve MIL problems. The FCMI algorithm is used to extract concept points in the instances' space which are then transformed into multiple instance rules. This approach is essentially based on intuition. Although premise and consequent parameters of the MI-Mamdani and MI-Sugeno systems can be learned from data, the processes of identifying both set of parameters are independent. In the next chapter, we introduce a neuro-fuzzy architecture capable of learning from ambiguously labeled data without having to use FCMI to identify multiple instance rules, and can jointly learn the set of the optimal premise and consequent parameters using the backpropagation algorithm.

CHAPTER 5

MI-ANFIS: A MULTIPLE INSTANCE ADAPTIVE NEURO-FUZZY ARCHITECTURE

In this chapter, we introduce an adaptive neuro-fuzzy architecture based on the framework of multiple instance fuzzy logic, that is designed to handle reasoning with bags of instances as input and capable of learning from ambiguously labeled data. The new architecture called Multiple Instance-ANFIS (MI-ANFIS), is an extension of the standard Adaptive Neuro Fuzzy Inference System (ANFIS) [50].

In the following, we describe the architecture of the proposed MI-ANIFS and introduce a corresponding learning algorithm.

5.1 MI-ANFIS Architecture

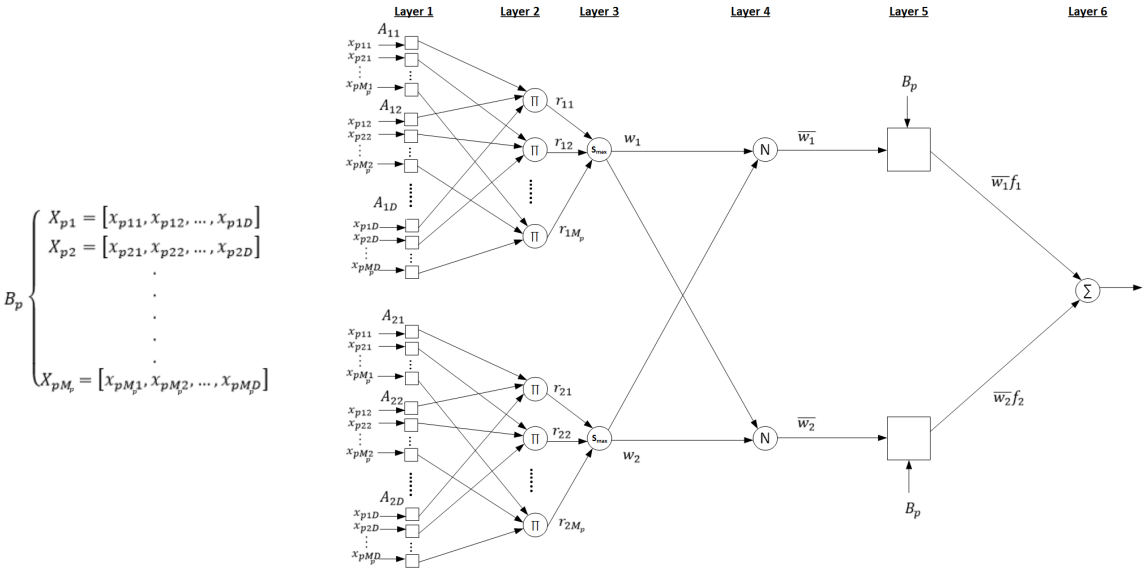


Figure 5.1: Architecture of the proposed multiple instance Adaptive Neuro-Fuzzy Inference System

Let B_p be a bag of M_p instances as defined in (3.2). For simplicity, we introduce our

MI-ANFIS for the case of two rules. The generalization to an arbitrary number of rules is trivial. The MI-ANFIS with two Sugeno rules can be described as:

$$\begin{aligned}\mathcal{R}^1(\mathbf{B}_p) &: \bigvee_{j=1}^{M_p} (\text{ If } x_{pj1} \text{ is } A_{11}, \ldots, \text{and } x_{pjD} \text{ is } A_{1D}), \text{ then } f_1 = C(\mathbf{x}_{p1} \cdot \mathbf{b}^1, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^1) \\ \mathcal{R}^2(\mathbf{B}_p) &: \bigvee_{j=1}^{M_p} (\text{ If } x_{pj1} \text{ is } A_{21}, \ldots, \text{and } x_{pjD} \text{ is } A_{2D}), \text{ then } f_2 = C(\mathbf{x}_{p1} \cdot \mathbf{b}^2, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^2)\end{aligned} \tag{5.1}$$

Figure 5.1 illustrates the proposed MI-ANFIS architecture. As in the traditional ANFIS, nodes at the same layer have similar functions. We denote the output of the ith node in layer l as $O_{l,i}$

Layer 1 is an adaptive layer, it calculates the degree to which a given input instance satisfies a quantifier A. Every node evaluates the membership degree of an input instance in the fuzzy set $A_{k,j}$ of membership function $\mu_{A_{k,j}}$. Generally, $\mu_{A_{k,j}}$ is a parameterized membership function (MF), for example a Gaussian MF with

$$\mu_{A_{k,j}}(x) = exp(\frac{-(x - c_{kj})^2}{2{\sigma_{kj}}^2}). \tag{5.2}$$

In (5.2), c_{kj} and σ_{kj} are the mean and variance of the gaussian function, and are referred to as the **premise parameters**.

Layer 2 is a fixed layer where every node computes the product of all incoming inputs. It evaluates the degree of truth of proposition instances, or simply, "truth instances". The output of this layer is

$$O_{2,i} = r_{\lceil i/M_p \rceil, i[M_p]} = \prod_{j=1}^{D} \mu_{A_{\lceil i/M_p \rceil, j}}(x_{p,i[M_p],j}), \tag{5.3}$$

where $\lceil \rceil$ is a ceiling operator, and $i[M_p]$ is $i \ mod \ M$. As in the traditional ANFIS, any T-norm can be used as the node function in this layer.

Layer 3 is a new addition when compared to the traditional ANFIS. Every node aggregates the truth instances of the previous layer by means of a smooth T-conorm. We use a smooth approximation of the "max" T-conorm known as the "softmax" function ($\mathcal{S}_\alpha$):

$$softmax_\alpha(x_1, x_2, \ldots, x_n) = \mathcal{S}_\alpha(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \frac{x_i \cdot e^{\alpha x_i}}{\sum_{j=1}^{n} e^{\alpha x_j}}. \tag{5.4}$$

In (5.4), α determines how closely softmax approximates the max operator. As α approaches ∞ , softmax's behavior approaches max. When $\alpha = 0$, it calculates the mean. As α approaches $-\infty$, softmax's behavior approaches the min operator. The outputs of this layer are the firing strength of the multiple instance fuzzy rules defined by layers 1 through layer 3. i.e.,

$$O_{3,i} = w_i = \mathcal{S}_\alpha(\{r_{i,j}\}_{j=1}^{M_p}). \tag{5.5}$$

Layer 3 is also a fixed layer.

Layer 4 is a fixed layer. Every node in this layer calculates the **normalized firing strength** of each rule, i.e.,

$$O_{4,i} = \overline{w}_i = \frac{w_i}{\sum_{j=1}^{|O_3|} w_j}, \tag{5.6}$$

where $|O_3|$ is the number of rules.

Layer 5 is an adaptive layer. Every node i in this layer computes the output of the i'th multiple instance rule using

$$O_{5,i} = C(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i), \tag{5.7}$$

The parameters $\{\mathbf{b}^i\}_{i=1}^{|O_3|}$ are referred to as the **consequent parameters**. The only constraint on C is that it has to be a smooth function to allow for optimization techniques to be applied. We use the "softmax" as the combining function for this layer. In this case, (5.7) is equivalent to:

$$O_{5,i} = \overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i), \tag{5.8}$$

note that the constant α here is not necessary the same as in Layer 3.

Optionally an activation function (such as Tanh or Sigmoid) could be applied to the individual linear responses (i.e., $x_{pj} \cdot \mathbf{b}^i$). This has advantage of protecting against the exploding gradient problem when using the backpropagation algorithm [92].

Layer 6 is a fixed layer with a single node labeled Σ. As in the traditional ANFIS, it computes the overall output of the system using

$$O_{6,1} = \sum_{i=1}^{|O_3|} O_{5,i} = \sum_{i=1}^{|O_3|} \overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i). \tag{5.9}$$

Algorithm 5.1 highlights MI-ANFIS frowrad pass.

Algorithm 5.1 MI-ANFIS Forward Pass Algorithm

Inputs: $\mathcal{B}$: the set of training bags.
$\mathcal{T}$: the set of training labels.
M: the number of instances in each bag.
R: the number of rules.
D: the dimensionality of instances.
α: the constant used in the "softmax" function.
Outputs: $O_{6,1}$: the output of the network.

for each instance **do**
  for j from 1 to D **do**
    for k from 1 to R **do**
      Fuzzification of inputs using (5.2).
    end for
  end for
end for
for each instance **do**
  for each rule **do**
    Evaluation of truth instances using (5.3).
  end for
end for
for each rule **do**
  Compute activation degree using (5.5).
  Computed normalized activation degree using (5.6).
end for
for each instance **do**
  for j from 1 to D **do**
    for k from 1 to R **do**
      Evaluate the output of $O_{5,k}$ using (5.7).
    end for
  end for
end for
Evaluate the overall output$O_{6,1}$ using (5.9).
return $O_{6,1}$

5.2 Basic Learning Algorithm

To identify the parameters of the proposed MI-ANFIS network, we propose a variation of the basic learning algorithm presented by Jang [50] (also outlined in chapter 2 of this dissertation). Our variation is different from the ANFIS standard backpropagation learning rule due to the additional layers and the use of "softmax" function. Thus, all update equations need to be rederived.

5.2.1 BackPropagation Learning Rule

In the following, we assume that we have N training bags, $\mathcal{B} = \{\mathbf{B}_p \mid p = 1, \ldots, N\}$, and their corresponding labels $\mathcal{T} = \{t_p \mid p = 1, \ldots, N\}$. After presenting the pth training bag, we compute its squared error measure commonly used in the backpropagation algorithm and defined as

$$E_p = (t_p - O_p)^2, \tag{5.10}$$

In (5.10), t_p is the desired bag output, and O_p is the computed output of the network when presented with training bag B_p. Equation (5.10) demonstrates the need for MI-ANFIS. In fact, due to the absence of instances' labels, errors can be computed only at the bag level. Errors at the instance level cannot be computed and are not needed as we will show later. The overall error measure of the network after presenting all N bags is

$$E = \sum_{p=1}^{N} E_p. \tag{5.11}$$

To develop the gradient descent optimization on E, we compute the error rate for the pth training bag and for each node output $O_{l,i}$. This error rate $\varepsilon_{l,i}$ ($1 \leq l \leq 6$ indicates an MI-ANFIS layer) is defined as

$$\varepsilon_{l,i} = \frac{\partial E_p}{\partial O_{l,i}}. \tag{5.12}$$

The error rate at the output node is given as following

$$\varepsilon_{6,1} = \frac{\partial E_p}{\partial O_{6,1}} = \frac{\partial E_p}{\partial O_p} = -2(t_p - O_p). \tag{5.13}$$

For non-output nodes (i.e. internal nodes, $l < 6$), we derive the error rate using the chain rule

$$\varepsilon_{l,i} = \frac{\partial E_p}{\partial O_{l,i}} = \sum_{h=1}^{Card(l+1)} \frac{\partial E_p}{\partial O_{l+1,h}} \frac{\partial O_{l+1,h}}{\partial O_{l,i}}, \tag{5.14}$$

where $Card(l+1)$ refers the number of nodes at layer $l+1$.

Next, we seek to minimize the network error with respect to the premise parameters $\{c_{kj}, \sigma_{kj} \mid 1 \leq k \leq |O_3|, 1 \leq j \leq D\}$, and with respect to consequents parameters $\{\mathbf{b}^i\}_{i=1}^{|O_3|}$. The error rate with respect to a generic parameter θ can be computed using

$$\frac{\partial E_p}{\partial \theta} = \sum_{O^* \in G} \frac{\partial E_p}{\partial O^*} \frac{\partial O^*}{\partial \theta}, \tag{5.15}$$

where G is the set of nodes whose outputs depend on θ.

Using(5.11), the total error rate is given by

$$\frac{\partial E}{\partial \theta} = \sum_{p=1}^{N} \frac{\partial E_p}{\partial \theta}. \tag{5.16}$$

5.2.1.1 Update Rule For Premise Parameters

First we compute the error rate for the premise parameters c_{kj} and σ_{kj} using

$$\frac{\partial E_p}{\partial c_{kj}} = \sum_{i=1}^{M_p} \frac{\partial E_p}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} \frac{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}{\partial c_{kj}}. \tag{5.17}$$

and,

$$\frac{\partial E_p}{\partial \sigma_{kj}} = \sum_{i=1}^{M_p} \frac{\partial E_p}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} \frac{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}{\partial \sigma_{kj}}. \tag{5.18}$$

Using the chain rule defined in (5.14), we have

$$\begin{aligned} \frac{\partial E_p}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} = \frac{\partial E_p}{\partial O_{6,1}} \times \frac{\partial O_{6,1}}{\partial O_{5,k}} \times \frac{\partial O_{5,k}}{\partial O_{4,k}} \times \frac{\partial O_{4,k}}{\partial O_{3,k}} \times \frac{\partial O_{3,k}}{\partial O_{(2,i+(k-1)M_p)}} \\ \times \frac{\partial O_{(2,i+(k-1)M_p)}}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}. \end{aligned} \tag{5.19}$$

Hence, (5.17) is equivalent to

$$\begin{aligned} \frac{\partial E_p}{\partial c_{kj}} = \frac{\partial E_p}{\partial O_{6,1}} \times \frac{\partial O_{6,1}}{\partial O_{5,k}} \times \frac{\partial O_{5,k}}{\partial O_{4,k}} \times \frac{\partial O_{4,k}}{\partial O_{3,k}} \\ \times \sum_{i=1}^{M_p} \frac{\partial O_{3,k}}{\partial O_{(2,i+(k-1)M_p)}} \times \frac{\partial O_{(2,i+(k-1)M_p)}}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} \times \frac{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}{\partial c_{kj}}. \end{aligned} \tag{5.20}$$

From (5.13), we have

$$\frac{\partial E_p}{\partial O_{6,1}} = -2(t_p - O_p). \tag{5.21}$$

It is also straightforward to show that,

$$\frac{\partial O_{6,1}}{\partial O_{5,k}} = \frac{\partial(\sum_{i=1}^{|O_3|} O_{5,i})}{\partial O_{5,k}} = 1. \tag{5.22}$$

and,

$$\frac{\partial O_{5,k}}{\partial O_{4,k}} = \frac{\partial(\overline{w}_k \mathcal{S}_\alpha(\mathbf{x}_{p1}\cdot\mathbf{b}^k,\ \mathbf{x}_{p2}\cdot\mathbf{b}^k,\ldots,\mathbf{x}_{pM_p}\cdot\mathbf{b}^k))}{\partial(\overline{w}_k)} = \mathcal{S}_\alpha(\mathbf{x}_{p1}\cdot\mathbf{b}^k,\ \mathbf{x}_{p2}\cdot\mathbf{b}^k,\ldots,\mathbf{x}_{pM_p}\cdot\mathbf{b}^k). \tag{5.23}$$

Continuing with the derivation

$$\frac{\partial O_{4,k}}{\partial O_{3,k}} = \frac{\partial \overline{w}_k}{\partial w_k} = \frac{\partial\left(\frac{w_i}{\sum_{l=1}^{|O_3|} w_l}\right)}{\partial w_k} = \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\left(\sum_{l=1}^{|O_3|} w_l\right)^2}. \tag{5.24}$$

and,

$$\frac{\partial O_{3,k}}{\partial O_{(2,i+(k-1)M_p)}} = \frac{\partial \mathcal{S}_\alpha(\{r_{k,j}\}_{j=1}^{M_p})}{\partial r_{k,(i+(k-1)M_p)}} = \frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}}\left[1+\alpha\left(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p})\right)\right]. \tag{5.25}$$

The details of the derivation of the "softmax" function details can be found at [39].

Next, we need to compute $\frac{\partial O_{(2,i+(k-1)M_p)}}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}$

$$O_{(2,i+(k-1)M_p)} = \prod_{d=1}^{D} \mu_{A_{\left(\lceil (i+(k-1)M_p)/M_p \rceil, d\right)}}\left(x_{p,(i+(k-1)M_p)[M_p],d}\right). \tag{5.26}$$

and,

$$O_{(1,i+[(k-1)D+(j-1)]M_p)} = \mu_{A_{k,j}}(x_{p,(i+(k-1)M_p)[M_p],d}). \tag{5.27}$$

Thus,

$$\begin{aligned}\frac{\partial O_{(2,i+(k-1)M_p)}}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} &= \frac{\partial\left(\prod_{d=1}^{D} \mu_{A_{\left(\lceil (i+(k-1)M_p)/M_p \rceil, d\right)}}\left(x_{p,(i+(k-1)M_p)[M_p],d}\right)\right)}{\partial\left(\mu_{A_{k,j}}(x_{p,(i+(k-1)M_p)[M_p],d})\right)} \\ &= \prod_{d=1, d\neq j}^{D} \mu_{A_{\left(\lceil (i+(k-1)M_p)/M_p \rceil, d\right)}}\left(x_{p,(i+(k-1)M_p)[M_p],d}\right).\end{aligned} \tag{5.28}$$

Finally, we have

$$\frac{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}{\partial c_{kj}} = \frac{\partial \mu_{A_{k,j}}(x_{p,(i+(k-1)M_p)[M_p],j})}{\partial c_{kj}} = \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})}{\sigma_{kj}^2} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2}). \tag{5.29}$$

Substituting the derivatives in (5.20), we obtain

$$\begin{aligned}\frac{\partial E_p}{\partial c_{kj}} = & -2(t_p - O_p) \times \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^k,\ \mathbf{x}_{p2} \cdot \mathbf{b}^k, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^k) \times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\Big(\sum_{l=1}^{|O_3|} w_l\Big)^2} \\ & \times \sum_{i=1}^{M_p} \left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}} \Big[1 + \alpha\Big(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p})\Big)\Big] \right. \\ & \times \prod_{d=1,d\neq j}^{D} \mu_A\Big(\big\lceil (i+(k-1)M_p)/M_p \big\rceil, d\Big)\Big(x_{p,(i+(k-1)M_p)[M_p],d}\Big) \\ & \left. \times \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})}{\sigma_{kj}^2} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2}) \right).\end{aligned} \tag{5.30}$$

As in the standard ANFIS, an update formula for the parameter c_{kj} is given by

$$\triangle c_{kj} = -\eta \frac{\partial E}{\partial c_{kj}}, \tag{5.31}$$

where η is a learning rate determined in a similar manner to that of the standard backpropagation algorithm [50].

The update formula for σ_{kj} can be derived in a similar manner. It can be shown that

$$\frac{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}{\partial \sigma_{kj}} = \frac{\partial \mu_{A_{k,j}}(x_{p,(i+(k-1)M_p)[M_p],j})}{\partial \sigma_{kj}} = \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{\sigma_{kj}^3} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2}). \tag{5.32}$$

Using (5.18), we obtain

$$\begin{aligned}\frac{\partial E_p}{\partial \sigma_{kj}} &= -2(t_p - O_p) \times \mathcal{S}_\alpha(\mathbf{x}_{p1}\cdot\mathbf{b}^k,\ \mathbf{x}_{p2}\cdot\mathbf{b}^k, \ldots, \mathbf{x}_{pM_p}\cdot\mathbf{b}^k) \times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\Big(\sum_{l=1}^{|O_3|} w_l\Big)^2} \\ &\times \sum_{i=1}^{M_p}\left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}}\Big[1+\alpha\Big(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p})\Big)\Big]\right. \\ &\times \prod_{d=1,d\neq j}^{D} \mu_A\Big(\lceil (i+(k-1)M_p)/M_p \rceil, d\Big)\Big(x_{p,(i+(k-1)M_p)[M_p],d}\Big) \\ &\times \left.\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{\sigma_{kj}^3} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2})\right).\end{aligned} \tag{5.33}$$

And the update formula for σ_{kj} is as follows

$$\triangle\sigma_{kj} = -\eta\frac{\partial E}{\partial \sigma_{kj}}, \tag{5.34}$$

where η is the same learning rate as in (5.31)

5.2.1.2 Update Rule For Consequent Parameters

The error rate for the consequent parameters $\{\mathbf{b}^i = \{b_0^i, ..., b_D^i\}, i = 1\ldots|O_3|\}$ is defined as

$$\frac{\partial E_p}{\partial \mathbf{b}^i} = (\frac{\partial E_p}{\partial b_0^i}, \frac{\partial E_p}{\partial b_1^i}, \ldots, \frac{\partial E_p}{\partial b_D^i}). \tag{5.35}$$

where,

$$\frac{\partial E_p}{\partial b_j^i} = \frac{\partial E_p}{\partial O_{(5,i)}}\frac{\partial O_{(5,i)}}{\partial b_j^i},\ for\ \ j = 1, \ldots, D. \tag{5.36}$$

Next, we compute $\frac{\partial E_p}{\partial O_{(5,i)}}$ using the previously defined chain rule in (5.14), and obtain

$$\frac{\partial E_p}{\partial O_{(5,i)}} = \frac{\partial E_p}{\partial O_{6,1}} \times \frac{\partial O_{6,1}}{\partial O_{5,i}}. \tag{5.37}$$

From (5.13), we have

$$\frac{\partial E_p}{\partial O_{6,1}} = -2(t_p - O_p). \tag{5.38}$$

And from (5.39), we have

$$\frac{\partial O_{6,1}}{\partial O_{5,i}} = 1. \tag{5.39}$$

Continuing with the derivation, we have

$$\begin{aligned}
\frac{\partial O_{(5,i)}}{\partial b_j^i} &= \frac{\partial(\overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1}\cdot\mathbf{b}^i,\ \mathbf{x}_{p2}\cdot\mathbf{b}^i,\ldots,\mathbf{x}_{pM_p}\cdot\mathbf{b}^i))}{\partial b_j^i} \\
&= \frac{\partial}{\partial b_j^i}\Big(\overline{w}_i \sum_{m=1}^{M_p} \frac{\mathbf{x}_{pm}\cdot\mathbf{b}^i exp(\alpha \mathbf{x}_{pm}\cdot\mathbf{b}^i)}{\sum_{h=1}^{M_p} exp(\alpha \mathbf{x}_{ph}\cdot\mathbf{b}^i)}\Big) = \overline{w}_i \sum_{m=1}^{M_p} \frac{\partial}{\partial b_j^i}\Big(\frac{\mathbf{x}_{pm}\cdot\mathbf{b}^i exp(\alpha \mathbf{x}_{pm}\cdot\mathbf{b}^i)}{\sum_{h=1}^{M_p} exp(\alpha \mathbf{x}_{ph}\cdot\mathbf{b}^i)}\Big) \\
&= \overline{w}_i \sum_{m=1}^{M_p} \frac{1}{\Big(\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i))\Big)^2}\Big[\Big(x_{pmj}\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i)\Big) \\
&- \Big(\mathbf{x}_{pm}\cdot\mathbf{b}^i \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i)\alpha(x_{phj} - x_{pmj})\Big)\Big].
\end{aligned} \tag{5.40}$$

Thus, the overall error rate with respect to the consequent parameter b_j^i is given according to (5.16) as follows

$$\begin{aligned}
\frac{\partial E}{\partial b_j^i} &= \sum_{p=1}^{N} \frac{\partial E_p}{\partial b_j^i} \\
&= \sum_{p=1}^{N} -2(t_p - O_p) \\
&\times \overline{w}_i \sum_{m=1}^{M_p} \frac{1}{\Big(\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i))\Big)^2}\Big[\Big(x_{pmj}\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i)\Big) \\
&- \Big(\mathbf{x}_{pm}\cdot\mathbf{b}^i \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph}\cdot\mathbf{b}^i - \mathbf{x}_{pm}\cdot\mathbf{b}^i)\alpha(x_{phj} - x_{pmj})\Big)\Big].
\end{aligned} \tag{5.41}$$

Hence, the update formula for consequent parameter b_j^i

$$\triangle b_j^i = -\eta \frac{\partial E}{\partial b_j^i}, \tag{5.42}$$

where η is the same learning rate as in (5.31)

Equations (5.31), (5.34) and (5.42) can be used to update c_{kj}, σ_{kj} and b_j^i parameters either on-line, bag by bag (we want to emphasis here that the on-line learning is not achieved instance by instance, but rather bag by bag), or off-line in batch mode after presentation of the entire data set.

The proposed MI-ANFIS basic learning algorithm is summarized in Algorithm 5.2.

Algorithm 5.2 MI-ANFIS Basic Learning Algorithm

Inputs: $\mathcal{B}$: the set of training bags.
$\mathcal{T}$: the set of training labels.
M: the number of instances in each bag.
α: the constant used in the "softmax" function.
η: the learning rate.
E_{max}: number of epochs.
ϵ: minimum parameters change value.
Outputs: $\mathbf{b}^i$: the sets of consequent parameters.
$\mathbf{c}^i$: the set of membership functions' centers.
σ^i: the set of membership functions' widths.

Initialize $\mathbf{b}^i$, $\mathbf{c}^i$, and σ^i.
repeat
Update $\mathbf{b}^i$ using (5.42) and $b^{i(new)} = b^{i(old)} + \triangle b^i$.
Update $\mathbf{c}^i$ using (5.31) and $c^{i(new)} = c^{i(old)} + \triangle c^i$.
Update σ^i using (5.34) and $\sigma^{i(new)} = \sigma^{i(old)} + \triangle \sigma^i$.
until $max(\|b^{i(new)} - b^{i(old)}\|, \|c^{i(new)} - c^{i(old)}\|, \|\sigma^{i(new)} - \sigma^{i(old)}\|) < \epsilon$ *or* $Number\ of\ epochs > E_{max}$
return $\mathbf{b}^i$, $\mathbf{c}^i$, σ^i

5.3 Illustrative Example

After deriving the necessary learning algorithms, we now study an example to show the potential of using MI-ANIFS to learn concepts from ambiguously labeled data. For this purpose, let us consider the syntectic dataset used in Section 4.3.1. For illustrative purposes, we only update the premise parameters during the training epochs, and show that the MI-ANFIS Basic Learning Algorithm (Algorithm 5.2) is capable of identifying positive concepts as well as their corresponding multiple instance fuzzy rules.

To initialize the premise parameters, we use the FCM [60] algorithm to partition the instances' space into 4 clusters[1]. We use the clusters' centers as initial centers for the Gaussian MFs, and we initialize all standard deviation parameters to a default value of 0.5. We want to emphasize here that the FCM setp is for the purpose of initialization only. It is used to identify dense regions of the input space, these region can contain positive and/or negative instances. The learning phase will keep and tune only regions that corresponds to positive concepts.

[1] A grid or manual partitioning could also be used

The initial fuzzy sets (MFs) of the rules' premise parts, before training, are displayed in Figure 5.3a. As it can be seen, the initial 4 clusters simply cover the 4 quadrants of the 2D instance space (if no label information is used, as in this case, data would appear to have uniform distribution (refer to Figure 4.5)). The updated parameters after 20 training epochs are shown in Figure 5.3b, and the learned fuzzy sets after convergence are shown in Figure 5.3c.

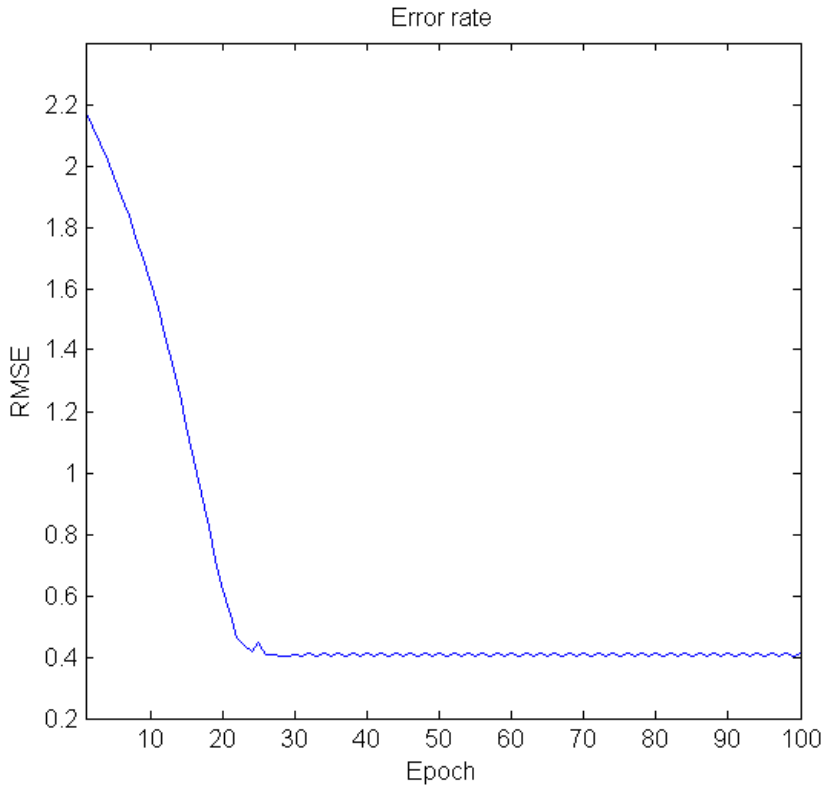
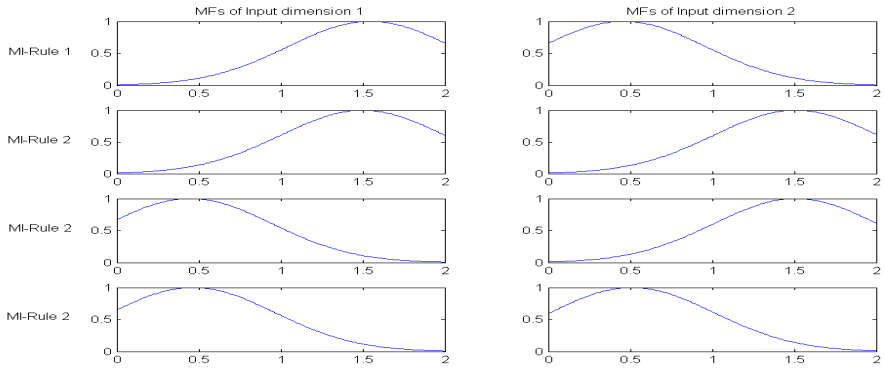
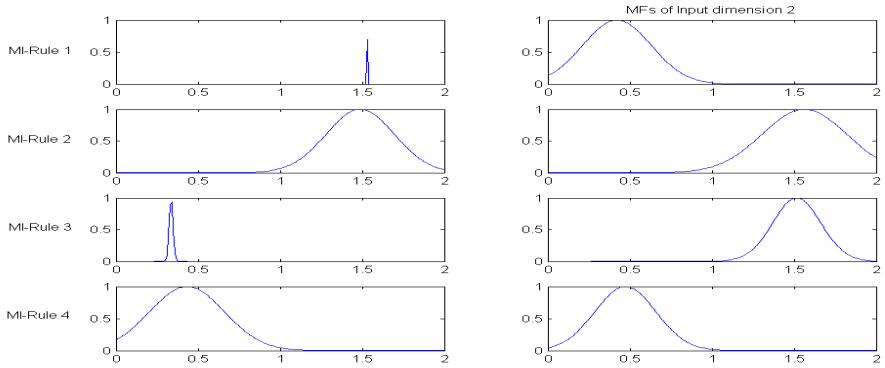


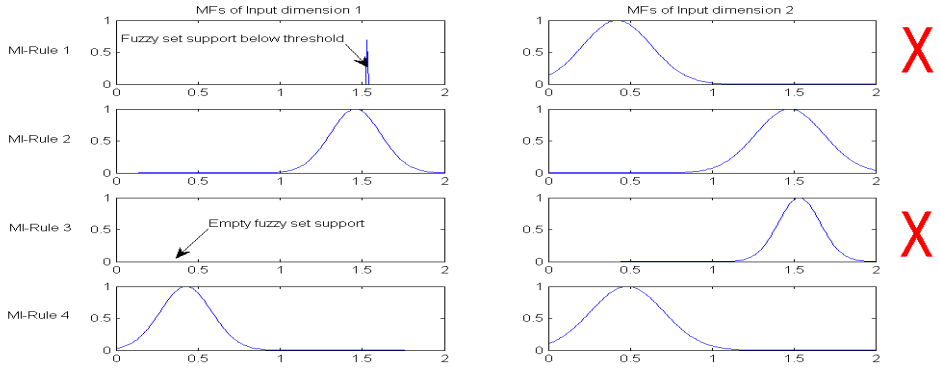
Figure 5.2: Root Mean Squared Error of 100 Epochs of MI-ANFIS training.



(a) Initial MFs before starting the training process.



(b) Input MFs during MI-ANFIS training (Epoch number 20).



(c) Learned MFs after MI-ANFIS training completed. Rules marked with red crosses are considered vanished and are pruned. Remaining rules (MI-Rule 2 and MI-Rule 4) correctly describe the positive concepts of the dataset

.

Figure 5.3: Input MFs before, during, and after MI-ANFIS training.

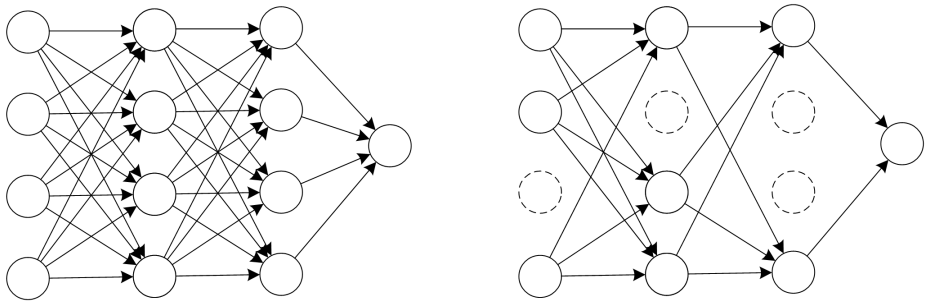
The system has correctly identified the positive concepts, and at the same time identified irrelevant rules (MI-Rule 1 and MI-Rule 3 marked with red crosses in Figure 5.3c). These rules are considered irrelevant either because some of its fuzzy sets has empty support (per consequence it cannot be activated), or very narrow fuzzy set support which may indicate overfitting and will not be general enough to use during testing. After training, it is recommended to detect and prune such rules to improve the MI-ANFIS testing efficiency (e.g., requiring minimum support).

5.4 Preventing Overfitting: Rule Dropout

Neural networks with large number of parameters are susceptible to overfitting. MI-ANFIS is no exception, particularly when using large number of multiple instance fuzzy rules and relatively small training datasets. In such scenario, some rules could co-adapt to the training data and degrade the network ability to generalize to unseen examples. In the previous paragraph we suggested pruning irrelevant rules, in this section, we present a more general technique, known as Dropout, used to prevent overfitting and rules' co-adaptation. Dropout is a new regularization method that was introduced by Hinton et al. [93] to alleviate the serious problem of overfitting in deep neural networks. Over the years, many methods have been developed to reduce overfitting, including using a validation dataset to stop the training as soon as the performance gets worse, adding weight penalties using L1 and L2 regularization, or artificially augmenting the training dataset using label-preserving transformations. However, as noted by Hinton [93], the best way to regularize a fixed-size model is to average the predictions of all possible settings of the parameters weighted by its posterior probability given the training data. This can be achieved by combining the predictions of an exponential number of models. Combining several models with different architectures have the advantage of better generalization and per consequence better testing performance. While generating an ensemble of models is trivial, training them all is prohibitively expensive.

Generally, Dropout works by setting to 0 the output of each node in a given layer with

probability $1 - \mathbf{p}$ (p typically equals 0.5), during training. Nodes that are dropped out do not contribute to the parameters updates. During testing, all nodes are used but the outputs are weighted by the probability $\mathbf{p}$. Following this strategy, every time a new training example is presented, the network samples and trains a different architecture. In other words, Dropout trains an ensemble of networks (2^N networks, N being the number of nodes) simultaneously leading to an important speedup in training time as compared to traditional ensemble methods. Figure 5.4 and Figure 5.5 illustrate the Dropout model.



(a) Standard Neural Network

(b) Network after Dropout

Figure 5.4: Dropout neural network model. (A) is a standard neural network. (B) is the same network after applying dropout. Doted lines indicate a node that has been dropped. (source [93])

(a) At training time

(b) At test time

Figure 5.5: Illustration of Dropout application. (A) a node is dropped with probability $1-p$ at training time. (B) at test time the node is always present and its outputs are weighted by p. (source [93])

We propose to adopt Dropout to regularize MI-ANFIS networks. Typically, overfitting occurs in MI-ANFIS networks when a set of multiple instance rules co-adapt to the provided data early during the training process, making the remaining rules irrelevant to

learn. Thus, degrading the network's generalization capability. While the Dropout technique could be applied to MI-ANFIS as is (given the inherited neural network nature of the architecture), care should be exercised when selecting nodes to include in the list of the randomly dropped out nodes. MI-ANFIS nodes are different from that of standard neural networks, when grouped into a rule, they model meaning and express linguistic terms. Dropping few nodes from a given rule could severely handicap the fuzzy inference process. Hence, Dropout should be executed differently. In deep neural nets, Dropout is applied to selected layers (vertically), for MI-ANFIS, we propose to apply Dropout on a rule by rule basis (i.e., horizontally). Either the whole rule is included, or the whole rule is dropped. This can be achieved by applying Dropout to Layer 5 (see Figure 5.7), i.e., setting to zero the output of the "to be dropped out" rules. We will refer to this derived technique as "**Rule Dropout**". Using a Rule Dropout strategy to train MI-ANFIS networks is approximatively equivalent to sampling and training 2^R (R is the number of rules) ensemble of networks.

Let p be the probability with which a rule is present, formally, Rule Dropout is applied to Layer 5 during training as following

$$h_i \sim Bernoulli(p) \tag{5.43}$$

$$O_{5,i} = h_i \overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i), \tag{5.44}$$

where h_i is a Bernoulli random variable with probability p of being 1. During testing, Layer 5 output is scaled by p, i.e., $O_{3,i} = p\overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i)$. Figure 5.6 and Figure 5.7 show an illustration of an MI-ANFIS network with 3 multiple instance fuzzy rules and implementing Rule Dropout.

Deriving the new update equations for MI-ANFIS parameters requires taking into consideration the added Bernoulli random variable, h_i. It is straightforward to show that

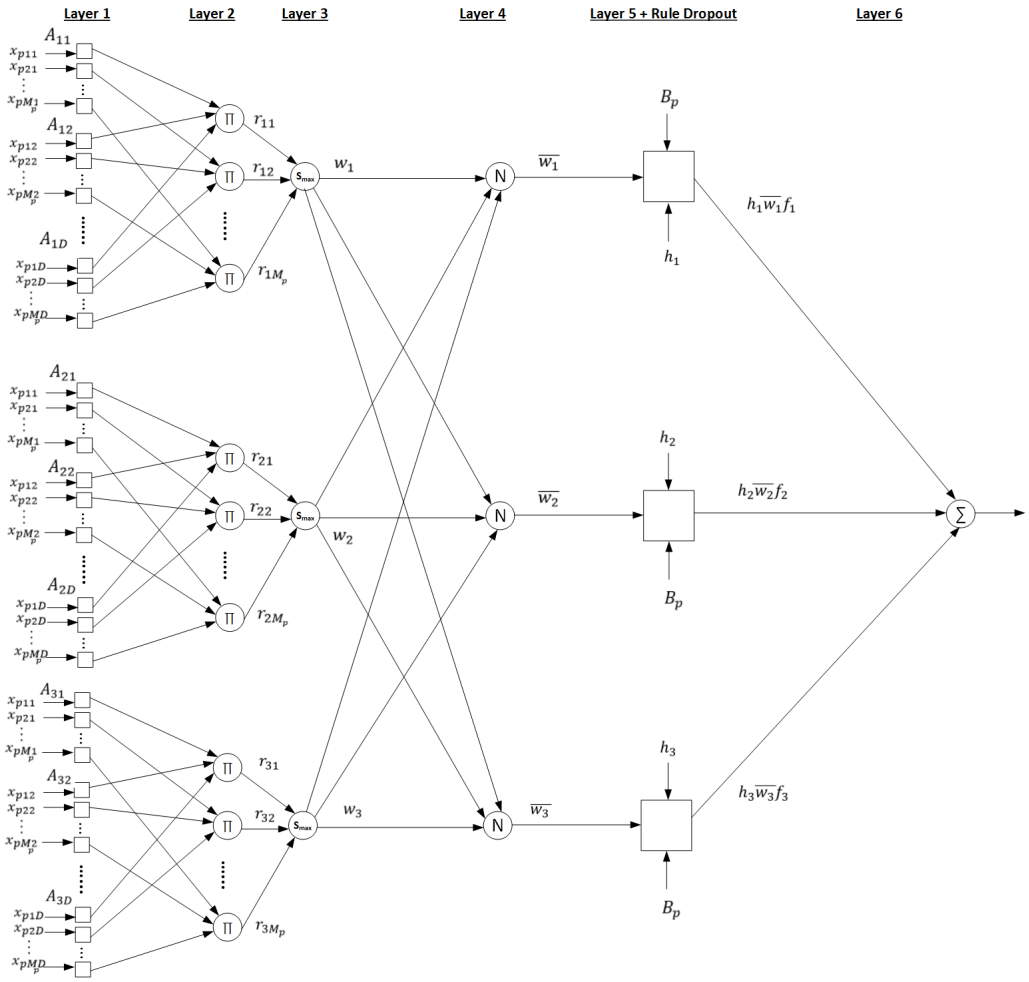


Figure 5.6: Rule Dropout MI-ANFIS model.

the new gradients with respect to premise and consequent parameters are given by

$$\begin{aligned}\frac{\partial E_p}{\partial c_{kj}} = &-2(t_p - O_p) \times h_k \times \mathcal{S}_\alpha(\mathbf{x}_{p1}\cdot\mathbf{b}^k,\ \mathbf{x}_{p2}\cdot\mathbf{b}^k, \ldots, \mathbf{x}_{pM_p}\cdot\mathbf{b}^k) \times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\left(\sum_{l=1}^{|O_3|} w_l\right)^2} \\ &\times \sum_{i=1}^{M_p}\left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}}\left[1+\alpha\left(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p})\right)\right]\right. \\ &\times \prod_{d=1,d\neq j}^{D} \mu_A\left(\left\lceil (i+(k-1)M_p)/M_p \right\rceil, d\right)\left(x_{p,(i+(k-1)M_p)[M_p],d}\right) \\ &\left.\times \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})}{\sigma_{kj}^2} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2})\right).\end{aligned} \tag{5.45}$$

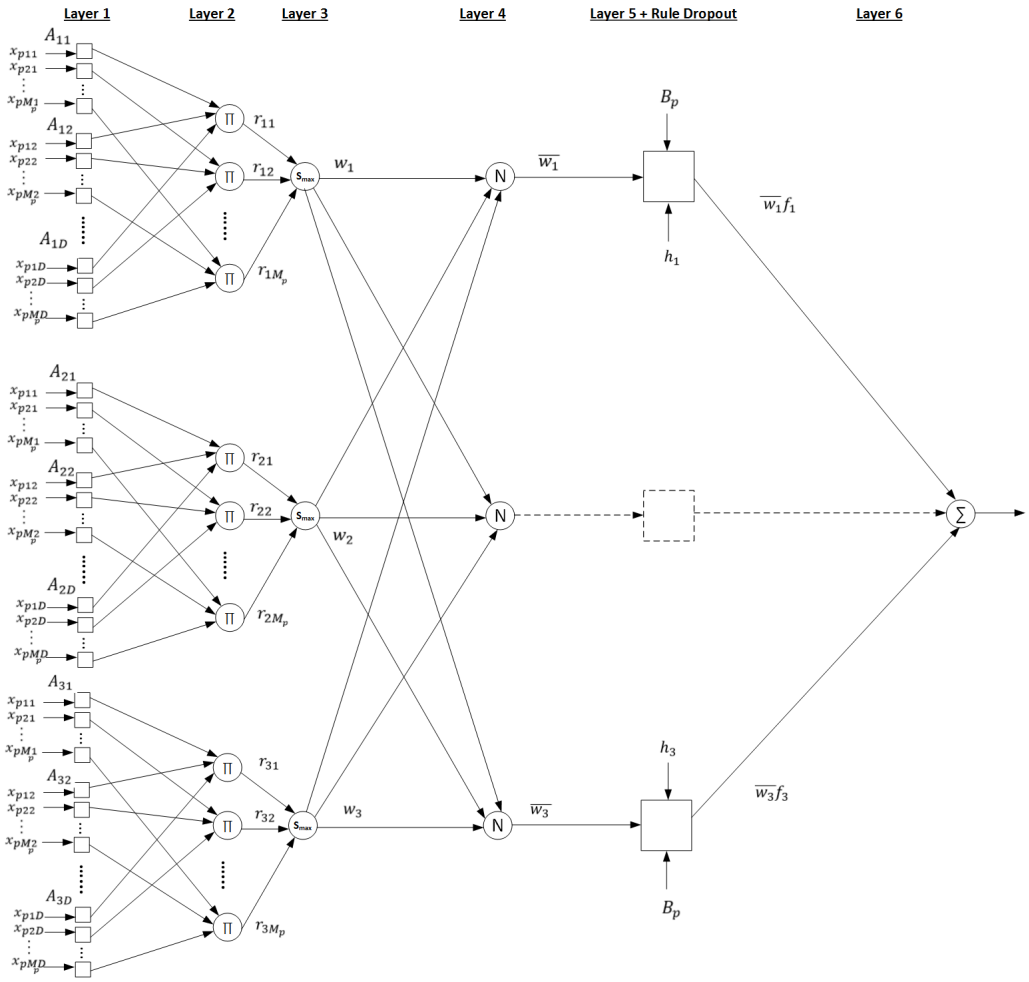


Figure 5.7: Illustration of Rule Dropout application. Doted lines indicate a rule that has been dropped.

and,

$$\frac{\partial E_p}{\partial \sigma_{kj}} = -2(t_p - O_p) \times h_k \times \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^k,\ \mathbf{x}_{p2} \cdot \mathbf{b}^k, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^k) \times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\left(\sum_{l=1}^{|O_3|} w_l\right)^2}$$
$$\times \sum_{i=1}^{M_p} \left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}} \left[1 + \alpha\left(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p})\right)\right] \right.$$
$$\times \prod_{d=1, d\neq j}^{D} \mu_A\left(\left\lceil (i+(k-1)M_p)/M_p \right\rceil, d\right) \left(x_{p,(i+(k-1)M_p)[M_p],d}\right) \tag{5.46}$$
$$\left. \times \frac{\left(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj}\right)^2}{\sigma_{kj}^3} \times exp(-\frac{\left(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj}\right)^2}{2\sigma_{kj}^2}) \right).$$

In a similar manner,

$$\frac{\partial E}{\partial b_j^i} = \sum_{p=1}^{N} -2(t_p - O_p) \times h_i \overline{w}_i \sum_{m=1}^{M_p} \frac{1}{\left(\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i))\right)^2} \Big[\Big(x_{pmj} \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i)\Big) - \Big(\mathbf{x}_{pm} \cdot \mathbf{b}^i \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i)\alpha(x_{phj} - x_{pmj})\Big)\Big]. \tag{5.47}$$

As it can be seen, equations (5.45), (5.46), and (5.47) will get zeroed when the rule is dropped out (i.e., $h_k = 0$ and $h_i = 0$). Thus, its premise and consequent parameters are not updated.

In order to demonstrate the gain in generalization acquired by MI-ANFIS when utilizing Rule Dropout, we train an MI-ANFIS architecture with and without Rule Dropout on a multiple instance dataset sampled from COREL [94]. The dataset classify whether an image contains an elephant or not, and consists of 200 images (bags): 100 positive images containing the target animal and 100 negative images containing other animals. Each image is represented as a set of patches (instances) and each patch is in turn represented by 230 features describing color, texture and shape information. Before training, we applied PCA to reduce the dimensionality of the features to 10 dimensions to speedup MI-ANFIS. Table 5.1 summarizes the dataset characteristics. Two MI-ANFIS networks composed of 15 rules each, with one network employing Rule Dropout (with $p = 0.7$), were trained on 90% of the data, and the remaining 10% was used for testing (split was done randomly). Figure 5.8 shows the training and testing errors for both networks during 100 epoches. As it can be seen, without Rule Dropout, starting at epoch 20, testing performance begins to degrade while training error continues to decrease. In other words, overfitting begins to occur. On the other hand, when using Rule Dropout, overfitting is significantly reduced and MI-ANFIS achieved better testing performance at the end of the training phase (0.1123 testing SSE with Rule Dropout compared to 0.1451 testing SSE without Rule Dropout).

TABLE 5.1

Benchmark data sets

Data set	dim.(PCA)	No. Bags	Positive	Negative	No.Instances
Elephant	230(10)	200	100	100	$2 \rightarrow 13$

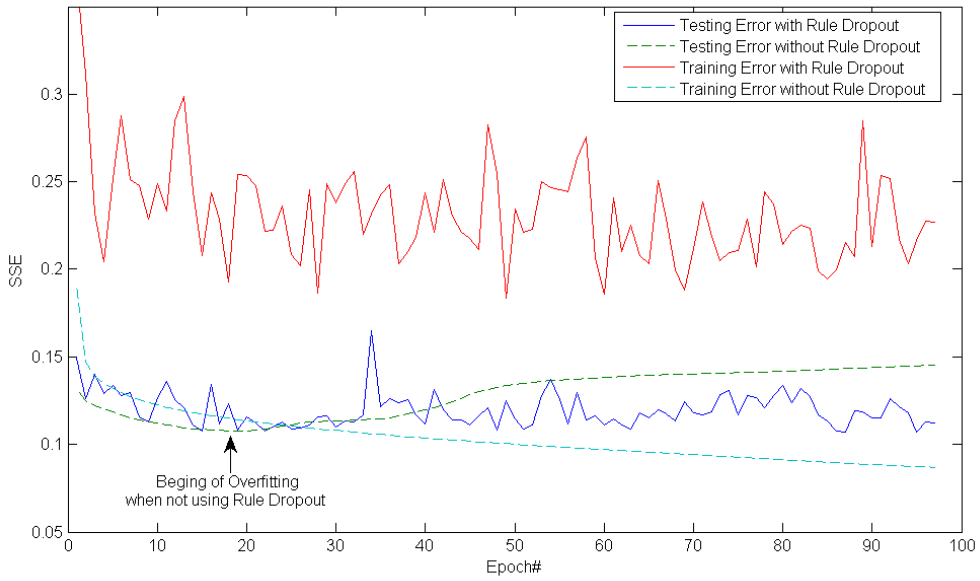


Figure 5.8: Training and testing errors for two MI-ANFIS networks with and without Rule Dropout.

5.5 Multi-Class MI-ANFIS

The basic MI-ANFIS architecture illustrated in Figure 5.1 computes one single output. It is suitable for binary classification problems, and through the use of a sum of squared error (SSE) loss function, it can effectively be used to solve multiple instance regression problems. The extension of MI-ANFIS to multiple class classification problems can be achieved through the use of a set of multiple independent MI-ANFIS networks and using a one versus the rest training pattern. Formally, for a set of N training bags, $\mathcal{B} = \{\mathbf{B}_p \mid p = 1, \ldots, N\}$, and their corresponding labels $\mathcal{T} = \{t_p \mid p = 1, \ldots, N,\ t_p \in [1 \ldots T]\}$, where T is the number of classes of a given multiple class classification problem, we define T different MI-ANFIS networks, denoted as $\{net_t\}_1^T$. To train each network, bags are relabeled as positive for

bags that belong to the positive class, negative otherwise. During testing, a new unseen bag is fed through the T networks and T outputs are computed, the bag is then assigned the class with the highest output. While this extension is straightforward and works with an arbitrary number of classes, it requires an extensive amount of preprocessing to relabel the data and generate networks. Moreover, doing so makes the data unbalanced and sampling should be used before training. In addition, some classes may share the same concepts, therefore, training different networks may lead to redundant rules being learned and wasting CPU cycles. Thus, it is better to restructure MI-ANFIS to support multiple classes.

In the following, we describe the extended Multi-Class MI-ANFIS (MCMI-ANFIS), and re-derive the necessary update equations. MCMI-ANFIS employs multiple instance fuzzy rules and has similar architecture to MI-ANFIS, Figure 5.9 is an illustration of the extended architecture. Layer 1 through Layer 5 are the same as in MI-ANFIS. Layer 6 is a fully connected layer, it's composed of T nodes that compute the sum of all incoming signals as following,

$$O_{6,j} = \sum_{i=1}^{|O_3|} v_{ij} h_i \overline{w}_i \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^i,\ \mathbf{x}_{p2} \cdot \mathbf{b}^i, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^i). \tag{5.48}$$

where v_{ij}, are weights as in standard feedforward neural networks. Layer 7 is an additional layer that computes the log-probabilities of Layer 6's outputs through the application of the LogSoftmax function which is given by

$$O_{7,j} = log\Big[\frac{exp(O_{6,j})}{\sum_{k=1}^{T} exp(O_{6,k})}\Big]. \tag{5.49}$$

The reason behind applying LogSoftmax is to prepare the network's outputs to be used with a negative log likelihood criterion that is typically used to train classification problems with multiple classes. Given this criterion, the loss function of MCMI-ANFIS, for a given bag B_p with class t_p (t_p is a class index in $[1 \ldots T]$), is defined by

$$E_p = -O_{7,t_p} \tag{5.50}$$

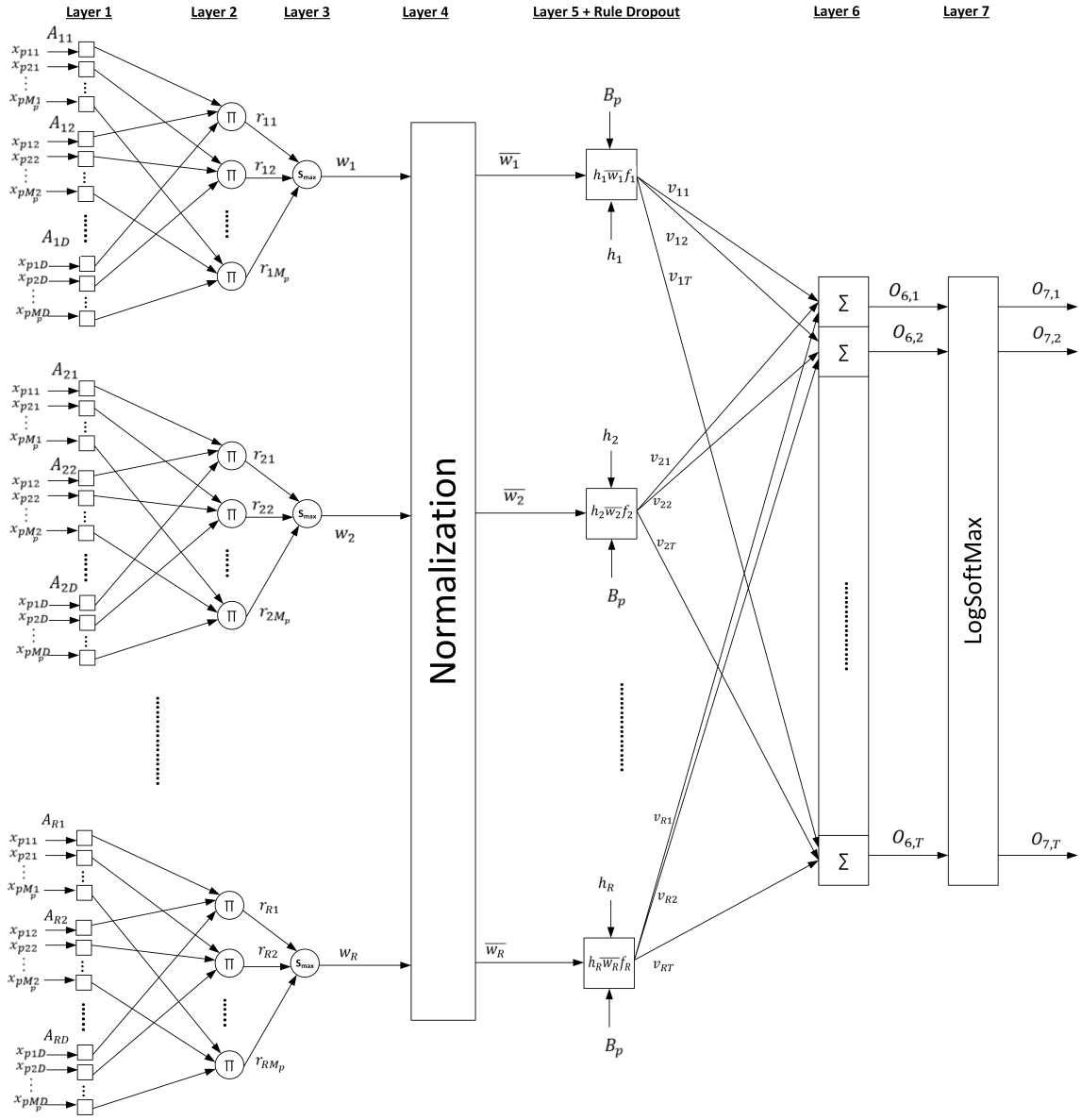


Figure 5.9: Multi-Class MI-ANFIS with R rules and T classes (outputs).

We now apply the chain rule (5.14) and derive MCMI-ANFIS update equations. First, we compute the gradients with respect to the premise parameters using (5.17) and (5.18), we have

$$\begin{aligned}\frac{\partial E_p}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}} = &\sum_{t=1}^{T} \frac{\partial E_p}{\partial O_{7,t}} \times \frac{\partial O_{7,t}}{\partial O_{6,t}} \times \frac{\partial O_{6,t}}{\partial O_{5,k}} \times \frac{\partial O_{5,k}}{\partial O_{4,k}} \\ &\times \frac{\partial O_{4,k}}{\partial O_{3,k}} \times \frac{\partial O_{3,k}}{\partial O_{(2,i+(k-1)M_p)}} \times \frac{\partial O_{(2,i+(k-1)M_p)}}{\partial O_{(1,i+[(k-1)D+(j-1)]M_p)}}.\end{aligned} \tag{5.51}$$

It is straightforward to show that

$$\sum_{t=1}^{T} \frac{\partial E_p}{\partial O_{7,t}} \times \frac{\partial O_{7,t}}{\partial O_{6,t}} \times \frac{\partial O_{6,t}}{\partial O_{5,k}} = v_{kt_p} \times \Big[\frac{exp(O_{6,t_p})}{\sum_{k=1}^{T} exp(O_{6,k})} - 1 \Big]. \tag{5.52}$$

Thus, the update equations for the premise parameters are as following

$$\begin{aligned} \frac{\partial E_p}{\partial c_{kj}} &= v_{kt_p} \times \Big(\frac{exp(O_{6,t_p})}{\sum_{k=1}^{T} exp(O_{6,k})} - 1 \Big) \times h_k \times \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^k,\ \mathbf{x}_{p2} \cdot \mathbf{b}^k, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^k) \\ &\times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\Big(\sum_{l=1}^{|O_3|} w_l \Big)^2} \times \sum_{i=1}^{M_p} \left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}} \Big[1 + \alpha \Big(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p}) \Big) \Big] \right. \\ &\times \prod_{d=1, d \neq j}^{D} \mu_A \Big(\lceil (i+(k-1)M_p)/M_p \rceil, d \Big) \Big(x_{p,(i+(k-1)M_p)[M_p],d} \Big) \\ &\left. \times \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})}{\sigma_{kj}^2} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2}) \right). \end{aligned} \tag{5.53}$$

and,

$$\begin{aligned} \frac{\partial E_p}{\partial \sigma_{kj}} &= v_{kt_p} \times \Big(\frac{exp(O_{6,t_p})}{\sum_{k=1}^{T} exp(O_{6,k})} - 1 \Big) \times h_k \times \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^k,\ \mathbf{x}_{p2} \cdot \mathbf{b}^k, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^k) \\ &\times \frac{\sum_{l=1}^{|O_3|} w_l - w_k}{\Big(\sum_{l=1}^{|O_3|} w_l \Big)^2} \times \sum_{i=1}^{M_p} \left(\frac{e^{\alpha r_{k,(i+(k-1)M_p)}}}{\sum_{m=1}^{M_p} e^{\alpha r_{k,m}}} \Big[1 + \alpha \Big(r_{k,(i+(k-1)M_p)} - \mathcal{S}_\alpha(\{r_{k,m}\}_{m=1}^{M_p}) \Big) \Big] \right. \\ &\times \prod_{d=1, d \neq j}^{D} \mu_A \Big(\lceil (i+(k-1)M_p)/M_p \rceil, d \Big) \Big(x_{p,(i+(k-1)M_p)[M_p],d} \Big) \\ &\left. \times \frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{\sigma_{kj}^3} \times exp(-\frac{(x_{(p,(i+(k-1)M_p)[M_p],j)} - c_{kj})^2}{2\sigma_{kj}^2}) \right). \end{aligned} \tag{5.54}$$

Similarly the update equation for the consequent parameters is given by

$$\begin{aligned} \frac{\partial E}{\partial b_j^i} &= \sum_{p=1}^{N} v_{kt_p} \times \Big(\frac{exp(O_{6,t_p})}{\sum_{k=1}^{T} exp(O_{6,k})} - 1 \Big) \\ &\times h_i \overline{w}_i \sum_{m=1}^{M_p} \frac{1}{\Big(\sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i)) \Big)^2} \Big[\Big(x_{pmj} \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i) \Big) \\ &- \Big(\mathbf{x}_{pm} \cdot \mathbf{b}^i \sum_{h=1}^{M_p} exp(\alpha(\mathbf{x}_{ph} \cdot \mathbf{b}^i - \mathbf{x}_{pm} \cdot \mathbf{b}^i) \alpha(x_{phj} - x_{pmj}) \Big) \Big]. \end{aligned} \tag{5.55}$$

Finally, the gradients with respect to the fully connected layer weights, v_{kt}, is given by

$$\frac{\partial E_p}{\partial v_{kt}} = h_k \overline{w}_k \mathcal{S}_\alpha(\mathbf{x}_{p1} \cdot \mathbf{b}^k,\ \mathbf{x}_{p2} \cdot \mathbf{b}^k, \ldots, \mathbf{x}_{pM_p} \cdot \mathbf{b}^k) \times \Big(\frac{exp(O_{6,t_p})}{\sum_{k=1}^{T} exp(O_{6,k})} - 1 \Big). \tag{5.56}$$

Equations (5.53), (5.54), (5.55), and (5.56) can be used to train MCMI-ANFIS either on-line (e.g., using stochastic gradient descent), or off-line in batch mode.

5.6 Complexity Analysis

We now study the asymptotic complexity for the execution of the proposed MI-ANFIS algorithms in term of four parameters: the number of training bags N, the average number of instances per bag M, the dimensionality of instances D, and the number of the multiple instance rules R. MI-ANFIS performs two passes: (1) a forward pass to compute the network output, as illustrated in Algorithm 5.1, and (2) a backward pass to backpropagate the gradients and update the parameters, as illustrated in Algorithm 5.2. First, for the forward pass, we perform the following sequential operations for each training bag:

1. Fuzzification of inputs: $M \times D \times R$ operations.
2. Evaluation of truth instances: $M \times R$ operations.
3. Computation of the rules activation degrees: R operations.
4. Normalization of the activation degrees: R operations.
5. Evaluation of Layer 5 outputs: $M \times D \times R$ operations.
6. Evaluation of the overall output: 1 operation.

Thus, the total number of operations during the forward pass for a given bag is asymptotically given by $R \times [2M \times D + M + 2] + 1 \backsimeq O(M \times D \times R)$. For MCMI-ANFIS networks we need to take into considerations the two additional layers which contribute an additional $T \times R + T$ operations (T being the number of classes). In the backward pass, for

each training bag we compute the gradients with respect to the premise and consequents parameters. The number of operations required to compute each gradient is:

1. For MF centers (equation (5.17)): $D \times R \times (1 + 2M \times D + R)$.
2. For MF standard deviations (equation (5.18)): $D \times R \times (1 + 2M \times D + R)$.
3. For the consequent parameters (equation (5.42)): $D \times R \times [1 + M \times (M \times D)]$.
4. For MCMI-ANFIS networks, there are $R \times T \times (M \times D + T)$ additional operations needed to compute the gradients with respect to the fully connected layer weights.

Therefore, the backward step performs approximatively $3DR+4MRD^2+2DR^2+D^2M^2R \backsimeq O(DR^2 + D^2M^2R)$ for MI-ANFIS networks, and $3DR + 4MRD^2 + 2DR^2 + D^2M^2R + RDMT + RT^2 \backsimeq O(DR^2 + D^2M^2R + RT^2)$ for MCMI-ANFIS networks. The overall asymptotic running time for a given training dataset with N bags is dominated by the backward pass and is equal to $O(NDR^2+ND^2M^2R)$, and $O(NDR^2+ND^2M^2R+NRT^2)$ for MCMI-ANFIS.

For problems with large number of training bags, relatively small number of rules, low dimensionality features, and constant number of instances, the big-O running time of the network is linear in terms of N, i.e., $O(N)$.

5.7 Discussion

MI-ANFIS deals with ambiguity by introducing the novel concept of truth instances: when carrying reasoning using a bag of instances at Layer 2 (Figure 5.1), a proposition will not only have one degree of truth, it will have multiple degrees of truth (r_{ij}), we call truth instances. Thus, effectively encoding the third vagueness component of ambiguity and increasing the expressive power of traditional fuzzy logic.

Learning positive concepts from ambiguously labeled data has been the core task of various MIL algorithms (e.g. Diverse Density [39]). MI-ANFIS has proven that it can learn positive concepts effectively while jointly providing a fuzzy representation of such regions. The fuzzy

representation is combined into meaningful and simple multiple instance rules that can be easily visualized and interpreted.

Compared to previously proposed multiple instance neural networks, such as Multiple Instance Neural Networks [63] (MI-NN) and Multiple Instance RBF Neural Networks [95] (RBF-MIP), MI-ANFIS advantage is the use of multiple instance fuzzy logic to learn a fuzzy representation of true positive concepts. MI-NN only learns standard neural network weights that do not carry any information regarding concepts. On the other hand, while standard RBF neural networks have been shown to be equivalent to zero order traditional Sugeno systems under certain constraints [96], thus, capable of learning a fuzzy representation of the inputs, RBF-MIP networks have different architecture and they do not employ adaptive radial basis functions in the first layer. Instead, they represent the inputs by computing their distances to clusters of training bags. This later method is expensive and its success greatly depends on the quality of the training data as it takes into consideration all the training examples which my include wrongly (nosily) labeled bags. It does not lead to learning true positive concepts, only learning other discriminative regions of the bags space. Moreover, MI-ANFIS learning algorithms can be updated to support a wide range of loss functions (criterions) such as cross entropy [97], maximum margin [98], etc. MI-NN is designed to use a handcrafted loss function (see section 2.1.4) which is largely responsible for the multiple instance behavior of the system and cannot be changed without substantially changing the architecture of MI-NN. This could be disadvantageous if MI-NN is to be used to solve multiple instance - multiple class classification problems.

Finally, when compared to our proposed MI-Mamdani system, MI-ANFIS is fully independent. MI-Mamdani does require positive concepts to be learned using a different algorithm (e.g. FCMI), or based on intuition. MI-ANFIS does not rely on any traditional MIL algorithms and can learn its rule base from data.

CHAPTER 6

EXPERIMENTAL RESULTS

In this chapter, we provide a quantitative evaluation of the proposed farmework by applying it to benchmark datasets commonly used to evaluate MIL methods. First, we apply MI-MAMDANI and MI-ANFIS to the MUSK [37], Fox, Tiger, and Elephant datasets [94]. Then, we apply our muliple class MI-ANFIS (MCMI-ANIFS) to solve a 20 class classification problem derived from the COREL dataset [94]. The datasets are described as following.

6.1 Benchmark Datasets

- **The MUSK Dataset:**

  The MUSK dataset is the most commonly used data in the context of MIL. This MIL problem is a case of polymorphism ambiguity. The goal is to classify molecules by looking at their shapes. Each molecule can appear in several distinct shapes because of binding and twisting that might occur during interactions. Thus, a molecule can have different forms of expression. The objective is to classify whether a molecule smells musky [99]. To solve this problem using standard single instance learning, we first need to identify which form is responsible for the molecule behaviour. However, this process is tedious. Hence, the problem is better represented as a multiple instance problem. Two versions of the dataset were released, MUSK1 and MUSK2. In both datasets, each bag represents a molecule and instances within each bag represent the different low-energy conformations of the molecule. Each instance is characterized by 166 features. MUSK1 has 92 bags, of which 47 are positive, and MUSK2 has 102 bags, of which 39 are positive.

- **Fox, Tiger, and Elephant Datasets:**

These datasets classify whether an image contains the corresponding animal. Each dataset consists of 200 images (bags): 100 positive images containing the target animal and 100 negative images containing other animals. Each image is represented as a set of patches (instances) and each patch is in turn represented by 230 features describing color, texture, and shape information [94].

- **The COREL Dataset:**

  To evaluate our proposed multi-class MCMI-ANFIS algorithm, we use it to categorize images from the COREL image dataset. In particular, we use the Corel-1000 and Corel-2000. Corel-1000 has 1000 images that cover 10 categories and Corel-2000 has 2000 images with 20 categories, respectively. Each category has 100 images and each image is represented by a bag consisting of instances obtained via extracting features from segmented regions of the images. Each instance is a 9-D feature vector characterizing the color, texture, and shape properties of a segmented region. For the sake of fair comparison we adopted the same data settings and image segmentation algorithm used in previous state of the art work [43]. Figure 6.1 shows images randomly sampled from the 20 categories and the corresponding segmentation results.

Table 6.1 summarizes the characteristics of the MUSK, Fox, Tiger, and Elephant Datasets. Table 6.2 describes the categories of the COREL datasets and the corresponding number of instances. We should note that for each dataset, the bags have a variable number of instances. For instance, for MUSK1 data, the number of instances per bag varies from 2 to 40. To reduce the dimensionality of the features in order to speedup MI-FIS training and increase the interpretability of the generated multiple instance fuzzy rules, we apply the PCA. In Table 6.1 we show both the original and reduced dimensions for each dataset.

Figure 6.1: Images randomly sampled from 20 categories and the corresponding segmentation results. Segmented regions are shown in their representative colors (source [43]).

TABLE 6.1

MUSK, Fox, Tiger, and Elephant Datasets

Dataset	dim.(PCA)	No. Bags	Positive	Negative	No. Instances - Avg - Median
MUSK1	166(25)	92	47	45	2 $\rightarrow$ 40 - 5.17 - 4
MUSK2	166(25)	102	39	63	1 $\rightarrow$ 1044 - 64.69 - 12
Fox	230(10)	200	100	100	2 $\rightarrow$ 13 - 6.47 - 6
Tiger	230(10)	200	100	100	1 $\rightarrow$ 13 - 5.44 - 6
Elephant	230(10)	200	100	100	2 $\rightarrow$ 13 - 7.62 - 7

6.2 Evaluation of MI-MAMDANI and MI-ANFIS algorithms

For all experiments, to learn an MI-Mamdani system from the training data, first, as outlined in Section 4.3, we apply the FCMI to extract concept points. Next, we generate multiple instance fuzzy rules from concept points. Finally, the learned rules are combined into an MI-Mamdani multiple instance fuzzy inference system. We also construct zero-order MI-ANFIS systems with various number of multiple instance rules. We use Gaussian MFs

TABLE 6.2

20 Image Categories of the COREL dataset and the Corresponding Average Number of Instances (regions)

Category ID	**Category Name**	**Instances per Image**
1	African people and villages	4.84
2	Beach	3.54
3	Historical building	3.1
4	Buses	7.59
5	Dinosaurs	2.00
6	Elephants	3.02
7	Flowers	4.46
8	Horses	3.89
9	Mountains and glaciers	3.38
10	Food	7.24
11	Dogs	3.80
12	Lizards	2.80
13	Fashion models	5.19
14	Sunset scenes	3.52
15	Cars	4.93
16	Waterfalls	2.56
17	Antique furniture	2.30
18	Battle ships	4.32
19	Skiing	3.34
20	Desserts	3.65

to describe the input fuzzy sets. For initialization, we use the FCM [60] algorithm to cluster the instances of the positive bags into a prefixed number of clusters, and we initialize MFs' centers as the clusters centers. Table 6.3 summarizes all parameters used in training the MI-ANFIS. We note that the reason behind using large standard deviations For MUSK1, MUSK2 datasets is to allow the initial rules to cover the entirety of the input space.

To illustrate the advantage of using MI-ANFIS over the traditional ANFIS, we compare these two algorithms on the first five datasets. Since ANFIS cannot learn from ambiguously labeled data, for the sake of comparison, we consider the naive MIL assumption where

TABLE 6.3

MI-ANFIS Training Parameters

Parameter	**MUSK1**	**MUSK2**	**Fox**	**Tiger**	**Elephant**
No. of MI Rules	6	3	2	4	3
No. of Inputs	25	25	10	10	10
MF's σ	100	100	10	10	10
Output parameters	1s	1s	1s	1s	1s
Softmax's α	1	1	1	1	1
Learning rate	0.1	0.1	0.1	0.1	0.1

TABLE 6.4

Comparison of MI-ANFIS prediction accuracy (in percent) to Naive-ANFIS on the benchmark data sets (averaged over five runs)

Algorithms	**MUSK1**	**MUSK2**	**Fox**	**Tiger**	**Elephant**
MI-ANFIS	93.49	90.58	66.4	84.5	86.97
	±0.76	±1.31	±2.77	±0.61	±1.10
Naive-ANFIS	67.82	79.43	58.70	77.70	82.2
	±4.04	±5.04	±1.35	±0.83	±0.83

all instances from positive bags are considered positive and all instances from negative bags are considered negative. We refer to this implementation as Naive-ANFIS. The results are summarized in Table 6.4 where the performance is reported in terms of prediction accuracy averaged over all 10 cross validation sets (% of correct ± standard deviation). As it can be seen, MI-ANFIS outperforms Naive-ANFIS significantly. This is because inaccurately labeled instances within the positive bags were used for training the Naive-ANFIS.

Table 6.5 shows the performance of the proposed algorithms as compared to state of art MIL algorithms on the first five benchmark datasets. MI-MAMDANI and MI-ANFIS were trained and tested using ten fold cross validation. Table 6.6 summarizes the average running time of cross validation of MI-ANFIS as compared to other algorithms on the benchmark datasets.

Overall, MI-ANFIS is comparable to other MIL algorithms. In fact, on all tested

TABLE 6.5

Comparison of MI-ANFIS prediction accuracy (in percent) to other methods on the benchmark data sets. Results for 3 top performing methods are shown in bold font. We use reported results, N/A indicated that a given algorithm was not applied to that dataset

Algorithms	**MUSK1**	**MUSK2**	**Fox**	**Tiger**	**Elephant**
MI-ANFIS	**93.49** ±0.76	**90.58** ±1.31	**66.4** ±2.77	**84.5** ±0.61	**86.97** ±1.10
MILES [100]	86.3	87.7	N/A	N/A	N/A
APR [37]	92.4	89.2	N/A	N/A	N/A
DD [39]	88.9	82.5	N/A	N/A	N/A
DD-SVM [101]	85.8	**91.3**	N/A	N/A	N/A
EM-DD [58]	84.8	84.9	56.1	72.1	78.3
Citation-KNN [67]	92.4	86.3	N/A	N/A	N/A
MI-SVM [94]	77.9	84.3	57.8	84.0	81.4
mi-SVM [94]	87.4	83.6	58.2	78.4	82.2
MI-NN [102]	88.0	82.0	N/A	N/A	N/A
Bagging-APR [103]	**92.8**	**93.1**	N/A	N/A	N/A
RBF-MIP [95]	91.3 ±1.6	90.1 ±1.7	N/A	N/A	N/A
BP-MIP [63]	83.7	80.4	N/A	N/A	N/A
RBF-Bag-Unit [104]	90.3	86.6	N/A	N/A	N/A
MI-kernel [105]	88.0	89.3	60.3	84.2	84.3
PPPM-kernel [106]	**95.6**	81.2	60.3	80.2	82.4
MIGraph [105]	90.0	90.0	61.2	81.9	**85.1**
miGraph [105]	88.9	90.3	61.6	**86.0**	**86.8**
ALP-SVM [107]	86.3	86.2	**66.0**	**86.0**	83.5
MIForest [108]	85.0	82.0	64.0	82.0	84.0
MI-MAMDANI	88.33 ±1.67	74.0±3.2	**65.4** ±1.1	79.9 ±1.6	79.5 ±1.5

datasets, MI-ANFIS ranked consistently among the top three. For MUSK1, PPPM-kernel [106] performed the best (95.6%), but this algorithm did not perform as well for the other sets. For MUSK2 Bagging-APR [103] achieved the best accuracy, as reported by [100]. MI-ANFIS achieved the best average performance for the Fox and Elephant datasets, and second best performance after the miGraph [105] and ALP-SVM [107] methods for the Tiger

TABLE 6.6

Comparison of MI-ANFIS running time (in Minutes) to other methods on the benchmark data sets.

Algorithms	**MUSK1**	**MUSK2**	**Fox**	**Tiger**	**Elephant**
MI-ANFIS	1.1	8	6	5.5	0.5
MILES [100]	29.1	130.2	N/A	N/A	N/A
DD [39]	2.85	32	N/A	N/A	N/A
DD-SVM [101]	612	1740	N/A	N/A	N/A
EM-DD [58]	3.75	15.5	3.3	14.36	5
Citation-KNN [67]	0.01	2.57	N/A	N/A	N/A
MI-SVM [94]	0.5	5.3	0.28	0.21	2.43

dataset. On the other hand, MI-MAMDANI performed better than 10 algorithms out of 19 tested on MUSK1, it also showed better performance than 7 algorithms out of 9 algorithms tested on FOX. However, MI-MAMDANI did not exhibit consistent performance on the rest of the benchmark datasets. MI-MAMDANI systems are constructed based on transforming concept points extracted using FCMI (or other MIL methods) into multiple instance fuzzy rules. In scenarios where bags have large number of instances (such as MUSK2), this handcrafted method does lead to accurate fuzzy representation of concepts, but further fine tuning should be used to improve the generated rules' consequent parts.

6.3 MCMI-ANFIS

Using the COREL dataset we train an MCMI-ANFIS to solve the problem of region-based image categorization. We adopted the same training and testing settings as other state of the art algorithms: images within each class were randomly split equally into a training set and a testing set. In the following, we report average results of five runs.

For both Corel-1000 and Corel-2000 experiments we construct a first order MCMI-ANFIS with 60 multiple instance fuzzy rules and employing Rule Dropout. Table 6.7 summarizes MCMI-ANFIS properties. The system is then trained for 2000 epoches, Table 6.8 and Table 6.9 report the confusion matrices of the two experiments.

TABLE 6.7

MCMI-ANFIS Training Parameters

Parameter	**Value**
No. of MI Rules	60
No. of Inputs	9
MF's σ	10
Rule Dropout Rate	0.2
Softmax's α	1
Learning rate	0.1

Analysis of the confusion matrix of the Corel-1000 experiment shows that the largest classification error occured between category 2 (Beach) and category 9 (Mountains and glaciers): 18.4% of Mountains and glaciers images were classified as beaches and 16.7% of Beach images were confused as Mountains and glaciers. African people and villages category exhibited the lowest performance, 65.9%. These observations are inline with pervious work [43], the large classification errors are due to the semantic richness of these categories as they contain multiple concepts that are similar to other categories. Analysing the confusion matrix of the Corel-2000 experiment reveals similar confusions as the Corel-1000, in addition 10% of the Desserts category images were confused with Beach and 20.9% of Mountains and glaciers images were misclassified as Waterfalls. Even though these categories are visually similar, the classification accuracy can be improved through the use of more distinctive features. However, for fairness of comparison we used the same feature set as previous art.

TABLE 6.8

Confusion matrix of MCMI-ANFIS on the region-based image categorization experiments using Corel-1000 Dataset (showing the run with the best overall accuracy, 83.8%). Each row shows the percentage of images in one category classified to each of the 10 categories.

Cat.	1	2	3	4	5	6	7	8	9	10
1	**65.9**	4.9	4.9	0.0	2.4	12.2	2.4	2.4	0.0	4.9
2	4.2	**66.7**	0.0	4.2	2.1	2.1	0.0	2.1	16.7	2.1
3	5.2	10.3	**81.0**	0.0	0.0	1.7	0.0	0.0	1.7	0.0
4	0.0	3.6	3.6	**89.1**	0.0	0.0	0.0	0.0	3.6	0.0
5	0.0	0.0	0.0	0.0	**92.9**	3.6	0.0	0.0	3.6	0.0
6	0.0	0.0	2.3	0.0	0.0	**86.4**	0.0	0.0	11.4	0.0
7	2.2	0.0	0.0	0.0	0.0	0.0	**97.8**	0.0	0.0	0.0
8	3.6	0.0	0.0	0.0	0.0	7.3	0.0	**85.5**	0.0	3.6
9	2.0	18.4	0.0	0.0	0.0	2.0	0.0	0.0	**77.6**	0.0
10	2.0	2.0	2.0	0.0	0.0	0.0	0.0	2.0	0.0	**91.8**

TABLE 6.9

Confusion matrix of MCMI-ANFIS on the region-based image categorization experiments using Corel-2000 Dataset (showing the run with the best overall accuracy, 70.1%). Each row shows the percentage of images in one category classified to each of the 20 categories.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**59.6**	0.0	4.3	0.0	0.0	2.1	4.3	0.0	0.0	0.0	6.4	2.1	2.1	4.3	4.3	2.1	0.0	0.0	0.0	8.5
2	2.0	**50.0**	2.0	0.0	0.0	0.0	0.0	0.0	24.0	2.0	2.0	0.0	0.0	2.0	2.0	0.0	0.0	4.0	4.0	6.0
3	4.2	4.2	**70.8**	0.0	0.0	2.1	0.0	0.0	2.1	0.0	2.1	2.1	0.0	0.0	0.0	0.0	2.1	4.2	2.1	4.2
4	0.0	3.4	5.1	**83.1**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.0	0.0	3.4	3.4	0.0
5	0.0	0.0	0.0	0.0	**86.8**	1.9	0.0	0.0	0.0	0.0	0.0	0.0	1.9	0.0	0.0	0.0	3.8	0.0	0.0	5.7
6	5.7	3.8	0.0	0.0	1.9	**64.2**	0.0	0.0	7.5	0.0	1.9	0.0	0.0	0.0	0.0	5.7	1.9	0.0	3.8	3.8
7	0.0	0.0	0.0	0.0	0.0	0.0	**88.1**	0.0	0.0	0.0	2.4	0.0	0.0	4.8	2.4	0.0	0.0	0.0	2.4	0.0
8	1.7	0.0	0.0	0.0	0.0	1.7	0.0	**81.4**	0.0	0.0	10.2	5.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	4.7	0.0	0.0	0.0	7.0	0.0	0.0	**46.5**	0.0	2.3	0.0	0.0	0.0	2.3	20.9	0.0	2.3	9.3	4.7
10	3.8	1.9	0.0	1.9	0.0	0.0	1.9	0.0	0.0	**77.4**	0.0	3.8	1.9	1.9	3.8	0.0	0.0	1.9	0.0	0.0
11	6.8	0.0	2.3	0.0	0.0	0.0	2.3	4.5	0.0	2.3	**63.6**	4.5	4.5	2.3	0.0	0.0	0.0	0.0	4.5	2.3
12	3.9	2.0	7.8	0.0	0.0	0.0	0.0	2.0	0.0	0.0	2.0	**72.5**	0.0	0.0	2.0	0.0	0.0	0.0	3.9	3.9
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.1	0.0	8.5	0.0	**76.6**	0.0	0.0	0.0	2.1	0.0	4.3	6.4
14	0.0	0.0	0.0	0.0	0.0	0.0	8.9	0.0	0.0	1.8	0.0	0.0	0.0	**66.1**	16.1	0.0	0.0	0.0	1.8	5.4
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.3	0.0	10.0	1.7	10.0	**61.7**	0.0	5.0	3.3	0.0	5.0
16	2.0	2.0	2.0	0.0	0.0	7.8	0.0	0.0	13.7	0.0	0.0	2.0	0.0	0.0	0.0	**70.6**	0.0	0.0	0.0	0.0
17	3.8	0.0	1.9	0.0	0.0	1.9	0.0	0.0	0.0	0.0	1.9	0.0	1.9	0.0	0.0	0.0	**83.0**	0.0	1.9	3.8
18	0.0	4.1	2.0	2.0	0.0	0.0	0.0	0.0	6.1	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	**77.6**	2.0	4.1
19	0.0	9.6	0.0	0.0	5.8	0.0	0.0	0.0	9.6	0.0	1.9	0.0	1.9	0.0	0.0	1.9	13.5	1.9	**53.8**	0.0
20	0.0	10.0	0.0	0.0	0.0	3.3	0.0	0.0	6.7	3.3	3.3	0.0	0.0	0.0	0.0	3.3	0.0	6.7	3.3	**60.0**

TABLE 6.10

Comparison of MCMI-ANFIS classification accuracy (in percent) to other methods on the Corel-1000 and Corel-2000 benchmark datasets

Algorithms	**Corel-1000**	**Corel-2000**
MCMI-ANFIS	82.1 ±1.5	69.7 ±0.4
MIGraph [105]	83.9	72.1
miGraph [105]	82.4	70.5
MI-Kernel [105]	81.8	72.0
MILES [43]	82.6	68.7
MI-SVM [94]	74.7	54.6
DD-SVM [101]	81.5	67.5
Kmeans-SVM [43]	69.8	52.3

Table 6.10 reports the classification accuracy averaged over five runs (% of correct ± standard deviation). Overall MCMI-ANFIS showed consistent performances on both datasets and achieved competitive results compared to other MIL methods reported in the literature. When compared to the top performing method MIGraph [105], MCMI-ANFIS showed comparable results. In addition, MIGraph, and most other methods, were trained and tested using one versus all training pattern, whereas MCMI-ANFIS learned all the concepts in one training pass, which is usually a more difficult task. Also MCMI-ANFIS performance was better than MILES [43], which was considered the state of the art algorithm on the Corel dataset until MIGraph and MI-Kernel were published. It is worth noting that on the binary classification problems of the previous section MI-ANFIS was better than MIGraph on 4 out of the 5 datasets. In general MI-ANIFS and MCMI-ANFIS showed competitive and consistent results on all benchmark datasets.

We note that Rule Dropout was necessary to train MCMI-ANFIS. Without Rule Dropout we observed overfitting, which led to a low 44% accuracy on the Corel-2000 dataset. This emphasizes the importance of regularization and the need for large datasets to train neural networks in general. For the Corel experiments, our the MCMI-ANFIS has 2880 parameters[1]

[1] $9\times2\times60$ premise parameters, 10×60 consequent parameters, and 60×20 fully connected layer parameters

to be learned, versus only 2000 training bags, making overfitting more likely to occur. Rule Dropout helped reducing this artifact significantly, leading to competitive performance.

CHAPTER 7

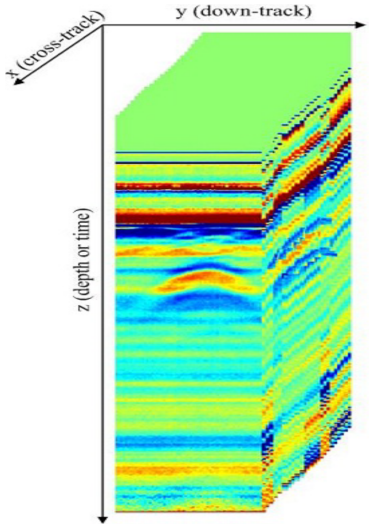
APPLICATION : LANDMINE DETECTION USING GROUND PENETRATING RADAR

In this Chapter, we apply the proposed multiple instance fuzzy inference framework to fuse multiple landmine detection algorithms. First, we start with an overview of the landmine detection problem and illustrate the need to solve this problem using multiple instance learning. Then, we describe the dataset used in the experiments. Next, we show how the fusion problem can be solved using traditional Mamdani and ANFIS inference. Finally, we develop fusion methods using our multiple instance fuzzy inference systems and report the results.

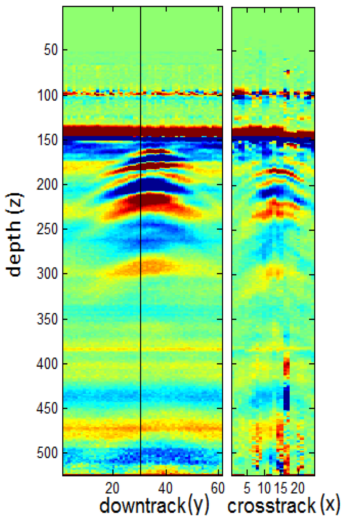
7.1 Landmine Detection

Detection and removal of landmines is a serious problem affecting human beings worldwide. The world is now littered with an estimated 200-215 million landmines in 91 countries, which maim or kill an estimated 500 people every week, mostly innocent civilians [109]. The task of detection of buried landmines is of extreme difficulty and this is mainly due to the large variety of landmine types, different soil type and compaction, temperature, moisture, shadow, time of day, weather conditions, and varying terrain, to name a few.

Varieties of sensors have been proposed or are under investigation for landmine detection. The research problem for sensor data analysis is to determine how well signatures of landmines can be characterized and distinguished from other objects under the ground using returns from one or more sensors. Recently, various discrimination algorithms [110–114] have been proposed for detecting buried objects using ground-penetrating radar (GPR)



(a) 3D GPR Raw data



(b) (depth, down-track) and (depth, cross-track) slices of GPR data

Figure 7.1: 3-dimensional and 2-dimensional raw GPR data.

[115, 116]. GPR offers the promise of detecting landmines with little or no metal content. The sensor works by emitting an electromagnetic wave covering a large frequency band into the ground through a wide-band antenna. Reflections from the soil caused by dielectric variations such as the presence of an object are measured. By moving the antenna, it is possible to reconstruct an image representing a vertical slice of the soil. The data generated are 3-dimensional and correspond to depth, down-track, and cross-track (Figure 7.1). Most discrimination algorithms process only 2-D slices of the 3-D cube: (down-track, depth) or (cross-track,depth). The performance of the down-track and cross-track discrimination algorithms can vary significantly depending on the target shape, burial orientation, and other environmental conditions. In some cases, these algorithms can provide complementary evidence, while in other cases they provide contradicting evidence. Thus, effective fusion of these algorithms can achieve higher probability of detection with fewer false alarms.

To train discrimination algorithms, we use data collected with known target locations. However, only the (down-track, cross-track) position can be extracted. The depth position is usually unknown as it depends on the burial depth, height of target, type of soil, height of GPR antenna above the ground, etc. Thus, there is uncertainty in the depth estimation of the targets that can affect both the training and testing phases of a fusion

system. For training, it is very difficult to localize the objects depth automatically, and it is a very tedious process to do it manually. Similarly, during testing, it is not trivial how to combine partial confidence values from multiple depths. Therefore, the MIL paradigm is suitable to solve this problem.

Several landmine discriminators could be used in the fusion system. In this dissertation, we validate our approach using four discrimination algorithms. Two of the algorithms are based on the Edge Histogram Descriptor (EHD) [117]. The first algorithm processes the 2-D (down-track, depth) slice of the 3-D GPR signal to generate partial confidence values at different depths, and is referred to as EHDDT (DT indicates down-track). Similarly, the second algorithm processes the 2-D (cross-track, depth) slice and is referred to as EHDCT (CT indicates cross-track). The other two discrimination algorithms are based on the Fisher Vector features [118]. In a like manner to EHD, one of the algorithms, called FisherVectorDT, extracts features from the (down-track, depth) view, the second algorithm, called FisherVectorCT, extracts information for the (cross-track, depth) view.

In the following, we briefly describe the GPR data and present the discrimination algorithms. More details can be found in [117, 119]. We also outline the extraction of two additional features that are used to refine the fusion rules when necessary.

7.1.1 GPR data

The data used in our multi-algorithms fusion system was collected using a vehicle mounted GPR (as shown in Figure 7.2). As the vehicle travels, it generates a 3-Dimensional matrix of sample values (shown in Figure 7.1a) that correspond to depth, down-track, and cross-track, $S(z, x, y), z = 1, ..., N_D; x = 1, ..., N_C; y = 1, ..., N_S$, where z, x, and y represent depth, cross-track, and down-track positions respectively, and N_D, N_C, and N_S represents the collected sample size along depth, cross-track, and down-track dimensions.

Figure 7.2: Vehicle mounted GPR system.

7.1.2 EHDDT and EHDCT algorithms

The EHDDT is the same as the standard Edge Histogram Descriptor (EHD) algorithm proposed by Frigui et al. [117]. The EHD uses translation invariant features, that are based on the histogram of edges in the GPR signatures, and a possibilistic $k-$Nearest Neighbors ($k-$NN) rule for confidence assignment [120]. The EHD is an adaptation of the MPEG-7 EHD feature [121] which captures the signature's texture as feature for recognition. It has been adapted to capture the spatial distribution of the edges within a $3-$D GPR data volume. To keep the computation simple, $2-$D edge operators are used, and two types of edge histograms are computed. The first one is obtained by fixing the cross-track dimension and extracting edges in the (depth, down-track) plane. The second edge histogram is obtained by fixing the down-track dimension and extracting edges in the (depth, cross-track) plane.

Let $S_{zy}^{(x)}$ be the x^{th} plane of the $3-$D signature $S(z,x,y)$. First, for each $S_{zy}^{(x)}$, four categories of edges are computed: vertical, horizontal, 45° diagonal, and 135° anti-diagonal. If the maximum of the edge strengths exceeds a preset threshold, the corresponding pixel is considered to be an edge pixel. Otherwise, it is considered a non edge pixel. Next, each $S_{zy}^{(x)}$ image is vertically subdivided into 7 overlapping sub-images $S_{zy_i}^{(x)}$, $i = 1,\ldots,7$. For each $S_{zy_i}^{(x)}$, a 5 bin edge histogram, $H_{zy_i}^{(x)}$, is computed. The bins correspond to the 4 edge categories, and the non-edge pixels.

The EHD is defined as the concatenation of the 7 five-bin histograms. That is,

$$\text{EHD}^y(S_{xyz}) = [\overline{H}_{zy_1}\ \overline{H}_{zy_2}\ \overline{H}_{zy_3}\ \ldots \overline{H}_{zy_7}], \tag{7.1}$$

where $\overline{H}_{zy_i}$ is the cross-track average of the edge histograms of sub-image $S^{(x)}_{zy_i}$ over N_C channels, i.e.,

$$\overline{H}_{zy_i} = \frac{1}{N_C}\sum_{x=1}^{N_C} \overline{H}^{(x)}_{zy_i}. \tag{7.2}$$

The EHDCT is a variation of the standard EHD and follows the same feature extraction process described above. However, while the EHDDT is mainly based on the (depth, down-track) slices, the EHDCT focuses only on the (depth, cross-track).

A given test GPR alarm has around 300 to 400 depth values. The buried object signature is not expected to cover all the depth values. Thus, extracting one global feature vector from the alarm may not discriminate between object and clutter signatures effectively. To avoid this limitation, each potential target (identified by a prescreener) needs to be tested at multiple depth values. Typically, a $30 \times 15 \times 7$ window is slided along the depth axis with a 50% overlap between 2 consecutive signatures. A total of 17 signatures are extracted for each target. Thus, each alarm would be represented by a bag of 17 instances. For each instance the EHD histograms (EHDDT and EHDCT) are extracted. Then, a possibilistic $k-$Nearest Neighbors ($k-$NN) rule is used to assign partial confidence values [120] for each instance individually. We should note here that the bag representation is used to group features from multiple depths, and is not used in an MIL context.

7.1.3 Fisher Vector discrimination algorithms

The Fisher Vector (FV) extracts features at multiple depths of the 3-D GPR signatures. First, each 2-D GPR view (i.e., (down-track, depth) or (corss-track, depth)) is divided into overlapping 60 windows along the depth axis. Next, each window is in turn divided into a set of sub patches using a grid partitioning. Then, 128-D dense SIFT [118] features are extracted for each sub patch, a sample window with extracted SIFT feature

is shown in Figure 7.3. Finally, the FV is used to aggregate the extracted set of features into a global feature vector for each window. In total, 60 FV features are extracted for the (down-track, depth) view and 60 FV features are extracted for the (cross-track, depth) view.

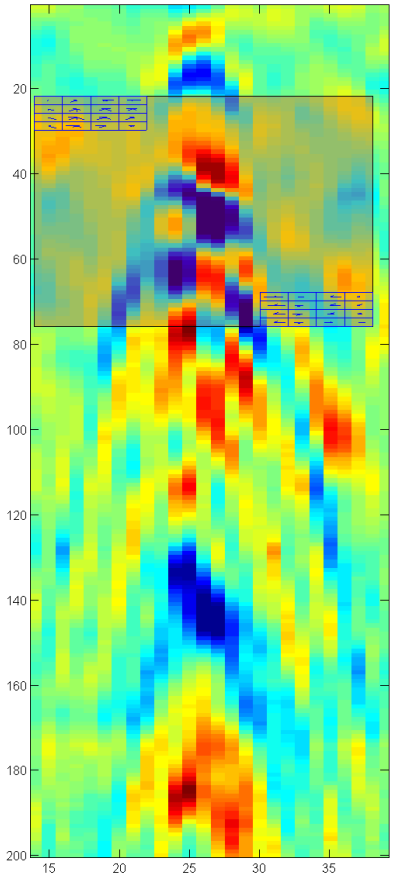


Figure 7.3: Sample GPR alarm with dense SIFT features (only first and last features are shown)

The FV patch aggregation mechanism is based on the Fisher Kernel. The Fisher Kernel characterizes a sub patch by its deviation from a generative model. The deviation is the gradient of the sub patch log-likelihood with respect to the generative model parameters. The vectorial representation of all the deviations is called the Fisher Vector (FV). For instance, using the extracted dense SIFT features (descriptors), a generative model, such as Gaussian

Mixture Model (GMM) with K words, is learned. It can be regarded as a "probabilistic visual vocabulary". Let $\mathcal{I} = (\mathbf{x_1}, \ldots, \mathbf{x_N})$ be a set of D dimensional feature vectors. Let $\Theta = (\mu_k, \Sigma_k, \pi_k : k = 1, \ldots, K)$ be the parameters of a GMM fitting the distribution of descriptors, where π_k , μ_k, and Σ_k are respectively the mixture weight, mean, and covariance matrix of Gaussian k. The GMM associates each vector $\mathbf{x_i}$ to a mode k in the mixture with a strength given by the posterior probability:

$$q_{ik} = \frac{\exp\left[-\frac{1}{2}(\mathbf{x_i} - \mu_k)^T \Sigma_k^{-1}(\mathbf{x_i} - \mu_k)\right]}{\sum_{t=1}^{K} \exp\left[-\frac{1}{2}(\mathbf{x_i} - \mu_t)^T \Sigma_k^{-1}(\mathbf{x_i} - \mu_t)\right]}. \tag{7.3}$$

For each mode k, we compute the mean and covariance deviation vectors

$$u_{jk} = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} q_{ik} \frac{x_{ji} - \mu_{jk}}{\sigma_{jk}},$$
$$v_{jk} = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} q_{ik} \left[\left(\frac{x_{ji} - \mu_{jk}}{\sigma_{jk}} \right)^2 - 1 \right].$$

where $j = 1, 2, \ldots, D$ spans the vector dimensions. The FV of a given GPR window is the concatenation of the vectors $\mathbf{u_k}$ and $\mathbf{v_k}$ for each of the K modes in the Gaussian mixtures, i.e.,

$$\Phi(I) = [\mathbf{u_1} \cdots \mathbf{u_k} \cdots \mathbf{v_1} \cdots \mathbf{v_k} \cdots] \tag{7.4}$$

Due to the absence of ground truth at the window level, a simple heuristic was used to label the data. It consists of assigning positive labels to windows with high energy, and negative otherwise. Having labeled the windows, SVM is then used to learn a classifier and assign partial confidences to the extracted 60 windows. Thus, each alarm would be represented by a bag of 60 instances. Each instance is a 2-D vector composed of FisherVectorDT confidence value and FisherVectorCT confidence value.

7.1.4 Auxiliary Feature Extraction

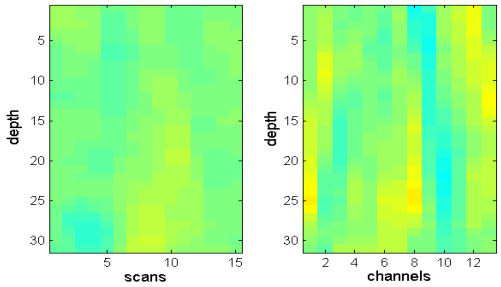
In some cases, EHDDT and EHDCT algorithms can provide complementary evidence, while in other cases they provide contradicting evidence. In the later case, a fusion system needs to trust one algorithm over the other. This can be achieved by learning appropriate linear combination of weights for algorithms within each local context. For this

method to be effective, extracted local contexts need to have: (i) a consistent algorithm that can be trusted and can lead to a better discrimination, or (ii) have a trivial solution due to context purity (a pure context includes mainly target signatures and only few to none non-targets signatures, or vice versa). However, because of the low dimensionality of the available inputs (only 2 dimensions, EHDDT and EHDCT), in some regions of the input space it may be difficult to obtain contexts in which a combination of the algorithms will improve the discrimination results. To improve the partition of the input space, we extract auxiliary features synthesized from the shape of the radar signal at certain depths.

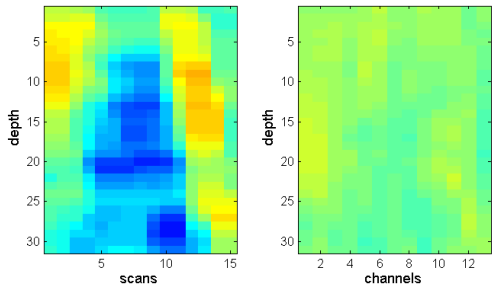
In the following, we outline the extraction of two auxiliary features: $SignatureWidth$ for Down-track; and $SignatureWidth$ for Cross-Track. As the names indicate, and by analogy to EHDDT and EHDCT, the two additional features consist of the effective width of the strong components within the GPR signal along (depth, down-track) slices and the width along (depth, cross-track) slices.

Let $B^{(x)}_{z_{(i)},y}$ be the 2 dimensional signature corresponding to the measured radar signal collected at a fixed cross-track position (referenced here by x) and encapsulating the 30 depth bins starting at $z_{(i)}$. In other words, $B^{(x)}_{z_{(i)},y}$ is one of the 17 signatures (instances) of one alarm. Similarly, let $B^{(y)}_{z_{(i)},x}$ be the 2 dimensional signature at a fixed down-track position (referenced here by y). Figure 7.4 displays 3 signatures extracted from target and non-target GPR alarms. As it can be seen, target signatures can be characterized by a right rising edge (45° diagonal), and a left decreasing edge (135° anti-diagonal). Typically, wider structures (covering more than 11 scans) can indicate the presence of an object of interest (due to known target sizes), and should lead to a higher probability of detection. $SignatureWidth$ auxiliary features are based on this observation.

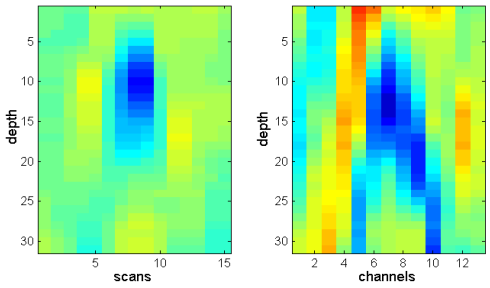
To extract the $SignatureWidth$ of a given instance, we use two of the edges computed for the EHD: 45° diagonal, and 135° anti-diagonal. These diagonal and anti-diagonal edge strengths are summed along the depth dimension. The resulting 1-D signals, called hereafter $DGStrength$ and $ADStrength$ respectively, are normalized by the number of instance depths (i). By thresholding the later signals we can extract two key locations



(a) Non target



(b) Target with strong DT signature



(c) Target with strong CT signature

Figure 7.4: Target and Non-Target signatures.

that define the spread of the strongest component within the instance, and thus obtain the $SignatureWidth$. These two key locations are respectively the points SD, where the $DGStrength$ starts rising above a threshold value $DGThresh$, and SA, where $ADStrength$ starts decreasing below a threshold value $ADThresh$.

Formally, SD is defined as

$$\text{SD} = min\{i \mid DGStrength_i > DGThresh\} \tag{7.5}$$

Similarly,

$$\text{SA} = max\{i \mid ADStrength_i > ADThresh\} \tag{7.6}$$

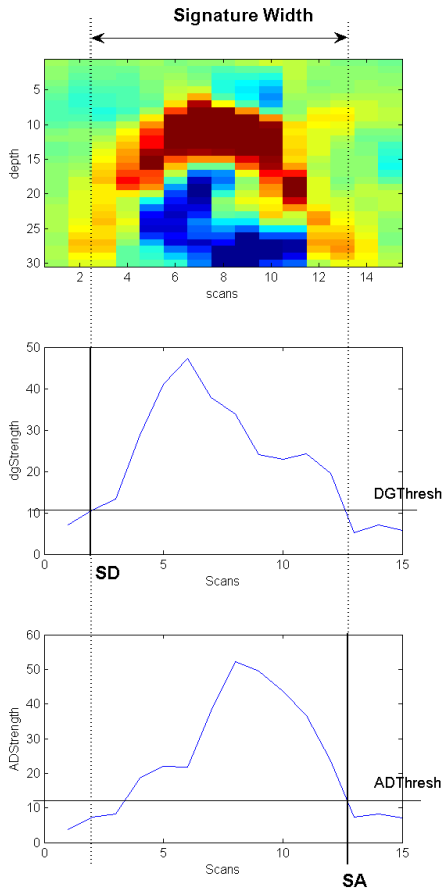


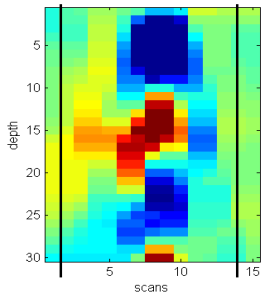
Figure 7.5: Illustration of the identification of the SA and SD points.

The $SignatureWidth$ is then defined as

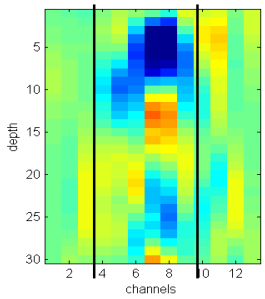
$$SignatureWidth = \begin{cases} SA - SD & \text{if} \quad SA > SD, \\ 0 & \text{otherwise} \end{cases} \tag{7.7}$$

The identification of the SD and SA points are illustrated in Figure 7.5. Examples of $SignatureWidth$ features are shown in Figure 7.6

Let $SignatureWidthDT$ be the width feature for Down-track and $SignatureWidthCT$ be the width feature for Cross-Track. Thus, each alarm is represented by a bag of 17 instances extracted at multiple depths. Each instance include 2 features: $SignatureWidthDT$, $SignatureWidthCT$.



(a) Down-Track, width = 13



(b) Cross-Track, width = 6

Figure 7.6: Examples of SignatureWidthDT (gprDT) and SignatureWidthCT (gprCT) features for a target object.

7.1.5 Data Collection

GPR data collected at different locations and different dates were used to evaluate our algorithms. In particular, two collections were used to train and test the proposed fusion methods. The first collection, Collection-1, was collected from two different sites and covers a variety of anti-tank mines including 319 encounters of anti-tank with high metal content(ATHM) and 422 encounters of anti-tank with low metal content (ATLM). In addition, a variety of clutter objects were surveyed in an effort to test the robustness of the fusion algorithms. The targets were buried up to 8 inches deep. First, a prescreener [122] is used to process the GPR data and identify regions of interest to be processed further by the discrimination algorithms. The prescreener identified 700 target encounters and 330 non-targets (false alarms). Collection-1 is used in the following to perform 10-fold corss-validation. The second collection, Collection-2, was collected from three different sites. The first two sites cover 789 target encounters of which 339 were of type ATHM and 450 ATLM, also 1577 non-targets were identified in the first two sites. The third site of Collection-2 covers 1948 targets (847 ATHM and 1097 ATLM) and 3018 non-targets. In the following, Site 1 & Site 2 of Collection-2 will be exclusively used for training and Site 3 will be exclusively used for testing.

In the next section, we describe the fusion system that we have developed based on a traditional Mamdani inference system [123] and using 4 features: EHDDT, EHDCT, $SignatureWidthDT$, $SignatureWidthCT$.

7.2 Fusion of Multiple Landmine Detection Algorithms Using Traditional Fuzzy Inference

Our goal is to design a system which accepts (as input) arbitrary sets of discrimination confidence values and additional contextual knowledge (such as $SignatureWidth$), and be able to: 1) derive a set of fuzzy rules from the available input knowledge; 2) learn associated output fuzzy sets; and 3) output a final confidence value representing the degree to which a GPR alarm should be considered as a target. To fulfill this functional requirement, first, we design two traditional fuzzy inference systems, based on Mamdani inference, and ANFIS. Next, we develop fusion systems using our proposed multiple instance fuzzy inference framework. In particular, we develop fusion methods based on our MI-Mamdani and MI-ANFIS and compare their performances to that of traditional fuzzy inference systems.

Given that traditional fuzzy systems cannot learn from ambiguously labeled data, information about correct target depths need to be provided (i.e., instances need to be labeled). To do so, for each positive bag we assign a positive label to the instances with the highest energy (energy can be computed by taking the sum of the absolute values of GPR signals within an instance), also a human expert is used to validate the labeling. On the other hand, our multiple instance framework does not require labels at the instance levels to be available and can learn from ambiguously labeled data. In the following, we show that even though our framework does not require instances' labels, it provided better results, this is because it uses all available information of a given bag to perform fuzzy reasoning.

7.2.1 Fusion of Multiple Landmine Detection Algorithms Using Mamdani Fuzzy Inference

To learn traditional Mamdani fuzzy rules, first, the input space is partitioned to identify local contexts. Second, input membership functions are learned based on the statistics of the partial confidence values of the input features (partial confidence values and auxiliary features) within each context. Third, output membership functions are generated by considering the distributions of targets and non-targets within each context. Finally, the

input and output membership functions are combined into a Mamdani-type fuzzy inference system. The output of the learning process is a fuzzy rule base (FRB) adapted to different contexts.

For this task, we have generated $N = 3050$ training observations, from Collection-1, with desired output $\mathcal{T} = \{t_j | j = 1, \ldots, N\}$ that correspond to instances processed by different discrimination algorithms and/or background features extractors (EHDDT, EHDCT, $SignatureWidthDT$, $SignatureWidthCT$). From each non-target alarm, we selected 5 instances at an equal sampling interval, and from each target alarm we selected 2 instances intuitively selected based on the highest value of the combined EHDDT and EHDCT confidence values. An expert is used to label the data.

The partial confidence values of a given discriminator d are denoted by $\mathcal{Y}_d = \{y_{dj} | j = 1, \ldots, N\}$. Each auxiliary feature e in denoted by $\mathcal{B}_e = \{b_{ej} | j = 1, \ldots, N\}$. The D ($D = 2$) discriminators and E ($E = 2$) background features are then concatenated to generate one global descriptor for each observation:

$$\mathcal{X} = (\bigcup_{d=1}^{D} \mathcal{Y}_d) \cup (\bigcup_{e=1}^{E} \mathcal{B}_e) = \{\mathbf{x}_j = [y_{1j}, \ldots, y_{Dj}, b_{1j}, \ldots, b_{Ej}] | j = 1, \ldots, N\}. \tag{7.8}$$

To simplify notation, we will use x_{ij} to donate either y_{ij} or b_{ij}, and rewrite (7.8) using:

$$\mathcal{X} = (\bigcup_{d=1}^{D} \mathcal{Y}_d) \cup (\bigcup_{e=1}^{E} \mathcal{B}_e) = \left\{\mathbf{x}_j = [x_{1j}, \ldots, x_{(K=D+E)j}] | j = 1, \ldots, N\right\}. \tag{7.9}$$

The proposed fusion system can be expressed by means of a fuzzy rule base composed of a union of $if-then$ fuzzy rules. A typical Mamdani-style fuzzy rule, $\mathcal{R}^i$ has the following form:

$$\mathcal{R}^i: \text{ If } x_1 \text{ is } M_1^i \text{ and } x_2 \text{ is } M_2^i, \ldots, \text{and } x_K \text{ is } M_K^i, \text{ then } o^i \text{ is } C^i. \tag{7.10}$$

In (7.10) $\mathcal{R}^i, i = 1, 2, \ldots, r$, is the ith fuzzy rule, M_j^i is a fuzzy set associated with the jth input, and C^i is the fuzzy set describing the output of the ith rule. The FRB is the union of all rules:

$$FRB = \bigcup_{i=1}^{r} \mathcal{R}^i. \tag{7.11}$$

The fuzzy sets in (7.10) consist of linguistic labels characterized by parameterized membership functions M^i and C^i. We use trapezoidal membership functions that can be completely determined by four scalar parameters l, m, h, and u, where l and u locate the "feet" (support) of the trapezoid and the parameters m and h locate the "shoulders" (core). Formally,

$$M_k^i(x_k) = max(min(\frac{x_k - l_k^i}{m_k^i - l_k^i}, 1, \frac{u_k^i - x_k}{u_k^i - h_k^i}), 0). \tag{7.12}$$

For the rules' outputs (i.e. C^i), we use Gaussian membership functions:

$$C^i(y) = e^{\frac{-(y-c_o^i)^2}{2{\sigma_o^i}^2}}, \tag{7.13}$$

where c_o^i and σ_o^i are the mean and variance of the gaussian function.

Identifying the FRB in (7.11) is equivalent to identifying its underlying parameters:

1. Premise parameters: $\mathcal{P} = \left\{l_k^i, m_k^i, h_k^i, u_k^i \mid i = 1, 2, \ldots, r; k = 1, 2, \ldots, K\right\}$; and
2. Consequent parameters: $\mathcal{C} = \left\{c_o^i, \sigma_o^i \mid i = 1, 2, \ldots, r\right\}$

To identify the premise parameters we first cluster the N training observations along each dimension $k = 1 \ldots K$ into r_i clusters using the K-means algorithm [124]. The K-means returns a list of clusters' centers(C) and the set of points associated with each cluster. Then, the Premise parameters are derived from the clusters' centers and widths (c_k^i, σ_k^i) along each input dimension by transforming the Gaussian membership function, defined by the cluster's center and width (c_k^i, σ_k^i) to a trapezoidal one using:

$$\begin{cases} l_j^i = c_j^i - \alpha \times \sigma_j^i \\ m_j^i = c_j^i - \beta \times \sigma_j^i \\ h_j^i = c_j^i + \beta \times \sigma_j^i \\ u_j^i = c_j^i + \alpha \times \sigma_j^i \end{cases} \textit{such that } \alpha \geq \beta \tag{7.14}$$

Trapezoidal membership functions have lager cores and are more suitable for fuzzification of discriminators' confidence values. In (7.14), the parameters α and β control the width of the core and support of the trapezoidal functions.

To learn the consequent parameters we count the number of target and non-target instances within each region of the input space (i.e., clusters generated by K-means in the previous step). Then, the proportion of target instances is used as the mean of the output membership function (i.e., c_o^i) and the width is fixed. Using this assignment, regions dominated by target instances will have an output closer to 1, while regions dominated by non-targets will have an output closer to 0.

Once the FRB is identified, we use the inference process described in Section 2.3.1. For each new depth instance, we start by fuzzification of the input. The fuzzification role is to determine the membership degree of each input dimension in the rules' input fuzzy sets. After this step the implication is executed using the product as a joint (and) operator, this will lead to some rules being activated with different degrees. Then, the rules' outputs are aggregated, and defuzification is executed to produce a crisp confidence value indicating the degree to which the instance should be considered as a target. To test a GPR alarm, each depth instance is fed to the system and its partial confidence value is computed . Then a final confidence value is assigned to the alarm by taking the average of the top 3 instances with largest confidence values [117].

7.2.1.1 Rule Generation

First, the confidence values of the EHDDT and EHDCT discriminators as well as SignatureWidthDT and SignatureWidthCT background features are extracted. To partition the input space using the K-means algorithm, the EHDDT and EHDCT were divided into 3 fuzzy sets as following: Low, Medium , and High. Whereas the SignatureWidthDT and SignatureWidthCT were quantized into Narrow, Medium, and Wide fuzzy sets. For the Gaussian output membership function, we set σ to 0.05. This partitioning generates a total of 81 clusters. We discard clusters that have few samples (< 10). This results in 21

rules.

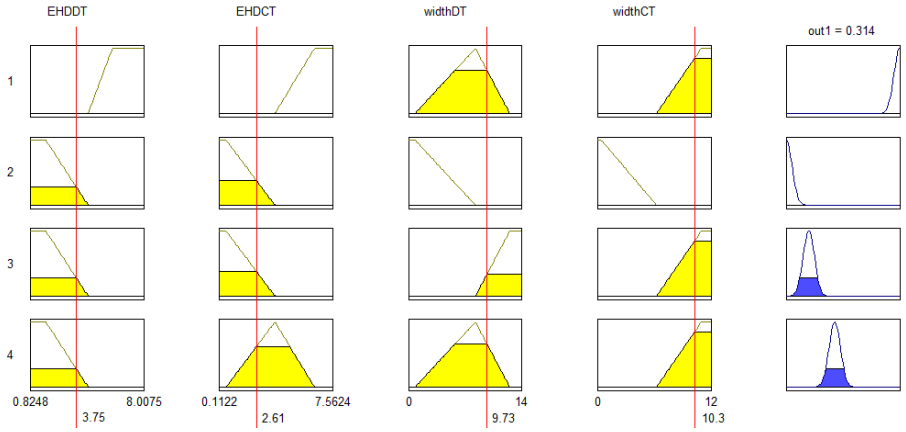


Figure 7.7: Illustration of the generated Mamdani Fuzzy Rule Base (FRB), showing 4 of the 21 rules.

The rules obtained are intuitive and easily interpretable. For instance in Figure 7.7, we display 4 of the 21 rules, when the input $= [EHDDT = 3.75,\ EHDCT = 2.61,\ SignatureWidthDT = 9.73,\ SignatureWidthDT = 10.3]$. Rule 1 and 2 state the following:

$\mathcal{R}^1$ If $EHDDT$ is $High$ and $EHDCT$ is Low and $SignatureWidthDT$ is $Medium$ and $SignatureWidthCT$ is $Wide$ then o^1 is $High$. (7.15)

$\mathcal{R}^2$: If $EHDDT$ is Low and $EHDCT$ is Low and $SignatureWidthDT$ is $Narrow$ and $SignatureWidthCT$ is $Narrow$ then o^2 is Low. (7.16)

Rule 3 is identical to Rule 2 expect the $SignatureWidthDT$ and $SignatureWidthDT$ are now both high. As a result, the output increases from Low to Medium.

$\mathcal{R}^3$: If $EHDDT$ is Low and $EHDCT$ is Low and $SignatureWidthDT$ is $High$ and $SignatureWidthCT$ is $High$ then o^3 is $Medium$. (7.17)

7.2.2 Fusion of Multiple Landmine Detection Algorithms Using ANFIS

In the following, we outline a fusion method based on Adaptive Neuro Fuzzy Inference Systems (ANFIS) [125] capable of simultaneously identifying local contexts as well as

learning optimal weights for combining local expert discriminators.

Given the same training data used to train the previous fusion method (i.e., $\mathcal{X}$ and $\mathcal{T}$), we use ANFIS to iteratively achieve: 1) structure identification, which relates to determining the number of fuzzy if-then rules and an optimal partition of the input space, and 2) parameter identification, which involves learning of the optimal partitions (contexts) and combination weights. To learn the rules, first, the input space is partitioned to identify local contexts. Second, input membership functions are learned based on the statistics of the partial confidence values of the individual discriminators as well as additional background information within each context. Third, the output parameters of the rules are initialized using a least squares estimator (LSE). Finally, the input and output membership functions are combined into a Sugeno-type fuzzy inference system. The resulting ANFIS system is then trained using a hybrid learning algorithm [125]. The output of the learning process is a fuzzy rule base adapted for different contexts.

As detailed in Section 2.3.3, ANFIS can be expressed by means of a fuzzy rule base (FRB) composed of a union of Sugeno type if-then fuzzy rules. A typical Sugeno fuzzy rule has the following form:

$$\mathcal{R}^i: \text{ If } x_1 \text{ is } M_1^i \text{ and } x_2 \text{ is } M_2^i, \ldots, x_K \text{ is } M_K^i, \text{ then} o^i = a_1^i \times x_1 + a_2^i \times x_2 + \ldots + a_K^i \times x_K + b^i. \tag{7.18}$$

Where $\mathcal{R}^i, i = 1, 2, \ldots, r$, denotes the ith fuzzy rule. M_j^i is a fuzzy set associated with the jth fusion input, a_j^i is a weight assigned to the jth discriminator or background feature, and b^i is a constant. As before, the FRB is then obtained by taking the union of all rules:

$$\mathcal{FRB} = \bigcup_{i=1}^{r} \mathcal{R}^i. \tag{7.19}$$

In this fusion system, we use gaussian membership functions that can be completely determined by two scalar parameters c and σ, the center and width of the gaussian function.

$$M_j^i(x_j) = \exp(-\frac{(x_j - c_j^i)^2}{2 \times \sigma_j^i}). \tag{7.20}$$

ANFIS parameters are then,

1. Premise parameters:
   $\mathcal{P} = \left\{c_j^i, \sigma_j^i | i = 1, 2, \ldots, r; j = 1, 2, \ldots, K\right\}$; and

2. Consequent parameters:
   $\mathcal{C} = \left\{a_j^i, b^i | i = 1, 2, \ldots, r; j = 1, 2, \ldots, K\right\}$

To identify the premise parameters (i.e. the parameters of the membership functions M^i), we cluster the N training observations into r clusters using the FCM algorithm [60]. FCM returns a list of clusters' centers(C) and a partition matrix ($\mathbf{U}$). The premise parameter σs are then derived from clusters's centers and widths using (2.57). To initialize the rules' output parameters we use an ordinary least squares estimator as defined in (2.58).

Once the structure of the network is defined and initialized, we continue with the learning process that yields a network with a fine-tuned membership functions. Thus, fine-tuned contexts. Each rule can be viewed as a context with its associated optimal combination weights (consequent parameters). When testing, an instance will activate certain rules (contexts) to certain degrees and the network output will be the weighted average off all rules outputs combined.

As before, to test a GPR alarm, each depth instance is fed through the ANFIS network and its partial confidence value is computed as the defuzzification of all rules outputs. Then a final confidence value is assigned to the alarm by taking the average of the top 3 instances with largest confidence values.

7.2.2.1 Rule Generation

First, the conffidence values of the EHDDT and EHDCT discriminators as well as SignatureWidthDT and SignatureWidthCT auxiliary features are extracted. Then, the input space is partitioned using the FCM algorithm into 16 clusters. Next, ANFIS parameters are identified as described above. Finally, rules's parameters are fine-tuned using the hybrid learning algorithm outlined in Section 2.3.3. The output of this process is a rule base optimized for the fusion of multiple landmine detection algorithms. Figure 7.8 is an

illustration of 2 of the 16 learned rules, when the input = $[EHDDT = 3.75,\ EHDCT = 2.61,\ SignatureWidthDT = 9.73,\ SignatureWidthDT = 10.3]$. Rule 1 and 2 state the following:

$\mathcal{R}^1$ If $EHDDT$ is Low and $EHDCT$ is Low and $SignatureWidthDT$ is $Medium$

and $SignatureWidthCT$ is $Narrow$ then

$$o^1 = 15.3 \times EHDDT + 0.1358 \times EHDCT - 7.681 \times SignatureWidthDT + 3.921 \tag{7.21}$$

$$\times SignatureWidthCT - 116.8 \tag{7.22}$$

$\mathcal{R}^1$ If $EHDDT$ is $Medium$ and $EHDCT$ is $Medium$ and $SignatureWidthDT$ is $Wide$

and $SignatureWidthCT$ is $Wide$ then

$$o^1 = -1.594 \times EHDDT - 0.05 \times EHDCT + 0.0058 \times SignatureWidthDT - 1.686 \tag{7.23}$$

$$\times SignatureWidthCT + 45.8 \tag{7.24}$$

Figure 7.8: Illustration of the generated ANFIS Fuzzy Rule Base (FRB), showing 2 of the 16 rules.

ANIFS rules (Sugeno rules in general) are not as interpretable as Mamdani rules. However, they are more optimized for the desired fusion application and yield better results as shown in the next section.

7.2.3 Results

Figure 7.9 displays a scatter plot of EHDDT vs. EHDCT. As it can be seen, the two detectors are consistent for most targets and false alarms (FA). However, there are several cases where the confidence values are not consistent. For instance, region $R2$ includes

samples where the EHDDT discriminator performed better than the EHDCT discriminator. Similarly, EHDCT can help identify targets (e.g., within $R1$) that may be missed by EHDDT. In some cases were both discriminators agree on low confidence values, SignatureWidth auxiliary features can help increase the final confidence value as shown in Mamdani Rule 3 in Figure 7.7.

We compare the performance of the proposed fusion methods (i.e., Mamdani and ANIFS) to the individual discriminators and two global fusion methods: (i) the first global fusion method performs the geometric mean of EHDDT and EHDCT, (ii) the second global fusion method performs the geometric mean of EHDDT, EHDCT, $SignatureWidthDT$, and $SignatureWidthCT$.

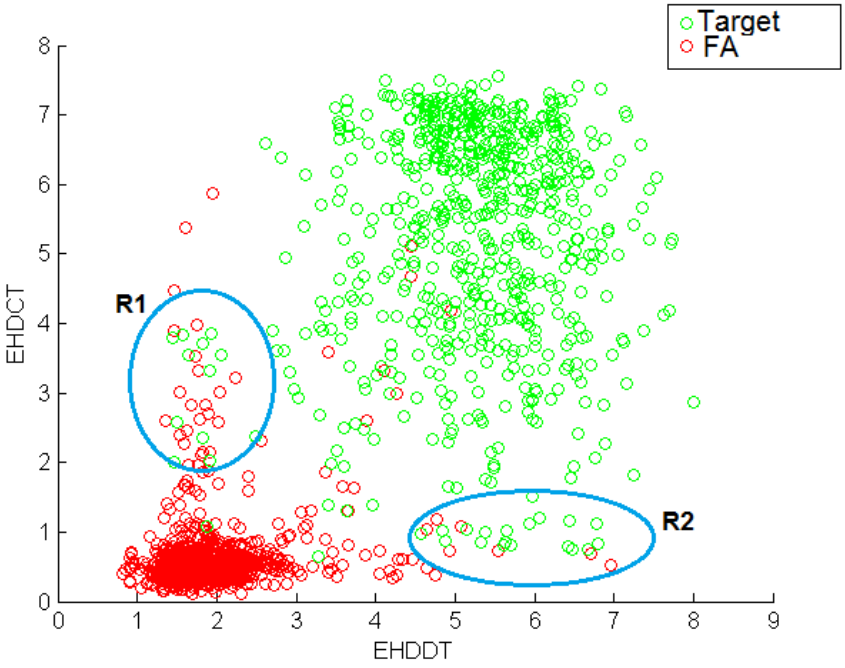


Figure 7.9: Comparison of the performances of EHDDT and EHDCT discriminators.

The individual discriminators and the proposed fusion were trained and tested using 10-folds cross validation. Figure 7.10 displays the ROC's of all methods. As it can be seen, the proposed Mamdani fuzzy fusion method outperformed the two global fusion methods and all of the individual discriminators. This is due mainly to the localized approach used by our system to better define local contexts by means of fuzzy rules resulting in a better

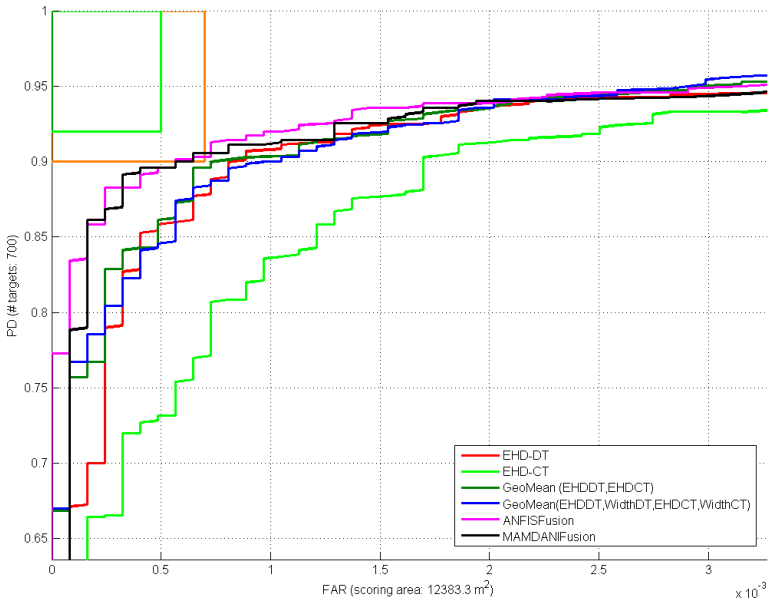


Figure 7.10: Comparison of the individual discriminators and the proposed fuzzy fusion method.

combination of the inputs. However, ANFIS gave the best overall performance. In addition to being a localized approach, ANFIS jointly identify local contexts and learns optimal weights for combining local discriminators.

7.3 Fusion of Multiple Landmine Detection Algorithms Using Multiple Instance Fuzzy Inference

Discrimination algorithms detect target candidates only in two-dimensions (down-track and cross-track position). Thus, there is uncertainty in the depth estimation of the targets that can affect both the training and testing phases of a fusion system. For training, it is very difficult to localize the objects depth automatically, and it is a very tedious process to do it manually. Similarly, during testing, it is not trivial to combine partial confidence values from the multiple windows.

The fusion training data are already grouped into bags. Each bag represents a GPR alarm and has instances extracted at multiple depths. Labels for the bags are available as binary ground truth: target/non-target (positive/negative). This formulation fits perfectly

TABLE 7.1

MI-Mamdani Parameters

Number of MI Rules	5
Number of Inputs	4
Membership functions	trapezoidal MFs
MFs' parameters	learned using FCMI
Number of Training bags	1030: 700 positive bags and 330 negative bags
Output parameters	singleton fuzzy set {1}.
Truth instances aggregation	average of top 3.

the MIL paradigm.

In the following, we develop two multiple instance fuzzy inference systems for the purpose of discriminators and auxiliary features fusion. The first system is based on the proposed MI-Mamdani inference, and the second system is based on the proposed MI-ANFIS. In addition we conduct two experiments: In first experiment, as the previous paragraph we use EHDDT, EHDCT, $SignatureWidthDT$, and $SignatureWidthCT$ to design a fusion system using MI-FISs. In the second experiment, we fuse the outputs of the EHDDT, EHDCT, FisherVectorDT and FisherVectorCT discriminators.

7.3.1 Fusion of Multiple Landmine Detection Algorithms Using MI-Mamdani

To learn an MI-Mamdani system from the training data (bags) for the purpose of fusion of discrimination algorithms, first, we apply the FCMI to extract concept points. Next, we generate multiple instance fuzzy rules from concept points as outlined in Section 4.3. Finally, the learned rules are combined into an MI-Mamdani multiple instance fuzzy inference system. We note that to aggregate the truth instances at the rules' level we used an Ordered Weighted Averaging Operator (OWA) that outputs the average of the top three highest truth instances.

After running FCMI, 5 concept points are identified and used to identify the parameters of 5 multiple instance fuzzy rules. The resulting rule base is illustrated in Figure 7.11. Table 7.1 summarizes the parameters used to identify the fusion rules.

The rules of our MI-Mamdani system describe concepts inferred from FCMI concept

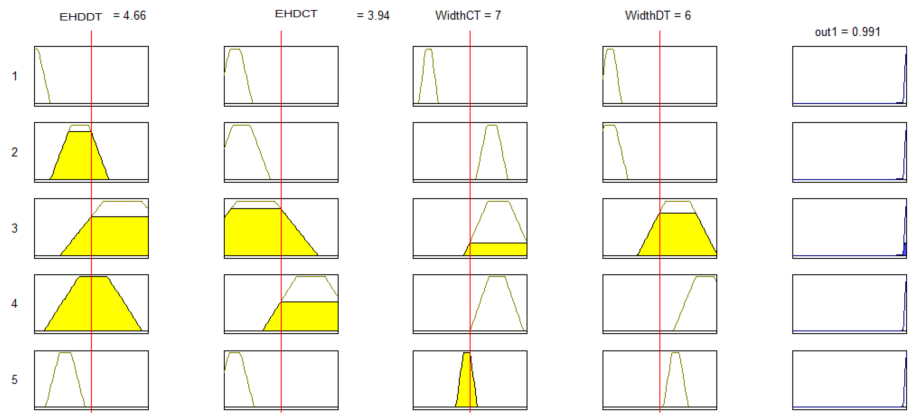


Figure 7.11: MI-Mamdami multiple instance fuzzy rules.

points. If a given target has an instance that can be described by any of the concepts it will lead to a high defuzzified output, and eventually to positive detection. However, non-targets should not have any instance within positive concepts and they will get assigned low output.

7.3.2 Fusion of Multiple Landmine Detection Algorithms Using MI-ANFIS

For this experiment, we construct a zero-order MI-ANFIS (constant consequent parameters) having 5 multiple instance rules, and employing Gaussian MFs to describe the input fuzzy sets. To initialize the system's parameters, first, we use the FCM algorithm to cluster the instances that belong to positive bags into 5 clusters, and we initialize the MFs' centers as the clusters' centers. Then, we set the standard deviations of the input MFs to a preset value of 1. Finally, we set the output parameters to 1. Table 7.2 summarizes all parameters used in training the MI-ANFIS.

After initialization, we run MI-ANFIS basic learning algorithm (Algorithm 5.2) to jointly learn a fuzzy description of the positive concepts as well as optimal rules' output. Figure 7.12 is a graphical representation of the 5 multiple instance rules prior to running the optimization process (dotted line curves) and the learned rules after training (continuous curves). Figure 7.13 plots the root mean squared error (RMSE) vs. the training epoch number. The fuzzy sets of the rules' antecedents describe the location and the extent of the positive concepts in the 4-D instance feature space. The rules' consequent values can

TABLE 7.2

MI-ANFIS Training Parameters

Number of MI Rules	5
Number of Inputs	4
Membership functions	Gaussian MFs
MFs' centers	initialized using FCM
MFs' standard deviations	preset to 1 (at epoch number 0)
Output parameters	constants = 1 ($\{b_0^i = 1\}_{i=1}^5$, at epoch number 0)
Number of Training bags	1030: 700 positive bags and 330 negative bags
Number of Training Epochs	150
Parameter α used in softmax function	2
Learning rate	0.1

be interpreted as an assessment of the "positivity" of each learned concept. For instance, the MI-ANFIS learned the following two positive concepts to describe targets:

$\mathcal{R}^1$: If $EHDDT$ is $Medium$ and $EHDCT$ is $Medium$ and $SignatureWidthDT$ is $High$ and $SignatureWidthCT$ is $High$ then $o^1 = 1.15$. (7.25)

$\mathcal{R}^2$: If $EHDDT$ is $Medium$ and $EHDCT$ is Low and $SignatureWidthDT$ is $High$ and $SignatureWidthCT$ is $High$ then $o^2 = 0.94$. (7.26)

7.3.3 Results

The proposed fusion methods were trained and tested using 10-fold cross validation on Collection-1. Figure 7.10 displays the ROC's (averaged over the 10 fold) of all methods. To provide a quantitative evaluation of the proposed multiple instance fuzzy inference fusion methods, we compare its performance to the previously presented fusion methods (Mamdani, ANIFS and the two global geometric mean methods). We also compare MI-Mamdani and MI-ANFIS performances to a naive MIL implementation of Mamdani (NaiveMamdani) and ANFIS (NaiveANFIS) where all instances from positive bags are considered positive and all instances from negative bags are considered negative.

Figure 7.14 displays the ROC's of all methods. Figure 7.15 shows the ROC's of MI-Mamdani, Mamdani, and NaiveMamdani fusion methods and the individual discriminators.

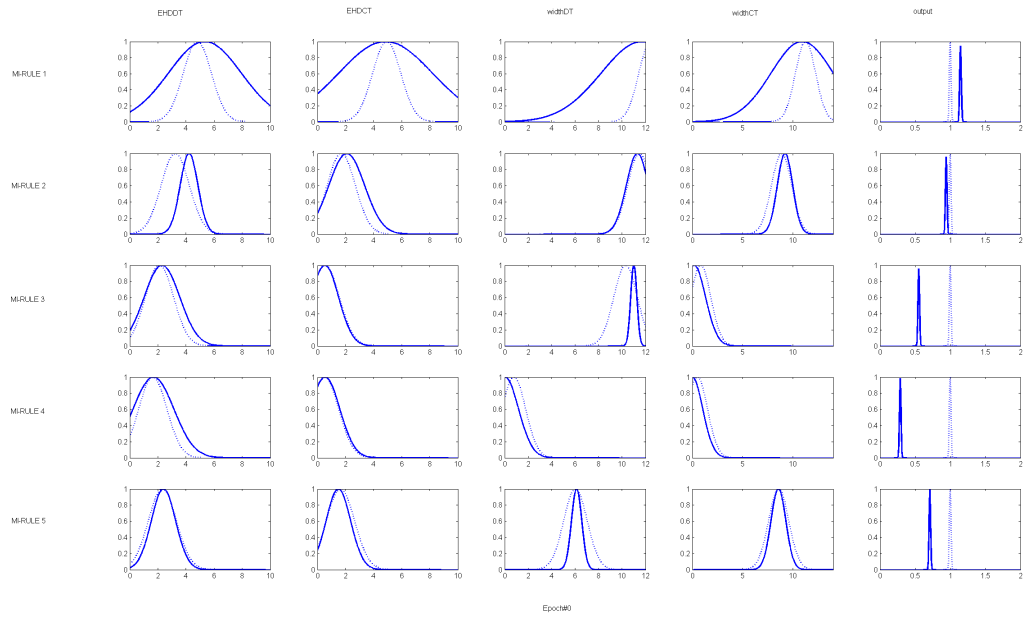


Figure 7.12: MI-ANIFS fusion rules before and after training (Dotted lines indicate the initial MFs).

Figure 7.16 displays ROC's of MI-ANFIS, ANFIS, and NaiveANFIS fusion methods as well as the individual discriminators. As it can be seen in Figure 7.14, MI-ANFIS performed better than the standard ANFIS on the large dataset, and as expected NaiveANFIS performed worse. MI-Mamdani outperformed the individual discriminators (EHDDT and EHDCT) and the NaiveMamdani fusion method. The standard Mamadani and ANFIS performed better at low FAR (False Alarms Rate), the reason behind this is that strong Mines are easy to identify manually and in this case, the ground truth helps. However, weaker mine signatures are not as easy to localize, so the truth may not be as accurate and can degrade the performance. Overall, MI-ANFIS outperformed all presented fusion approaches and the individual discriminators (EHDDT and EHDCT). This is due to the ability of MI-ANFIS to overcome labeling ambiguity by learning meaningful concepts.

In the second experiment, we used the same settings as before to train the two best performing algorithms, ANIFS and MI-ANFIS, to fuse the outputs of all discriminators, i.e., EHDDT, EHDCT, FisherVectorDT and FisherVectorCT. Fisher Vector based methods extract 60 instances per GPR alarm (bag), whereas EHD based methods extract 17 in-

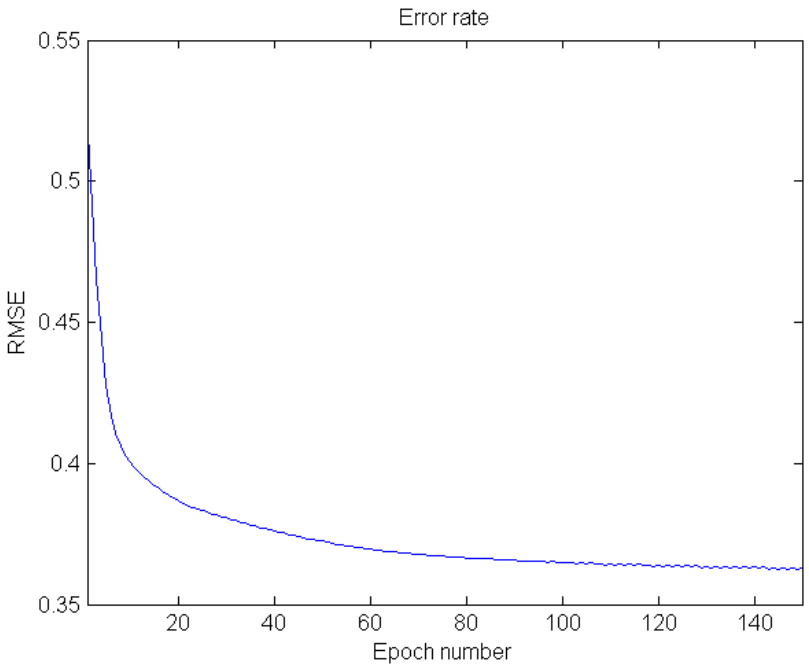


Figure 7.13: A plot of MI-ANFIS RMSE during 150 training epochs.

stances. Thus, Fisher Vector bags contain 60 instances and EHD bags contain 17 instances, all corresponding to the same GPR alarm. To be able to use the data within our multiple instance fusion system, we expanded the EHD instances from 17 to 60 (by taking averages of features extracted at different depths but corresponds to the same window used to generate the Fisher vector instances). The resulting bag has 60 4-D instances. Since the standard AFNIS cannot learn from partially labeled data, an expert is used to label all instances of all bags within the training data. We trained and tested all methods using 10-fold cross validation on the same data collection as before.

Figure 7.17 illustrates the resulting ROCs. As it can be seen, MI-ANFIS outperformed all discriminators and the standard ANFIS significantly. The performance boost over the individual discriminators is due to the substantial difference between the EHD and the Fisher Vector features; EHDDT and EHDCT features are derived from the standard MPEG-7 Edge Histogram Descriptors, whereas Fisher Vector is fundamentally based on aggregating SIFT features. Besides, EHD and Fisher Vector methods use different classifiers to assign confidence values to instances: EHD methods use a possibilistic KNN rule

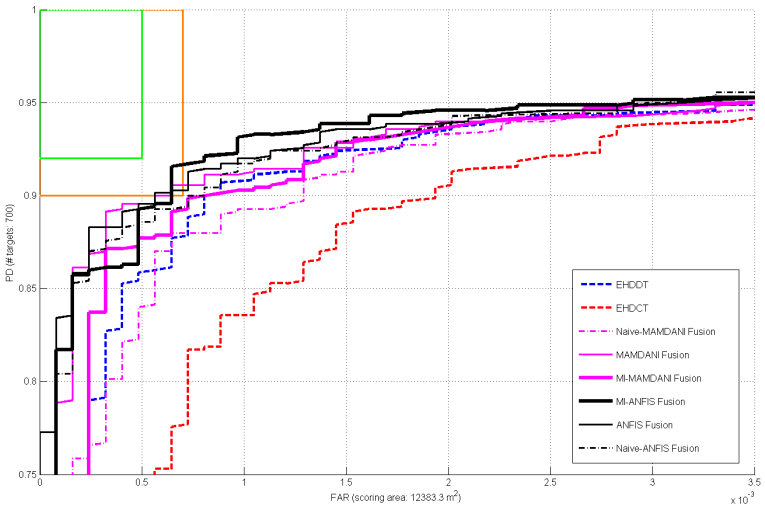


Figure 7.14: Comparison of the individual discriminators and all proposed fuzzy fusion methods.

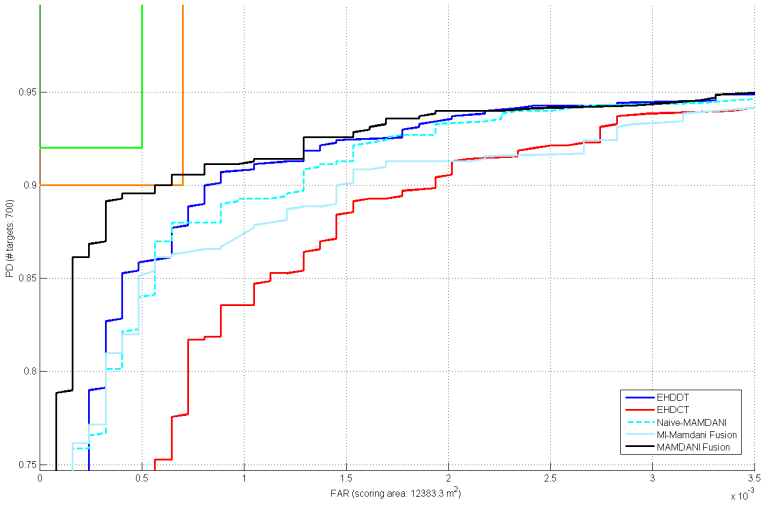


Figure 7.15: Comparison of the individual discriminators, the proposed MI-Mamdani , Mamdani, and NaiveMamdanifuzzy fusion methods.

and Fisher Vector methods use SVM. Thus, increasing the amount of information available to the fusion algorithms and per consequence increasing positive detections while lowering false alarms rates. On the other hand, the degraded performance of the standard ANIFS is linked to the degraded quality of the labeling of instances. The number of instances compared to the previous experiment has more than tripled (60 vs 17), making assigning correct

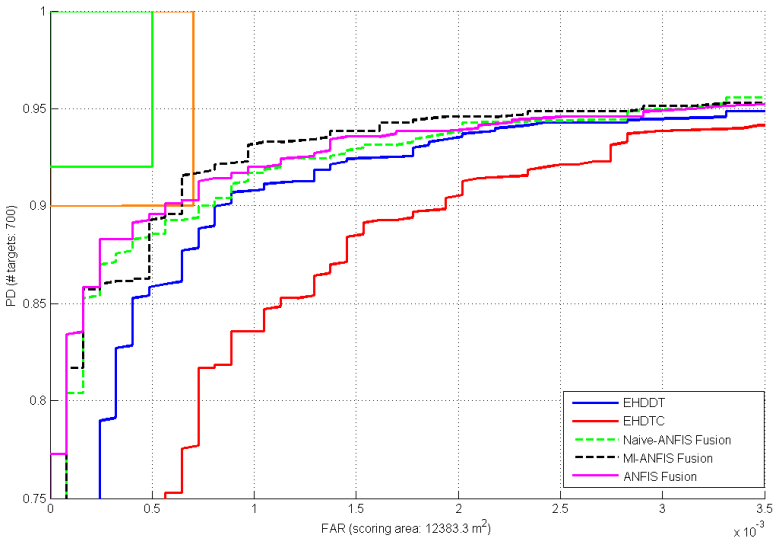


Figure 7.16: Comparison of the individual discriminators, the proposed MI-ANFIS , ANFIS, and NaiveANFIS fuzzy fusion methods.

labels by an expert an increasingly inaccurate process. Hence, the lower performance.

Thus far we have used cross validation to report on the performance of the proposed algorithms. Typically, cross validation is an adequate technique to predict the performance of a given model on unseen examples. However, for applications such as landmine detection, it is important as well to report the results of blind testing to assess how well a model performs on real world situations – outside of lab settings. In the following, we use Collection-2 to train and test our fusion methods. The collection is very large and was collected at three different sites. The main statistics are summarized in Table 7.3. Collection-1 was used to train ANIFS and MI-ANFIS to fuse the outputs of all discriminators, i.e., EHDDT, EHDCT, FisherVectorDT and FisherVectorCT. Collection-2 was exclusively used for testing. Figure 7.18 shows the blind test ROCs.

MI-ANIFS showed consistent performance in the blind test. It outperformed the individual discriminators and the standard ANFIS fusion. In spite of the fact that, an expert was used to label the training instances for ANFIS, the system could not overcome the ambiguity associated with locating the target depths correctly on the testing data.

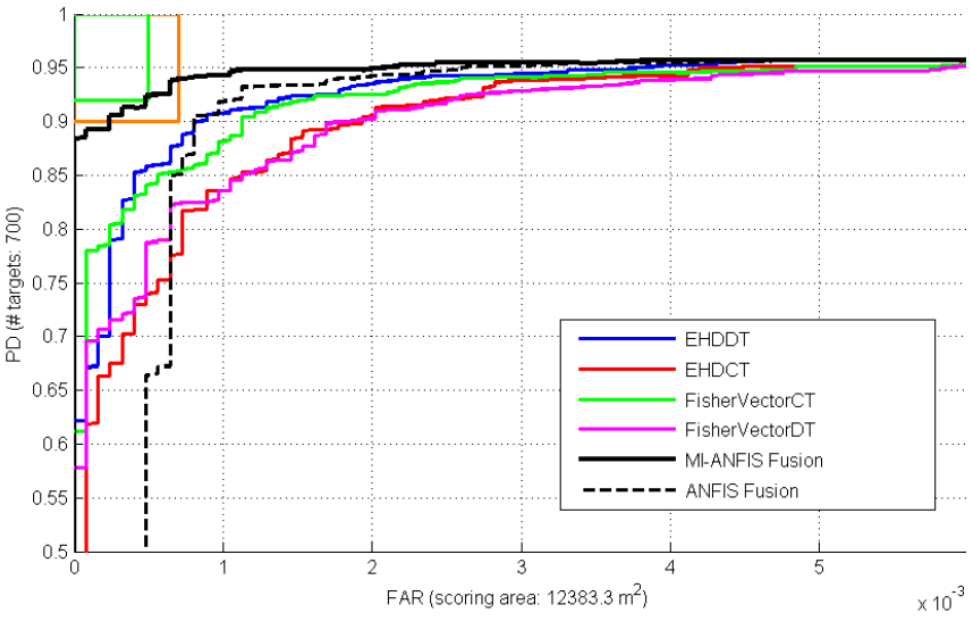


Figure 7.17: Comparison of all individual discriminators, ANFIS, and the proposed MI-ANFIS fuzzy fusion methods.

TABLE 7.3

Data Collections

Collection-2	**Site 1 & Site 2**	**Site-3**
Phase	Training	Testing
Total alarms	2366	4967
Mine encounters	789	1948
False alarms	1577	3018
Total number of Instances	141,960	297,960
Number of mine instances	47,340	116,880
Number of false alarms instances	94,620	181,080

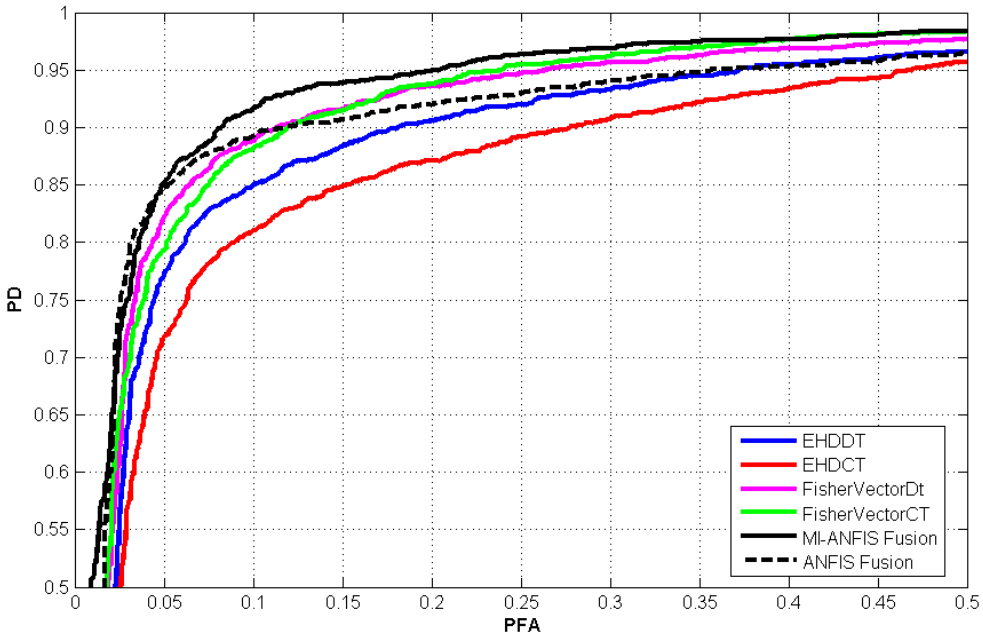


Figure 7.18: Comparison of the performances of all individual discriminators, ANFIS, and MI-ANFIS fuzzy fusion methods on the larger collection.

CHAPTER 8

CONCLUSIONS AND POTENTIAL FUTURE WORK

8.1 Conclusions

In this dissertation, we have introduced a new framework to accomplish fuzzy inference with multiple instance data. In multiple instance problems, the training data is ambiguously labeled. Instances are grouped into bags, labels of bags are known but not those of individual instances. Our work generalizes traditional fuzzy logic and fuzzy systems to enable reasoning with bags rather than single instances. The following sections summarize our contributions.

8.1.1 Multiple Instance Fuzzy Logic (MI-FL)

First, we have presented our generalization of fuzzy logic to enable fuzzy reasoning with bags of instances instead of a single instance at a time. In particular, we have introduced multiple instance variations of fuzzy propositions, fuzzy implication, fuzzy if-then rules, and fuzzy reasoning. These building blocks are then used to derive more complex fuzzy inference systems. Our formalization was derived using a thoroughly and abstract mathematical formulation.

Fuzzy logic is powerful at modeling knowledge uncertainty and measurements imprecision. More generally, it is one of the best frameworks to model vagueness. However, in addition to uncertainty and imprecision, there is a third vagueness concept that standard fuzzy logic does not address quiet well. This vagueness concept is due to the ambiguity that arises when data have multiple forms of expression as is the case for multiple instance problems. Our framework deals with ambiguity by introducing the novel concept of *truth instances*: when carrying reasoning using multiple instance fuzzy logic, a proposition will not only

have one degree of truth, it will have multiple degrees of truth, we call truth instances. Thus, effectively encoding the third vagueness component of ambiguity and increasing the expressive power of traditional fuzzy logic.

8.1.2 Multiple Instance Fuzzy Inference Systems (MI-FIS)

The traditional Mamdani and Sugeno inference systems outlined in chapter 2 are limited to reason with individual instances. First, the systems' inputs are an individual instance. Second, the rules describe fuzzy regions within the instances' space. Third, the outputs of the systems correspond to the fuzzy inference using the D dimensions of a single instance. Fourth, labels of the individual instances are required to learn the parameters of the systems. In this dissertation, we have used our multiple instance fuzzy logic framework to derived multiple instance Mamdani and Sugeno fuzzy inference styles capable of handling MIL problems effectively. In addition, we have presented a method to learn multiple instance rules from multiple instance data. First, we use the FCMI algorithm to extract target concept points in the instances' space. Target concepts are defined as regions in the instance space that maximize the density of instances from positive bags and minimizes the density of instances from negative bags. Next, the target concepts are transformed into multiple instance fuzzy rules. This approach is essentially based on intuition. Although premise and consequent parameters of the MI-FISs are learned from training data, the processes of identifying both set of parameters are independent.

8.1.3 Multiple Instance Adaptive Neuro-Fuzzy Inference System (MI-ANFIS)

Another major contribution of this dissertation is the MI-ANFIS, a novel neuro-fuzzy architecture that extends the standard Adaptive Neuro-Fuzzy Inference System (ANFIS) to reason with bags of instances. We first argued that the standard ANFIS can be used in the context of MIL only if bags are labeled at the instances level. Unfortunately, this process is tedious, ambiguous, subjective, and prone to errors.

The proposed generalization, MI-ANFIS, deals with ambiguity by using our proposed con-

cept of truth instances. Specifically, when carrying reasoning using a bag of instances at Layer 2 (Figure 5.1), a proposition will not only have one degree of truth, it will have multiple degrees of truth (r_{ij}). Thus, effectively encoding the third vagueness component of ambiguity and increasing the expressive power of the standard ANFIS. We have also developed a BackPropagation learning algorithm and showed that the proposed system is capable of learning meaningful concepts from ambiguously labeled data. Unlike MI-FIS, MI-ANFIS does not rely on any traditional MIL clustering algorithms and can learn simultaneously its rule base from data.

8.1.3.1 Rule Dropout

It is well-known fact that neural networks with large number of parameters are susceptible to overfitting. MI-ANFIS is no exception, particularly when using large number of multiple instance fuzzy rules and relatively small training datasets. In such scenario, some rules could co-adapt (memorize) to the training data and degrade the network ability to generalize to unseen examples. In situations where overfitting is imminent, we have proposed a regularization technique, we called Rule Dropout, and showed that it could be used to train MI-ANFIS systems with better generalization. Rule Dropout works by randomly dropping out few rules (with a fixed probability $1 - p$) before the presentation of a given training sample. During testing, all rules are used but the outputs are weighted by the probability p. Using a Rule Dropout strategy is approximatively equivalent to sampling and training 2^R (R is the number of rules) ensemble of MI-ANFIS networks. As a result, a more robust generalization can be achieved.

8.1.3.2 Multi-Class MI-ANFIS (MCMI-ANFIS)

Initially the MI-ANFIS has been proposed and developed for the two-class problem. We have also presented MCMI-ANFIS, a multiple class MI-ANIFS, that could be used to solve multiple class classification problems effectively. Most MIL methods deal with these type of problems by using a one versus all training pattern. While this extension

is straightforward and works with an arbitrary number of classes, it requires an extensive amount of preprocessing to relabel the data and generate networks. Moreover, doing so makes the data unbalanced and sampling should be used before training. In addition, some classes may share the same concepts, therefore, training different networks may lead to redundant rules being learned and wasting CPU cycles. The proposed MCMI-ANFIS minimizes a negative log likelihood criterion to learn all classes simultaneously reducing the possibility of learning redundant rules.

8.1.4 Validation

Using synthetic and benchmark datasets we showed that the proposed Multiple Instance Fuzzy Inference is comparable to state of the art MI machine learning algorithms. First, using a synthetic dataset with a 150 bags of which 100 are positive, we showed that our MI-Mamdani and MI-ANFIS can learn meaningful multiple instance fuzzy rules describing positive concepts. Next, using five benchmark datasets of different sizes (size varies between 92 bags and 200 bags), namely MUSK1, MUSK2, FOX, TIGER, and ELEPHANT datasets, we compared the performance of our framework to other 19 state of the art MIL algorithms. MI-ANFIS outperformed all other methods on the FOX and ELEPHANT benchmark, otherwise consistently ranked among the top-3 best algorithms. MI-MAMDANI performed better than 10 algorithms out of 19 tested on MUSK1, it also showed better performance than 7 algorithms out of 9 algorithms tested on FOX. However, MI-MAMDANI did not exhibit consistent performance on the rest of the benchmark datasets. Finally, using the COREL dataset (2000 bags) we applied our proposed MCMI-ANFIS to the problem of region-based image categorization and showed that our algorithm exhibited competitive performance to that of the state of the art.

Additionally, we have applied our proposed multiple instance fuzzy inference framework to fuse the output of multiple discrimination algorithms for the purpose of landmine detection using Ground Penetrating Radar. In this problem, discrimination algorithms detect target candidates only in two-dimensions (downtrack and cross-track position). Thus,

there is uncertainty in the depth estimation of the targets that can affect both the training and testing phases of a fusion system. For training, it is very difficult to localize the objects depth automatically, and it is a very tedious process to do it manually. Moreover, each GPR alarm is represented as a bag of instances extracted at multiple depths. Only labels for the bags are available as binary ground truth: target/non-target (positive/negative). Therefore, we have used our multiple instance fuzzy inference framework to solve this problem effectively. We have used two different GPR data collections to measure the performance of our algorithms on this problem. The first collection was used for 10-fold cross validation, whereas the second collection was used for blind testing. In both testing scenarios, MI-ANFIS outperformed all other fusion methods that we have proposed, namely the MI-Mamdani, the standard Mamdani, and the standard ANFIS inference systems.

8.2 Potential Future Work

Although our approach is fully developed and has shown promising results, there is still room for improvement. For instance, MI-ANFIS uses a fixed hyper-parameter α to control the behavior of the Softmax function in Layer 3 and Layer 5. In our experiments we used $\alpha = 1$ to replicate the conditions of the standard MIL assumption [36, 39]. Future research may include learning this hyper-parameter online, during training, which may offer more flexibility for other non standard applications of MI-ANFIS. Another hyper-parameter that could be learned, is the Rule Dropout rate p. Rule Dropout deemed important to solve large problems such as multiple class classification tasks. Learning this hyper-parameter could improve the overall generalization capability of our system. This task could be achieved either offline, before training using cross-validation on a subset of data, or online during training.

Future work may also include the evaluation of our framework on other domains such as computer audition [56] and text document classification [57]. In these applications, features are extracted from audio segments or text paragraphs, and labels are only available at the audio clip level or text document level, respectively, making them MIL problems.

REFERENCES

[1] Jan Ramon and Luc De Raedt, "Multi instance neural networks," 2000.

[2] Lotfi Asker Zadeh, *A theory of approximate reasoning (AR)*, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, 1977.

[3] Oscar Cordn, "A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems," *International Journal of Approximate Reasoning*, vol. 52, no. 6, pp. 894 – 913, 2011.

[4] Lotfi A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-3, no. 1, pp. 28–44, Jan 1973.

[5] E.H. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant," *Electrical Engineers, Proceedings of the Institution of*, vol. 121, no. 12, pp. 1585–1588, December 1974.

[6] R. Babuka and H.B. Verbruggen, "An overview of fuzzy modeling for control," *Control Engineering Practice*, vol. 4, no. 11, pp. 1593 – 1606, 1996.

[7] Chen-Wei Xu and Yong-Zai Lu, "Fuzzy model identification and self-learning for dynamic systems," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 17, no. 4, pp. 683–689, July 1987.

[8] Masaharu Mizumoto, "Fuzzy controls under various fuzzy reasoning methods," *Information Sciences*, vol. 45, no. 2, pp. 129 – 151, 1988.

[9] C.-C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. i," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 20, no. 2, pp. 404–418, Mar 1990.

[10] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling," *Fuzzy Systems, IEEE Transactions on*, vol. 1, no. 1, pp. 7–, Feb 1993.

[11] R.R. Yager and D.P. Filev, "Unified structure and parameter identification of fuzzy models," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, no. 4, pp. 1198–1205, Jul 1993.

[12] E.C. Tacker, "Modeling stabilization policies in financial systems," in *Decision and Control including the 16th Symposium on Adaptive Processes and A Special Symposium on Fuzzy Set Theory and Applications, 1977 IEEE Conference on*, Dec 1977, pp. 194–194.

[13] P.K. Singh, S. Bhanot, and H.K. Mohanta, "Optimized adaptive neuro-fuzzy inference system for ph control," in *Advanced Electronic Systems (ICAES), 2013 International Conference on*, Sept 2013, pp. 1–5.

[14] R. Jager, H. Verbruggen, and P.M. Bruijin, "Fuzzy inference in rule-based control systems," in *Intelligent Systems Engineering, 1992., First International Conference on (Conf. Publ. No. 360)*, Aug 1992, pp. 232–237.

[15] C.-C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. ii," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 20, no. 2, pp. 419–435, Mar 1990.

[16] Jorge Casillas, *Interpretability issues in fuzzy modeling*, vol. 128, Springer, 2003.

[17] Chih-Yi Chiu, Hsin-Chih Lin, and Shi-Nine Yang, "A fuzzy logic cbir system," in *Fuzzy Systems, 2003. FUZZ '03. The 12th IEEE International Conference on*, May 2003, vol. 2, pp. 1171–1176 vol.2.

[18] AA Othman, H.R. Tizhoosh, and F. Khalvati, "Efis 2014;evolving fuzzy image segmentation," *Fuzzy Systems, IEEE Transactions on*, vol. 22, no. 1, pp. 72–82, Feb 2014.

[19] S.N. Hajimirza and E. Izquierdo, "Gaze movement inference for implicit image annotation," in *Image Analysis for Multimedia Interactive Services (WIAMIS), 2010 11th International Workshop on*, April 2010, pp. 1–4.

[20] H.K. Kwan and L. Y. Cai, "Supervised fuzzy inference network for invariant pattern recognition," in *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium on*, 2000, vol. 2, pp. 850–854 vol.2.

[21] Olfa Nasraoui and Christopher Petenes, "Combining web usage mining and fuzzy inference for website personalization," *Proceedings of the WebKDD workshop*, pp. 37–46, 2003.

[22] M.N.M. Adnan, M.R. Chowdury, I Taz, T. Ahmed, and R.M. Rahman, "Content based news recommendation system based on fuzzy logic," in *Informatics, Electronics Vision (ICIEV), 2014 International Conference on*, May 2014, pp. 1–6.

[23] Amine Ben Khalifa and Hichem Frigui, "Fusion of multiple algorithms for detecting buried objects using fuzzy inference," *Proc. SPIE*, vol. 9072, pp. 90720V–90720V–10, 2014.

[24] Min-You Chen and D.A Linkens, "Rule-base self-generation and simplification for data-driven fuzzy models," in *Fuzzy Systems, 2001. The 10th IEEE International Conference on*, 2001, vol. 1, pp. 424–427.

[25] D.Z. Saletic, "On data-driven procedure for determining the number of rules in a takagi-sugeno fuzzy model," in *Computer as a Tool, 2005. EUROCON 2005.The International Conference on*, Nov 2005, vol. 2, pp. 1132–1135.

[26] P. Zikopoulos, D. deRoos, K. Parasuraman, T. Deutsch, J. Giles, and D. Corrigan, *Harness the Power of Big Data – The IBM Big Data Platform*, Mcgraw-Hill, 2012.

[27] I.B.M.P. Zikopoulos, C. Eaton, and P. Zikopoulos, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*, Mcgraw-hill, 2011.

[28] Yahoo, "flikr," `http://www.flikr.com/`, 2014.

[29] "Craigslist," `http://www.craigslist.org/`, 2014.

[30] Wei-Tek Tsai, Guanqiu Qi, and Yinong Chen, "Choosing cost-effective configuration in cloud storage," in *Autonomous Decentralized Systems (ISADS), 2013 IEEE Eleventh International Symposium on*, March 2013, pp. 1–8.

[31] A Sorokin and D. Forsyth, "Utility data annotation with amazon mechanical turk," in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, June 2008, pp. 1–8.

[32] A Torralba, B.C. Russell, and J. Yuen, "Labelme: Online image annotation and applications," *Proceedings of the IEEE*, vol. 98, no. 8, pp. 1467–1484, Aug 2010.

[33] Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum, "reCAPTCHA: Human-Based Character Recognition via Web Security Measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, Sept. 2008.

[34] Ian Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," in *ICLR2014*, 2014.

[35] Oded Maron, *Learning from ambiguity*, Ph.D. thesis, Massachusetts Institute of Technology, 1998.

[36] Thomas G. Dietterich, Richard H. Lathrop, and Toms Lozano-Prez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 12, pp. 31 – 71, 1997.

[37] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial intelligence*, vol. 89, no. 1, pp. 31–71, 1997.

[38] Chengcui Zhang, Xin Chen, and Wei-Bang Chen, "An online multiple instance learning system for semantic image retrieval," in *Multimedia Workshops, 2007. ISMW '07. Ninth IEEE International Symposium on*, Dec 2007, pp. 83–84.

[39] Oded Maron and Tomás Lozano-Pérez, "A framework for multiple-instance learning," in *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, Cambridge, MA, USA, 1998, NIPS '97, pp. 570–576, MIT Press.

[40] Andrew Karem and Hichem Frigui, "A multiple instance learning approach for landmine detection using ground penetrating radar," in *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*. IEEE, 2011, pp. 878–881.

[41] Rouhollah Rahmani and Sally A Goldman, "Missl: Multiple-instance semi-supervised learning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 705–712.

[42] Soumya Ray and Mark Craven, "Supervised versus multiple instance learning: An empirical comparison," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 697–704.

[43] Yixin Chen, Jinbo Bi, and J.Z. Wang, "Miles: Multiple-instance learning via embedded instance selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 12, pp. 1931–1947, Dec 2006.

[44] Changbo Yang, Ming Dong, and Farshad Fotouhi, "Region based image annotation through multiple-instance learning," in *Proceedings of the 13th annual ACM international conference on Multimedia*. ACM, 2005, pp. 435–438.

[45] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie, "Robust object tracking with online multiple instance learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 8, pp. 1619–1632, 2011.

[46] Lotfi A Zadeh, "Fuzzy logic = computing with words," *Fuzzy Systems, IEEE Transactions on*, vol. 4, no. 2, pp. 103–111, May 1996.

[47] Pier L Lanzi, Wolfgang Stolzmann, and Stewart W Wilson, *Learning classifier systems: from foundations to applications*, Number 1813. Springer, 2000.

[48] Ebrahim H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *Computers, IEEE Transactions on*, vol. C-26, no. 12, pp. 1182–1191, 1977.

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

[50] J-SR Jang, "Anfis: adaptive-network-based fuzzy inference system," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, no. 3, pp. 665–685, 1993.

[51] James Foulds and Eibe Frank, "A review of multi-instance learning assumptions," .

[52] S. Hofmann T. Andrews, "Multiple-instance learning via disjunctive programming boosting," *Advances in neural information processing systems.*, , no. 16, pp. 65–72, 2004.

[53] Boris Babenko, "Multiple instance learning: algorithms and applications," *View Article PubMed/NCBI Google Scholar*, 2008.

[54] Hong-Dong Li, Rajasree Menon, Gilbert S. Omenn, and Yuanfang Guan, "The emerging era of genomic data integration for analyzing splice isoform function," *Trends in Genetics*, vol. 30, no. 8, pp. 340 – 347, 2014.

[55] Cha Zhang, John C Platt, and Paul A Viola, "Multiple instance boosting for object detection," in *Advances in neural information processing systems*, 2005, pp. 1417–1424.

[56] Michael I Mandel and Daniel PW Ellis, "Multiple-instance learning for music information retrieval," in *ISMIR 2008: Proceedings of the 9th International Conference of Music Information Retrieval*. Drexel University, 2008, pp. 577–582.

[57] Stuart Andrews, Thomas Hofmann, and Ioannis Tsochantaridis, "Multiple instance learning with generalized support vector machines," in *AAAI/IAAI*, 2002, pp. 943–944.

[58] Qi Zhang and Sally A Goldman, "Em-dd: An improved multiple-instance learning technique," in *Advances in neural information processing systems*, 2001, pp. 1073–1080.

[59] Andrew Karem and Hichem Frigui, "Fuzzy clustering algorithm of multiple instance data (fcmi)," in *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*.

[60] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[61] Soumya Ray and David Page, "Multiple instance regression," in *ICML*, 2001, vol. 1, pp. 425–432.

[62] Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani, "1-norm support vector machines," *Advances in neural information processing systems*, vol. 16, no. 1, pp. 49–56, 2004.

[63] Zhi-Hua Zhou and Min-Ling Zhang, "Neural networks for multi-instance learning," *Proceedings of the International Conference on Intelligent Information Technology, Beijing, China*, pp. 455–459, 2002.

[64] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[65] Christopher M Bishop et al., "Neural networks for pattern recognition," 1995.

[66] Min-Ling Zhang and Zhi-Hua Zhou, "Adapting rbf neural networks to multi-instance learning," *Neural Process. Lett.*, vol. 23, no. 1, pp. 1–26, Feb. 2006.

[67] Jun Wang and Jean-Daniel Zucker, "Solving multiple-instance problem: A lazy learning approach," 2000.

[68] James Richard Foulds, *Learning instance weights in multi-instance learning*, Ph.D. thesis, The University of Waikato, 2008.

[69] L. A. Zadeh, "Fuzzy sets," *INFCON Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[70] J.S.R. Jang, C.T. Sun, and E. Mizutani, *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*, MATLAB curriculum series. Prentice Hall, 1997.

[71] Ronald R Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 18, no. 1, pp. 183–190, 1988.

[72] ZS Xu and QL Da, "The ordered weighted geometric averaging operators," *International Journal of Intelligent Systems*, vol. 17, no. 7, pp. 709–716, 2002.

[73] Michael Negnevitsky, *Artificial intelligence: a guide to intelligent systems*, Pearson Education, 2005.

[74] D. Ramot, M. Friedman, G. Langholz, and A. Kandel, "Complex fuzzy logic," *Fuzzy Systems, IEEE Transactions on*, vol. 11, no. 4, pp. 450–461, 2003.

[75] Patricia Melin, Jesus Soto, Oscar Castillo, and Jose Soria, "A new approach for time series prediction using ensembles of {ANFIS} models," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3494 – 3506, 2012.

[76] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Human-Computer Studies*, vol. 51, no. 2, pp. 135–147, 1975.

[77] H.O. Wang, K. Tanaka, and M.F. Griffin, "An approach to fuzzy control of nonlinear systems: stability and design issues," *Fuzzy Systems, IEEE Transactions on*, vol. 4, no. 1, pp. 14–23, Feb 1996.

[78] D. Dubois and H.M. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Mathematics in science and engineering. Academic Press, 1980.

[79] Ronald R Yager and Antoine Kelman, "Fusion of fuzzy information with considerations for compatibility, partial aggregation, and reinforcement," *International journal of approximate reasoning*, vol. 15, no. 2, pp. 93–122, 1996.

[80] Meimei Xia and Zeshui Xu, "Hesitant fuzzy information aggregation in decision making," *International Journal of Approximate Reasoning*, vol. 52, no. 3, pp. 395–407, 2011.

[81] Lotfi A Zadeh, "Is there a need for fuzzy logic?," *Information Sciences*, vol. 178, no. 13, pp. 2751–2779, 2008.

[82] Ronald R Yager, "On the theory of bags," *International Journal Of General System*, vol. 13, no. 1, pp. 23–37, 1986.

[83] Krassimir T Atanassov, "Intuitionistic fuzzy sets," *Fuzzy sets and Systems*, vol. 20, no. 1, pp. 87–96, 1986.

[84] Vicenç Torra, "Hesitant fuzzy sets," *International Journal of Intelligent Systems*, vol. 25, no. 6, pp. 529–539, 2010.

[85] D. Ramot, R. Milo, M. Friedman, and A. Kandel, "Complex fuzzy sets," *Fuzzy Systems, IEEE Transactions on*, vol. 10, no. 2, pp. 171–186, Apr 2002.

[86] S. Dick, "Toward complex fuzzy logic," *Fuzzy Systems, IEEE Transactions on*, vol. 13, no. 3, pp. 405–414, 2005.

[87] Cheng Guosheng and Yang Jianwei, "Complex fuzzy reasoning schemes," in *Information and Computing (ICIC), 2010 Third International Conference on*, vol. 3, pp. 29–32.

[88] V. G. Kaburlasos and A. Kehagias, "Fuzzy inference system (fis) extensions based on lattice theory," *Fuzzy Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[89] A. Mahnot and M. Popescu, "Fumil-fuzzy multiple instance learning for early illness recognition in older adults," in *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, pp. 1–5.

[90] Jaume Casasnovas and Gaspar Mayor, "Discrete t-norms and operations on extended multisets," *Fuzzy sets and Systems*, vol. 159, no. 10, pp. 1165–1177, 2008.

[91] Kandel A. Tamir D.E, "Axiomatic theory of complex fuzzy logic and complex fuzzy classes," *Int. J. Comput. Commun. Control International Journal of Computers, Communications and Control*, vol. 6, no. 3, pp. 562–576, 2011.

[92] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, "On the difficulty of training recurrent neural networks," *arXiv preprint arXiv:1211.5063*, 2012.

[93] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[94] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann, "Support vector machines for multiple-instance learning," in *Advances in neural information processing systems*, 2002, pp. 561–568.

[95] Min-Ling Zhang and Zhi-Hua Zhou, "Adapting rbf neural networks to multi-instance learning," *Neural Processing Letters*, vol. 23, no. 1, pp. 1–26, 2006.

[96] J.-S.R. Jang and C.-T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems," *Neural Networks, IEEE Transactions on*, vol. 4, no. 1, pp. 156–159, Jan 1993.

[97] Ma Yi-de, Liu Qing, and Qian Zhi-Bai, "Automated image segmentation using improved pcnn model based on cross-entropy," in *Intelligent Multimedia, Video and Speech Processing, 2004. Proceedings of 2004 International Symposium on*. IEEE, 2004, pp. 743–746.

[98] Haifeng Li, Tao Jiang, and Keshu Zhang, "Efficient and robust feature extraction by maximum margin criterion," *Neural Networks, IEEE Transactions on*, vol. 17, no. 1, pp. 157–165, 2006.

[99] Yan Li, David MJ Tax, Robert PW Duin, and Marco Loog, "Multiple-instance learning as a classifier combining problem," *Pattern Recognition*, vol. 46, no. 3, pp. 865–874, 2013.

[100] Yixin Chen, Jinbo Bi, and James Ze Wang, "Miles: Multiple-instance learning via embedded instance selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 12, pp. 1931–1947, 2006.

[101] Yixin Chen and James Z Wang, "Image categorization by learning and reasoning with regions," *The Journal of Machine Learning Research*, vol. 5, pp. 913–939, 2004.

[102] Jan Ramon and Luc De Raedt, "Multi instance neural networks," 2000.

[103] Zhi-Hua Zhou and Min-Ling Zhang, "Ensembles of multi-instance learners," in *Machine Learning: ECML 2003*, pp. 492–502. Springer, 2003.

[104] Abdelhamid Bouchachia, "Multiple instance learning with radial basis function neural networks," in *Neural Information Processing*. 2002, pp. 440–445, Springer.

[105] Zhi-Hua Zhou, Yu-Yin Sun, and Yu-Feng Li, "Multi-instance learning by treating instances as non-iid samples," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 1249–1256.

[106] Hua-Yan Wang, Qiang Yang, and Hongbin Zha, "Adaptive p-posterior mixture-model kernels for multiple instance learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1136–1143.

[107] Peter V Gehler and Olivier Chapelle, "Deterministic annealing for multiple-instance learning," in *International conference on artificial intelligence and statistics*, 2007, pp. 123–130.

[108] Christian Leistner, Amir Saffari, and Horst Bischof, "Miforests: multiple-instance learning with randomized trees," in *Computer Vision–ECCV 2010*, pp. 29–42. Springer, 2010.

[109] YaBill Morse, "The cambodia landmine museum," `http://www.cambodialandminemuseum.org/`, 2014.

[110] O. Missaoui, H. Frigui, and P. Gader, "Land-mine detection with ground-penetrating radar using multistream discrete hidden markov models," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 6, pp. 2080–2099, June 2011.

[111] A. Hamdi and H. Frigui, "Landmine detection using an ensemble of continuous hmms with multiple features," in *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*, July 2011, pp. 63–66.

[112] P.A. Torrione, K.D. Morton, R. Sakaguchi, and L.M. Collins, "Histograms of oriented gradients for landmine detection in ground-penetrating radar data," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 3, pp. 1539–1550, March 2014.

[113] C.R. Ratto, K.D. Morton, L.M. Collins, and P.A. Torrione, "A hidden markov context model for gpr-based landmine detection incorporating stick-breaking priors," in *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*, July 2011, pp. 874–877.

[114] A.C. Gurbuz, "Determination of background distribution for ground-penetrating radar data," *Geoscience and Remote Sensing Letters, IEEE*, vol. 9, no. 4, pp. 544–548, July 2012.

[115] Kenneth J Hintz, "Snr improvements in niitek ground-penetrating radar," in *Defense and Security*. International Society for Optics and Photonics, 2004, pp. 399–408.

[116] T. R. Witten, "Present state of the art in ground-penetrating radars for mine detection," in *SPIE Conf Detection and Remediation Technologies for Mines and Minelike Targets III*, Orlando FL, 1998, pp. 576–586.

[117] Hichem Frigui and Paul Gader, "Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic k-nearest neighbor classifier," *Trans. Fuz Sys.*, vol. 17, no. 1, pp. 185–199, Feb. 2009.

[118] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek, "Image classification with the fisher vector: Theory and practice," *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[119] H. T. Kaskett and J. T. Broach, "Automatic mine detection algorithm using ground penetrating radar signatures," in *SPIE Conf. Detection and Remediation Technologies for Mines and Minelike Targets*, 1999, pp. 942–952.

[120] H. Frigui and P. D. Gader, "Detection and discrimination of land mines based on edge histogram descriptors and fuzzy k-nearest neighbors," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Vancouver, BC, Canada, July 2006.

[121] B. S. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG 7: Multimedia Content Description Language*, John Wiley, 2002.

[122] P.A. Torrione, C.S. Throckmorton, and L.M. Collins, "Performance of an adaptive feature-based processor for a wideband ground penetrating radar system," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 42, no. 2, pp. 644–658, April 2006.

[123] Ebrahim H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *Computers, IEEE Transactions on*, vol. C-26, no. 12, pp. 1182–1191, Dec 1977.

[124] J. A. Hartigan and M. A. Wong, "A K-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100–108, 1979.

[125] Jyh-Shing Roger Jang, Chuen-Tsai Sun, and Eiji Mizutani, "Neuro-fuzzy and soft computing-a computational approach to learning and machine intelligence [book review]," *Automatic Control, IEEE Transactions on*, vol. 42, no. 10, pp. 1482–1484, 1997.

CURRICULUM VITAE

NAME: Amine Ben Khalifa

ADDRESS: Computer Engineering & Computer Science Department
Speed School of Engineering
University of Louisville
Louisville, KY 40292

EDUCATION:

Ph.D., Computer Science & Engineering
December 2015
University of Louisville, *Louisville, Kentucky*

B.Eng., Telecommunications Engineering
June 2009
Higher School of Communications of Tunis, *Tunis, Tunisia*

JOURNAL PUBLICATIONS:

1. **A.B. Khalifa** and H. Frigui, *"Multiple Instance Fuzzy Inference"*, IEEE Transactions on Fuzzy Systems (Under review).

CONFERENCE PUBLICATIONS:

1. **A.B. Khalifa** and H. Frigui, *"MI-ANFIS: A Multiple Instance Adaptive Neuro-Fuzzy Inference System"*, IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Istanbul, Turkey, August 2015.

2. **A.B. Khalifa** and H. Frigui, *"A Dataset For Vehicle Make And Model Recognition"*,

Third Workshop on Fine-Grained Visual Categorization (FGVC3), at CVPR. Boston, MA. June 2015.

3. **A.B. Khalifa** and H. Frigui, *"A Multiple Instance Neuro-Fuzzy Inference System For Fusion of Multiple Landmine Detection Algorithms"*, IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan Italy, July 2015.

4. **A.B. Khalifa** and H. Frigui, *"Fusion of Multiple Landmine Detection Algorithms Using an Adaptive Neuro Fuzzy Inference System"*, IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Quebec City Canada, July 2014.

5. **A.B. Khalifa** and H. Frigui, *"Fusion of Multiple Algorithms For Detecting Buried Objects Using Fuzzy Inference"*, SPIE Defense + Security International Society for Optics and Photonics, April 2014.

HONORS AND AWARDS:

1. IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) Travel Award , August 2015.

2. First Place at the University of Louisville E-Expo Graduate Research Competition, March 2015.

3. Golden Key International Honour Society Member, September 2011.

4. Higher School of Communications of Tunis Travel Award, June 2008.

5. Tunisian National Scholarship for Engineering Studies, September 2006.