2012

Identifying MicroRNA Precursors Using Linear Dimensionality Reduction With Explicit Feature Mapping

Navid Shakibapour Tabrizi
University of Windsor

IDENTIFYING MICRORNA PRECURSORS USING LINEAR DIMENSIONALITY REDUCTION WITH EXPLICIT FEATURE MAPPING

by
Navid Shakibapour Tabrizi

A Thesis
Submitted to the Faculty of Graduate Studies
through Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada
2012

IDENTIFYING MICRORNA PRECURSORS USING LINEAR DIMENSIONALITY REDUCTION WITH EXPLICIT FEATURE MAPPING

by

Navid Shakibapour Tabrizi

APPROVED BY:

Dr. Sévérien Nkurunziza, External Reader
Mathematics and Statistics

Dr. Alioune Ngom, Internal Reader
Computer Science

Dr. Luis Rueda, Advisor
Computer Science

Dr. Robin Gras, Chair of Defense
Computer Science

September, 2012

Declaration of Co-Authorship

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

MicroRNAs are a class of small RNAs of about 20 nucleotides long, which regulate cellular processes in animals and plants. Identifying microRNAs is one of the important tasks in microRNA and transcriptional studies. The main signal that is used for identifying these tiny molecules is the hairpin secondary structure of microRNA precursors.

In this research, I propose to use a linear dimensionality reduction(LDR)-based classifier to identify precursor microRNAs from both pseudo hairpins and other non-coding RNAs. LDR has been shown to be widely used in machine learning and pattern recognition problems. Due to the complexity of the data and nature of the problem, linear-based classifiers might not have an acceptable performance. Therefore, I propose to use explicit mapping to project data onto a higher dimensional space in order to increase class separability. Feature selection methods are used in order to reduce the complexity of the classifier and find relevant biological descriptors.

Dedication

To my parents

and

to Vida

Acknowledgements

I am pleased to express my deepest sense of gratitude to Dr. Luis Rueda. His informative guidance, continues support and worthwhile feedback helped me throughout the course of this thesis. It was an honour to be supervised by him and I will always be grateful to him.

I would like to thank Dr. Alioune Ngom and Dr. Sévérien Nkurunziza for spending their invaluable time. Their indispensable inputs and discussions greatly improved the quality of this thesis.

I owe my parents a deep debt of gratitude. Without their unconditional love and support, the very possibility of my ever succeeding in life would be doubtful.

Special thanks to my sister, Mahtab, for her consistent support.

I would also like to thank my friends Iman Rezaian, Gokul Vasudev, Manish Kumer Pandit and our Pattern Recognition and Bioinformatics lab members for their moral support.

At the end, I would like to conclude by extending my sincere appreciation and express my undying love to my dearest Vida. Thanks for making my life so beautiful.

Contents

List of Figures

List of Tables

List of Algorithms

Part I

Background

Chapter 1

Introduction

1.1 MicroRNA

MicroRNAs are single-stranded non-coding RNAs of about 19–22 nucleotides and are considered a class of post-transcriptional gene regulators that are identified in almost all metazoan genomes, including worms, flies, plants and mammals. The two founding members of the microRNA family, *lin-4* and *14*, were originally identified in *Caenorhabditis elegans* as genes that were necessary for temporal regulation of larval development [3]. Researchers believe that about one third of human genes are regulated by microRNAs [3]. MicroRNAs perform many cellular tasks in cells including controlling cell developmental timing, cell death and stem cell characterization [7]. In addition, many studies show that malfunction of microRNAs may have devastating impacts on cell life and may cause different types of cancer, heart disease and nervous system disorder [3]. Accordingly, identification of microRNA is an essential process in discovering microRNA functions and its role in cellular processes.

1.2 Classification

Linear dimensionality reduction (LDR) has been shown to be successfully used in pattern recognition and machine learning [33]. However, LDR methods might not be very efficient and powerful, especially when the data is highly complex and non-linear. For some LDR methods, kernel tricks were proposed to improve classification performance [22, 26, 27]. The kernel trick aims to implicitly map data that is not linearly separable to higher dimensions hoping that the data become linearly separable or at least more "separable" than in the original space. Mapping implicitly is not feasible in all cases due to the complexity of kernelizing some LDR methods. Instead, the data could be explicitly mapped onto the target space and then LDR can be used on the mapped data.

In this thesis, LDR combined with mapping data to higher dimensions is employed to classify precursor microRNAs from both pseudo hairpins and other non-coding RNAs. As discussed later, mapping data to higher dimensions can significantly improve the performance of the classifiers. In addition, using LDR can resolve the class imbalance problem as it takes the distribution of the data into consideration. As opposed to this, SVM only considers data around the support vectors. In addition, a feature selection method is proposed for selecting a subset of features instead of employing the whole feature vector, yielding very good results.

1.3 Motivation and Objectives

MicroRNAs are one on the mechanisms of gene regulation after the transcription process in prokaryotic cells as well as eukaryotic cells. It has been shown that these molecules are responsible for around 30% of gene regulation of the human genome. Also, it has been

well-studied that microRNAs are involved in many diseases. Therefore identifying microRNAs is very important for biologists. As of August 2012, 21,264 mature microRNAs have been identified in miRBase [17], which is a biological database that acts as an archive of microRNA sequences and annotations. Figure 1.1 shows the number of microRNA sequences which are published in different releases of *miRBase* database. Although a large number of microRNA sequences have been identified, a vast number of them are yet to be identified. This rapid growing is due to the different approaches which have been proposed for identifying these tiny molecules in recent years.

Initially, the only techniques for identifying microRNAs were experimental methods. Experimental methods use DNA cloning for microRNA identification. However, these methods suffer from low performance because of environmental conditions and low level expression of microRNAs. As an alternative, computational methods can discover microRNAs without conducting any experiment in wet laboratories. The main signal in identifying microRNAs is the hairpin secondary structure of the precursor microRNA (pre-microRNA). Computational methods rely on this fact for distinguishing these tiny molecules from other types of sequences.

Dozens of methods have been proposed in recent years, especially after 2005, which propose approaches for identifying microRNAs and many of these methods have acceptable performance. However, the database that they are using in training and testing process is not representative of the whole genome, and classifiers are built on a not-so-complete dataset. In addition, many of the proposed methods use a large number of features and they do not select a subset of features for reducing the complexity of the classifier as well as a road for biologist to interpret the limited number of features in the feature subset. Thus, using a feature selection algorithm which can find fewer is very important. In this research

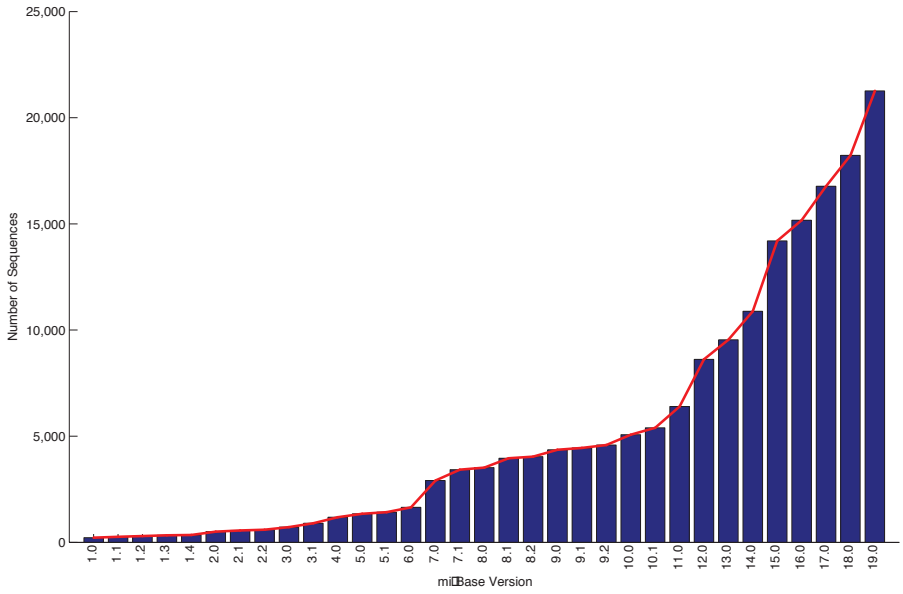

Figure 1.1: MiRBase database growth between December 2002 and August 2012.

work, a dataset is selected which both pseudo hairpins and other types of non-coding RNA sequences.

On the other hand, LDR methods are well-known and well-studied classifiers in machine learning and pattern recognition. These methods have shown very good performance in various applications. However, at this time there is no microRNA identification approach which is built based on LDR-based classifiers. In addition, the performance of the LDR-based classifier can be enhanced further by explicitly mapping the data to higher dimensional space, which again, at this time, have never been proposed to use with LDA-based classifiers. In addition, the feature selection algorithm that is chosen is based on wrapper methods which have advantages over filter methods in the sense that they uses the classifier itself for evaluating the performance comparing to using some evaluation metrics regardless of the classifier.

1.4 Problem

In this thesis, the problem that is being tackled is:

> Distinguishing microRNA precursor sequences from non microRNA precursors, pseudo hairpins and other non-coding RNAs sequences.

1.5 Contributions

The main contributions of this thesis are:

- Proposing a new classification scheme that combines LDR classification methods and microRNA features.
- Comparison of using explicitly mapped data fed into the classifier and using the original data. These methods have never been used with LDR classifiers.
- Utilizing feature selection algorithm for selecting fewer features.
- Designing and implementing a framework for automating and handling a large number of experiments and using a database server for storing the results.

In this thesis, we focus on identifying human pre-microRNAs from other molecules which are not human pre-microRNA. There are many methods available for this purpose but none have ever utilized the well-known LDR classifiers. In addition, I use wrapper feature selection methods for selecting a representative feature subset. In addition, the idea of mapping the data to higher dimensions in an explicit form have never been used in LDA classification algorithms but in this work we implement this idea and exploit it.

1.6 Thesis Organization

This thesis has six chapters. Chapter II presents information about microRNAs and different approaches which were introduced previously. Chapter III provides the required background about different pattern recognition concepts used in this research work. Chapter IV describes the proposed model, the dataset and the features which are used. In Chapter V, experimental results are presented and a comparison has been made with previously proposed methods as well as a discussion about the results. Finally, in Chapter VI, conclusions of this research work are presented in addition to few possibilities for future work.

Chapter 2

MicroRNAs

2.1 Gene Expression

The **central dogma of molecular biology** describes the way in which genetic information is transferred from DNAs to proteins: that is DNA $\rightarrow$ RNA $\rightarrow$ Protein (see Figure 2.1) [24]. The first process is called *transcription* in which an RNA molecule is synthesized from the information included in a section of DNA. The RNA molecule which is produced is called Messenger RNA (mRNA). The other process in which the protein molecule is produced from the mRNA is called *translation*.

Activation of an organism's genes depend on the cell's environment and needs of the cell in addition to the fact that genes might be expressed at different times. There are different mechanisms of gene regulation in *Prokaryotics* and *Eukaryotics*. However, one of

Figure 2.1: The central dogma of molecular biology.

the most important mechanisms is the *transcriptional regulation*. An enzyme called, RNA polymerase is responsible for the transcription regulation [24].

In addition to transcriptional regulation, gene regulation can be done at another step after transcription on the mRNA in a process called post-transcriptional process. A novel mechanism of gene regulation that happens after the transcription process is by tiny molecules called **microRNAs** (miRNA) [3].

2.2 MicroRNA

MicroRNAs are a large class of small non-coding RNAs that have post-transcriptional gene regulatory roles. The first two microRNAs that were discovered are *lin-4* and *let-7* of *Caenorhabditis elegans* [24]. It has been shown that these two microRNAs are involed in controlling the timing of larval development.

2.3 Biogenesis of MicroRNA

These tiny molecules repress translation of messenger RNAs (mRNA) into proteins in one of the two ways based on the complementarity level between the microRNA and their targets by binding into mRNAs. In the first method, microRNAs bind perfectly or almost perfectly to mRNA sequences and cause their cleavage by multiprotein RNA-induced-silencing complex (miRISC). This event causes degradation of the target mRNA. This mechanism of miRNA-mediated gene silencing can usually be found in plants, and in rare cases occurs in animals. However, in most animals, microRNA sequences use another mechanism for regulating the genes which does not lead to target mRNA's cleavage. These microRNAs imperfectly bind some sections within three prime untranslated regions (3'

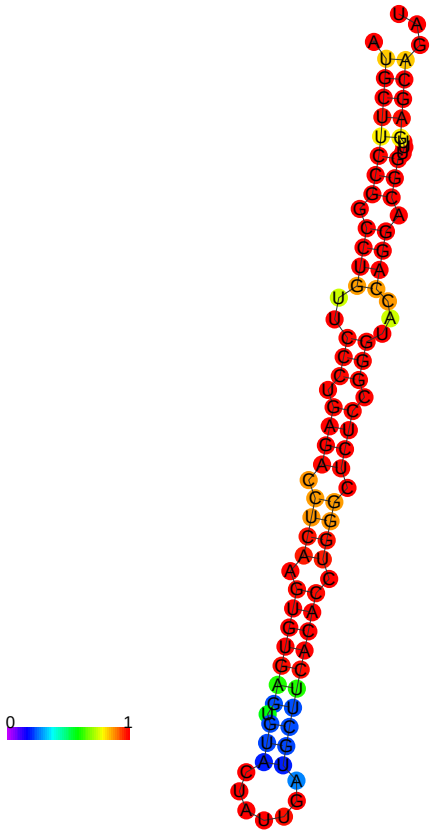

Figure 2.2: Secondary structure of *lin-4*.

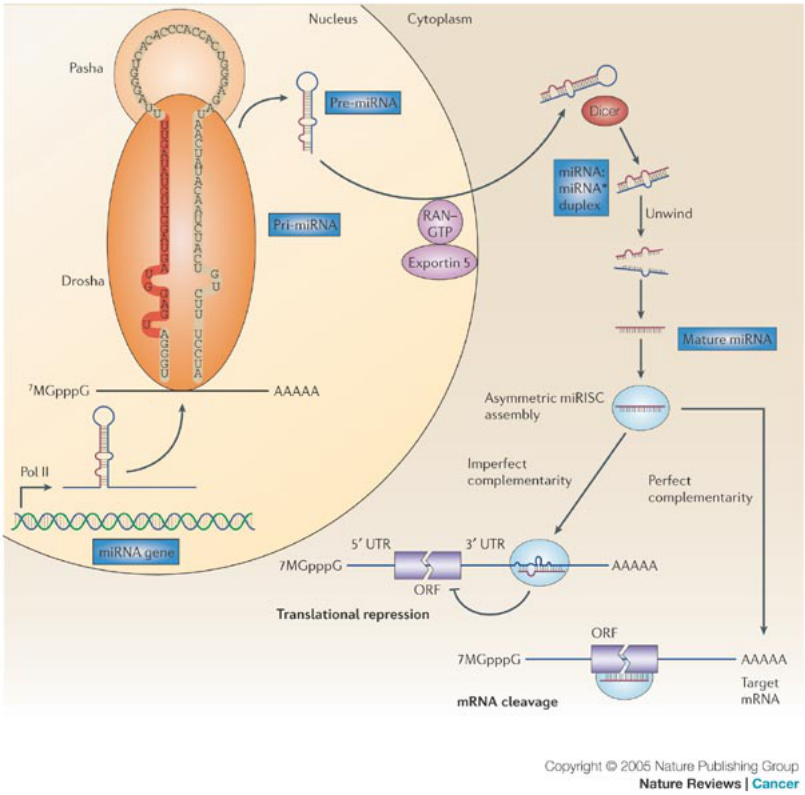

Figure 2.3: The biogenesis of microRNAs. Figure is taken from [12] by authors' permission.

UTR) of target mRNAs and repress the translation process of mRNAs into proteins. This is done through a RISC complex which is identical or highly similar to the one that is used in the first mechanism.

Since 2004, the biogenesis of miRNAs has been elucidated (Figure 2.3). Initially, microRNAs are transcribed by RNA polymerase II in nucleus as large RNA precursors that are called primary microRNAs (pri-microRNA) and are capped (MGpppG) and polyadenylated (AAAAA) [12]. In nucleus, pri-microRNA is processed by an enzyme called Drosha, and the double-stranded-RNA-binding protein, Pasha. A 70-nucleotide sequence called pre-microRNA is the product of this step which folds imperfectly into a hairpin secondary structure. The produced pre-microRNA is then exported into the cytoplasm by RAN–GTP

and exportin 5. After that, another enzyme known as Dicer, cuts the loop and generates a double-stranded RNA of about 22-nucleotides in length known as miRNA:miRNA* duplex. The duplex is detached and then one of the strands binds into miRISC complex. The mature microRNA strand that is incorporated miRISC complex can be used to negatively regulate its target genes [12].

2.4 MicroRNA Identification

Earlier, microRNAs were only identified by using experimental methods. The traditional experimental approaches to microRNA discovery are cloning and sequencing [10], and can detect novel microRNAs. Since microRNAs are usually expressed at low levels and depend on tissue and conditions of the cell, these methods may be unable to identify new microRNAs [3]. Recently high-throughput sequencing approaches, in particular, 454 sequencing, have become popular for discovering new microRNAs [13].

Another category of approaches for identifying microRNAs is computational methods. The main idea behind these methods is to analyze hairpin secondary structures of precursor microRNAs (pre-microRNA). Secondary structure of pre-microRNA allows researchers to propose computational methods that can distinguish these sequences from other sequences in the genome. The currently proposed computational methods for identifying microRNAs have been developed in two directions, comparative methods and non-comparative methods.

Comparative methods have been developed based on the study that shows microRNAs are highly conserved in related genomes [3]. Therefore, some methods use this property of microRNAs and introduce candidate microRNAs which fold into hairpin secondary structure and are conserved in related genomes that are considered as potential microRNA genes.

The other direction of computational methods does not rely on conservation characteristics of microRNAs. These methods are mainly based on effective and efficient identification of pre-microRNA among all other sequences which share similar secondary structure as pre-microRNA. As mentioned earlier, pre-microRNAs fold into stem-loop secondary structure. There are a few challenges which should be mentioned. The first one is that, there are thousands of other genome sequences which fold into hairpin secondary structure, called "pseudo hairpins" [5]. Second, many of other non-coding RNAs such as YRNAs, snRNAs and tRNAs fold into hairpin secondary structure as well. Therefore, the challenges are extracting set of features from sequences for forming a representative dataset and then applying a classification method that effectively identifies and distinguishes pre-microRNAs from other non-coding RNAs and pseudo hairpins.

In this research work, the focus is on developing a computational method for identifying pre-microRNAs based on approaches that do not rely on conservation information (non-comparative methods). In Section 2.5, many of these methods are reviewed.

2.5 Related Works

The idea of identifying microRNAs without relying on phylogenetic knowledge about the genome started with the works of [28], [34] and [39] and they all exerted great influences on the identifying microRNAs problem. Since 2005, many papers have been published in journals and conference proceedings and each of those made a contribution toward effectively identifying microRNAs. All papers in this area can be categorized based on the classification approach they have adopted and also based on the features they have introduced and employed for classification. There is a large degree of overlapping between the features and classification methods which have been used in the papers in this field. Thus,

defining a clear boundary between different approaches based on features and classification algorithms seems unfeasible. Therefore, in this work, papers have been introduced in a chronological order.

2.5.1 Focusing on Genome Regions Around Known MicroRNAs

Studies indicate that the total number of microRNA genes is larger than what had been identified prior to 2005. Therefore, microRNA gene discovery remained an important task in this field for understanding unknown regulation mechanisms. In particular, computational approaches had been found to be very useful for guiding experimental analysis. Then Sewer *et al.* [34] introduced a method for tackling this problem and called it *miRabela*.

In their work, the authors only focused on regions of the genome where known microRNAs had already been found. Then extracted sequences which can fold into stem-loops and have a robust secondary structure. The authors proposed 40 new features for characterizing sequential and secondary structure properties in relation to previously discovered microRNAs as well as negative samples as the input of the classification algorithm. They employed support vector machines for doing the classification. At the end, for guiding experimental investigations, the authors also developed a probabilistic statistical model which estimates the number of pre-microRNAs in a given genomic sequence.

Sewer *et al.* [34] used a dataset containing 178 positive samples as well as $5,395$ negative samples for classification. Positive samples were taken from the human *Rfam* repository. Negative samples are random sequence samples from tRNA, rRNA and mRNA.

Sewer *et al.* [34] claim that the model they have developed, recovers 71% of the positive pre-microRNA sequences with false positive rate of 3% and false negative rate of 29%. Also, they claim that their method can be used in guiding experimental approaches. In

addition, it is stated that the method can successfully identify microRNAs that are missed by previously developed methods.

2.5.2 Local Contiguous Structure-sequence Information of Stem-loops

Xue *et al.* [39] also felt the need for a method which can discern pre-microRNAs from other segments of sequences with similar hairpin structure (pseudo pre-microRNAs) and which does not rely on comparison with known microRNAs. Developing such methods is of some importance both for gaining more information about microRNAs and for identifying new microRNAs without comparing with previously discovered microRNAs. Xue *et al.* proposed 32 new triplet element features of "local contiguous structure-sequence information of stem-loops" for differentiating pre-microRNAs from pseudo pre-microRNAs.

For classification experiments, one training and two testing datasets were built and support vector machines (SVM) was used as the classification method. The training dataset, which was called TR–C, contained 163 and 168 human pre-microRNAs and pseudo pre-microRNAs, respectively. The first test dataset, called TE–C, included 30 human pre-microRNAs which did not overlap samples from TR-C and $1,000$ pseudo pre-microRNAs. The second test dataset, called CONSERVED–HAIRPIN, contained $2,444$ pseudo pre-microRNAs. Also the authors applied the classifier on 581 pre-microRNAs from 11 species other than human and called this dataset CROSS-SPECIES. They noted that pre-microRNAs with multiple loops had been filtered out from the datasets. In addition, they conducted an analysis on the "discriminant power of the different triplet elements".

Xue *et al.* claim that, their SVM classifier on the TE-C dataset, successfully classified 28 out of 30 human pre-microRNAs and 881 out of $1,000$ pseudo pre-microRNAs, which gives a sensitivity and specificity of 93.3% and 88.1%, respectively. Also, they claim that,

on the CONSERVED-HAIRPIN dataset, the classifier detected 2,174 out of 2,444 pseudo pre-microRNAs which gives a specificity of 89.0%. On the CROSS-SPECIES dataset, they claim that the classifier identified 90.9% of pre-microRNAs. In addition, they claim that due to high accuracy, pre-microRNAs and pseudo pre-microRNAs are distinct with respect to proposed triplet element features, despite the fact that they have similar hairpin structure. They also assert that the classifier is also capable of identifying microRNAs of other species rather than human even though it was trained with human pre-microRNAs and shows the proposed features might reflect a quality which is same for all species.

2.5.3 One-class Compared Two-class Classifiers

Machine learning algorithms which do not rely on defining the negative class and only depend on the positive class have been getting more attention from researchers in bioinformatics. That is because generation of the negative class might be problematic and not representative enough. Yousef *et al.* [40] used a one-class approach for finding microRNAs.

The authors criticize previously proposed approaches for relying on generation of an artificial negative class since if the negative class is not generated properly, performance estimation of the classifier might be biased and/or reduce the classification performance significantly.

Yousef *et al.* propose a method which only uses putative microRNAs as the positive class for the training procedure and does not need a negative class. The one-class approach only needs microRNA sequences for building the model. In addition, the authors propose 62 features which are extracted from both secondary structure and sequence of the microRNAs.

The authors conducted many experiments for evaluating and comparing the performance of the proposed method on various datasets using various classification methods.

They performed experiments on the following pre-microRNA datasets: human, mouse, *C.elegans*. The following classification methods were also used: one-class SVM, one-class Gaussian, one-class PCA, one-class KNN, two-class naïve Bayes and two-class SVM. Authors also conducted an experiment on finding microRNA genes in the *Epstein Barr Virus (EBV)* genome.

Yousef *et al.* state that results of one-class approaches for Gaussian and KNN show slightly better performance. Whereas on average accuracy of one-class approaches are around 8% to 10% lower than two-class methods. Authors claim that applying the method on EBV genome showed that all one-class approaches could distinguish EBV microRNAs with sensitivity of 72% – 90% in which one-class PCA has the highest sensitivity.

Yousef *et al.* [40] claim that their newly introduced features can describe microRNAs more accurately than previously proposed features. Also, it is stated that the one-class method is very useful especially when negative samples are not clearly defined which is usually true when a new organism is being analyzed.

2.5.4 Global and Intrinsic Folding Features

Identifying microRNAs from a pool of sequences without sacrificing putative microRNAs is a very challenging task. That is because microRNAs are relatively short in length and "have highly diverse base compositions". Ng and Mishra [29] propose a method for tackling this problem. The authors criticize the approach proposed in [39] in which they are limited to only microRNAs without multiple loops. They point out low sensitivity of [28]'s approach that is 73%. Authors also state that the approach presented in [41] rely on comparative analysis of results in order to reduce the false positive rate. In addition, they state that none of the previous works except [39] had conducted an analysis of the importance of

the features which are used in the referred approaches.

Ng and Mishra [29] propose 29 new "RNA global and intrinsic folding" features and employ SVM as the classification algorithm. Features can be categorized as follows: sequential, hairpin folding-related, statistical thermodynamics and topological. The authors refer to their classifier as *miPred*.

The authors obtained a total number of 2,241 pre-microRNAs from miRBase 8.2. They used 200 *human* pre-microRNAs and 400 randomly selected pseudo hairpins for training and finding parameters of *miPred* and called it TR-H. Another dataset containing 123 *human* and 246 pseudo hairpins were used in the testing procedure, which is referred to as TE-H. The authors also evaluated performance of the *miPred* on three other datasets from non-human species, ncRNAs and mRNAs which are referred to as IE-NH, IE-NC and IE-M, respectively. In addition, they experimented screening viral-encoded microRNA genes using four complete viral genomes. Finally, an analysis of contribution of each and every feature to *miPred* classification ability was done to see whether selecting a subset of features leads to improvement or worsening the performance of the classifier.

Ng and Mishra [29] claim that, they achieved 88.00% / 97.50% / 94.33% and 84.55% / 97.97 % / 93.50% which are sensitivity(*SE*), specificity (*SP*) and accuracy (*ACC*) for TR-H and TE-H, respectively. And, the authors state that *miPred* can achieve 87.65% / 97.75% / 94.38% for *SE*/*SP*/*ACC* when it was used on the IE-NH dataset. Applying the classifier on IE-NC and IE-M led to performance specificity of 76.15% and 87.10%, respectively. As for viral genome, the classifier can classify microRNAs with sensitivity and specificity of 100.00% and 93.75%, respectively. They claim that investigation on importance of the features in terms of discriminant power, shows that all the features are strongly and positively correlated and structural features have the strongest discriminant

power among all other features.

Ng and Mishra also claim that their approach has "comparable or significantly" better identification performance when comparing it to all previously proposed methods. They also suggest using *miPred* as a tool for experimental research.

2.5.5 Enhancing Global and Intrinsic Folding Features

MicroRNAs are an important type of non-coding RNAs which participate in post-transcriptional gene regulations. It is well studied that, there is an association between microRNA expression levels and many diseases. Therefore, Batuwita and Palade [4] believe that it is very important to provide a computational tool for biologists to be able to effectively identify microRNA genes in genomes. Thus, they introduced a model for microRNA identification.

Batuwita and Palade criticize previous works for only relying on genome pseudo hairpins for generating the negative class. However, they state that there are vast number of sequences which fold into hairpin secondary structure and are non-coding RNAs (ncRNAs). Although they mentioned that the authors of [34] considered tRNAs and rRNAs in the negative training dataset, and the dataset was not representative enough.

The authors introduce a new negative dataset containing other ncRNAs and genome pseudo hairpins and state that the dataset is "complete and representative". They also propose a new set of features, used feature selection algorithms and tried to solve class-imbalance problem.

The authors performed some experiments on a dataset containing 691 human pre-microRNA sequences in the positive class and 9,248 false pre-microRNAs in the negative class. They evaluated performance of the classifier using a "systematic" 5-fold cross-validation. They

performed experiments for finding the best subset of features and for tackling the class-imbalance problem which they say it arises when the number of samples in the positive class and the negative class is highly unbalanced. Finally, they applied their method on a dataset containing pre-microRNAs across 49 and 12 animals and viruses, respectively.

Batuwita and Palade [4] state that their method achieved 80.23% / 98.71% / 89.04% for $SE/SP/G_m$ when using all features. Applying feature selection algorithms resulted in a subset of features containing 21 features instead of all 48 features with 83.36% / 99.00% / 90.84% for $SE/SP/G_m$. It is shown that the class imbalance learning results are as follows: 90.02% / 97.28% / 93.58% for $SE/SP/G_m$. Finally, applying the proposed method on non-human datasets showed accurate microRNA prediction. The authors [4] claim that, their method has better performance by comparing it to previous methods. In addition, they claim that their method could be coupled with deep-sequencing data to incorporate advanced features introduced in those methods.

2.5.6 Co-learning of Sequence and Structure

Nam *et al.* [28]'s approach mainly relies on a hidden Markov model for identification of microRNAs. Identification of microRNA genes is a very important problem for understanding post-transcriptional gene regulation. Computational approaches for identifying microRNAs could be used even when microRNA is expressed low or in a particular tissue.

Nam *et al.* propose a probabilistic co-learning approach that is based on a paired hidden Markov model (HMM) for identification of microRNAs while continuously considering structure and sequence of pre-microRNAs. In Nam *et al.* method [28], each pre-microRNA is represented as a pairwise sequence which can be modeled as a sequence of matched pairs. The state of each pair can be formulated based on its base paring status, whereas

each position of the pairwise sequence has two states, structural and hidden.

A dataset consisting of 136 human pre-microRNAs in positive class and 1,000 randomly selected pseudo hairpins as negative, was used during the experiments. The authors performed some experiments on the positive and negative datasets using 5-fold cross-validation, and ROC curves were also plotted for analyzing the performance. They also used the method for scanning human chromosomes 16, 17, 18 and 19 for detecting pre-microRNA candidates.

Nam *et al.* state that on average the method successfully classified pre-microRNAs with 72.8% sensitivity and 95.9% specificity. The method was able to detect 253, 274, 83 and 207 pre-microRNA candidates on chromosomes 16, 17, 18 and 19, respectively.

2.5.7 The Ranking Algorithm Based on Random Walks

These are some of the reasons why Xu *et al.* [38] introduced a new method for solving this problem. They noted that previous works were not effective on regions of genome which are not annotated very well and this is because obtaining a set of negative examples is difficult. In addition, they comment on lack of positive examples in many species except in well studied species, such as *A.gambiae*.

Xu *et al.* proposed a ranking algorithm which is based on random walks. The approach tries to find new microRNAs in genomes even when only a few number of microRNAs are known and the genome is annotated poorly. It is stated that the algorithm requires no negative samples. Basically, the authors formulate identifying microRNAs as an information retrieval problem in which microRNAs should be retrieved from a set of microRNA candidates. Each sample is represented as a vector containing 36 features. Among these features, 32 of them are taken from [39] and the other four structural and topological fea-

tures are as follows: “normalized free energy of folding (MFE)”, “normalized base-pairing propensities” of both strands of the pre-microRNA and “normalized loop length”.

Xu *et al.* [38] performed experiments on *H.sapiens* and *A.gambiae* (533 and 38, respectively). Also, they generated other sequences from *H.sapiens* and *A.gambiae* genomes for making a pool of sequences. They evaluated performance of the method on the two mentioned datasets. The authors also conducted an analysis on conservation of the *A.gambiae*.

They claim that their method achieved accuracy higher than 95.00% on putative human microRNAs, and in the *A.gambiae* experiment, the algorithm could correctly predict 200 microRNAs. Also, conservation analysis revealed that 78 out of 200 microRNAs are conserved in at least one other animal species.

In addition, the authors claim that their method can be applied on newly sequenced genomes in which full annotation has not been done. They also state that it does not rely on conservation between species. Thus, they believe that their method can be used as a powerful tool for prediction of novel microRNAs in viral genomes.

2.5.8 The Naïve Bayes Algorithm

Yousef *et al.* [41] refer to the work of *Nam et al.* [28] and state that they also used features of microRNA genes instead of relying on conservation of microRNAs between related species.

The authors pointed out that Nam *et al.* [28] only used human microRNAs and a limited set of negative samples for training and testing. In addition, the approach proposed in [28] is a very specific probabilistic model which uses prior knowledge for constructing the model and defining the states.

Yousef *et al.* [41] described a new approach that is based on the naïve Bayes classifier. Initially, prior knowledge is used for filtering out the data, followed by a naïve Bayes classi-

fier and analyzer for selecting the sequences with highest probability of being a microRNA. The classifier is built using a dataset consisting of sequential and structural features of putative microRNAs from multiple species. Finally, the model uses a comparative analysis to reduce the number of false positive potential microRNAs. The authors state that the novelty of the work is in using a variety of organisms for building the model.

The authors state that their various experiments were conducted for evaluating the performance of the model. First, the training process was applied to microRNAs of *C.elegans* and *Mouse*. Experiments were done with different sizes of negative sample sets. Evaluation was followed by 5-fold cross validation and the receiver operating characteristic (ROC) curve. After single species, the learning procedure was done with microRNAs from multiple organisms with the same training and evaluating methods used in single species step. Finally, an experiment was conducted for predicting microRNA genes in the *Mouse* genome.

The authors claim that their model can achieve specificity and sensitivity of 96% and 83% for *C.elegans* and 91% and 97% for *Mouse*, respectively. For multiple species experiments, they claim that their model can successfully classify the data with high accuracy. Finally, they stated that the model detected a reasonable number of microRNA genes.

The authors claim that their model has a high generalization ability since it is trained using microRNAs of multiple organisms. Thus, they state that this method can be used for identifying microRNAs in a wide range of *Eukaryotes*. Also, they state that their algorithm can achieve higher specificity and similar sensitivity compared to all previously developed algorithms.

2.5.9 The Random Forest Algorithm For Classification

Jiang *et al.* [21] proposed a method called MiPred for distinguishing pre-microRNAs from other sequences with similar stem-loop secondary structure. Identifying pre-microRNAs systematically and experimentally tend to miss novel pre-microRNAs and is highly dependent on cell's condition. That is why computational approaches which do not rely on comparative genomic-based method play very important roles.

The authors propose a method that uses a set of 34 features including minimum free energy (MFE) of the secondary structure, "local contiguous triplet structure composition" and *P*-value of randomization test. All these features are then given to a machine learning algorithm, called random forest (RF) [21].

Jiang *et al.* [21] state that they used a dataset containing human pre-microRNAs and human pseudo hairpins for training and testing the classifier while using different features. Then after training and testing they used SVM instead of random forest to evaluate distinctive power of the proposed features regardless of the classifier algorithm. The *homo sapiens* dataset contained 462 human pre-microRNA whereas the pseudo pre-microRNA dataset contained 8,494 pre-microRNA-like hairpins. The authors also conducted an analysis on the importance of the features in order to rank features based on their prediction performance throughout the training procedure. In addition, they also implemented a *t*-test for comparing the performance of random forest and SVM.

The authors state that their classifier predicts pre-microRNAs while using all features (local contiguous, MFE and *P*-value of randomization test with 90.47%/95.09%/96.68% specificity / sensitivity / accuracy. Also, the classifier achieves its lowest performance when it uses contiguous features. Jiang *et al.* [21] state that the SVM classifier with the same set of features slightly outperforms the random forest (RF) classifier with the same features.

They claim that on average SVM is about 0.50% lower than RF with p-value of 0.003. Finally, the authors claim that analyzing importance of the features show that p-value and MFE are the two most influential features among all other features.

The authors claim that minimum free energy and p-value of randomization test are very important features which can be used in identifying pre-microRNAs when random forest is used as the classifier.

2.5.10 Structural Motifs

Brameier and Wiuf [6] propose a method for distinguishing microRNAs by genome scanning which only depends on secondary structure of pre-microRNAs from pseudo hairpins. The proposed classifier relies on linear genetic programming which contains "multiple regular expressions (motifs)" matched to pre-microRNAs secondary structure. The authors also propose a new criterion for selecting potential microRNA candidates.

The authors used 474 human pre-microRNAs from miRBase 9.0 for the positive set and 100,000 sequences, which were taken from 20,000 random locations in the human genome, for the negative set. After pre-filtering the two datasets, the authors used datasets for training the motif-based classifier and tried with 16 different motifs. An ROC curve was used for comparing the performance when using different numbers of motifs. In another experiment, the authors applied 5-fold cross-validation to evaluate the classification performance. In addition, other experiments were performed which are as follows: evaluating the performance on other species, scanning human genome for finding new microRNAs.

Brameier and Wiuf [6] claim that they achieved 99.90% / 87.00% and 99.10% / 95.00% for sensitivity and specificity when using 16 motifs and 1 motif, respectively. In 5-fold cross-validation, the authors claim that the classifier achieved 95.00% and 90.00% sensi-

tivity during training and testing, respectively, whereas on average, specificity remained at 99.10%. Also, the authors claim that their method predicts 74% and 81% percent of mouse and rat microRNAs when using all motifs. However, it identified 91 and 98 percent when using half of the motifs. It was stated that scanning human genome resulted in identifying 117 new microRNAs on human chromosome 19.

Brameier and Wiuf [6] claim that their method is competitive when compared to all previously developed approaches. Also, they state that their method requires less amount of knowledge about pre-microRNAs. In addition, using motifs and genetic programming enhances knowledge interpretation and extraction.

2.5.11 The Kernel Density Estimation Algorithm

Chang *et al.* [8] proposed a method for identifying microRNAs. They believe that having a computational approach which does not rely on analyzing the similarity of the sequence with putative microRNAs, and can work without prior knowledge about microRNA homology, is necessary. They focus on a classification methodology for microRNA identification and use the relaxed variable kernel density estimator (RVKDE) which is an instance-based classification algorithm. The authors use 40 features which were all previously introduced in other works.

Chang *et al.* [8] conducted some experiments for evaluating the performance of the classifier. They used a dataset containing 400 human pre-microRNAs (HU400) and they used 5-fold cross-validation for measuring the performance as well as comparing it with previous methods. Then, they used the trained classifier for extending the experiment to non-human pre-microRNAs. The dataset they used for this experiment includes 1,675 pre-microRNAs from 39 non-human species. The authors also investigated the contribution of

using the RVKDE on the performance of the classifier. Finally, they performed an experiment for explaining characteristics of the RVKDE in microRNA prediction.

The authors state that their classifier can achieve 90.5% / 97.5% / 94.0% of *SE* / *SP* / *ACC* on HU400 dataset. Sensitivity, specificity and accuracy of the RVKDE classifier are 96.7%±2.7%, 93.9%±2.1% and 95.3%±1.4%. The authors state that investigating the effect of using RVKDE revealed that its performance is identical or better than SVM and also it tends to maximize specificity, whereas SVM tries to maximize sensitivity. At the end, Chang *et al.* [8] state that the RVKDE is "instance based and highly dependent on the local information" of training samples.

The authors claim that the *RVKDE* is more suitable for microRNA identification since it uses more local information about the sequences. On the whole, the authors believe that good performance of their classifier should encourage more research on classification methods and feature extraction.

2.5.12 Feature Selection via a *Genetic Algorithm*

As noted earlier, identification of microRNAs play a crucial role in understanding their biological functions in cells and can potentially lead to curing many diseases. There are plenty of methods and dozens of features proposed for distinguishing these tiny molecules. However, selecting a subset of features which can help biologists to interpret them is very important. Thus, Wang *et al.* [36] propose a feature selection method that is based on a genetic algorithm (GA) for selecting the best subset of features.

Wang *et al.* [36] refer to microPred [4] in which they use a *filter* based feature selection method. However, it has been shown that the performance of *wrapper* feature selection method is better than filter based methods. Thus, the authors propose a classifier which

uses a GA-based algorithm to optimize the feature subset of an SVM classifier.

Wang *et al.* [36] use the dataset which was used in [4]. They use 183 features extracted from literature for the original feature set. The authors use the accuracy of five fold cross-validation of SVM classifier as the fitness function of the GA algorithm. They use many performance metrics in their work such as accuracy, specificity, sensitivity, F-measure and Matthews correlation coefficient for evaluating the performance.

Wang *et al.* [36] claim that their proposed classifier recognized 13 features as the best subset of features and it achieved an *accuracy* of 93.97% which is higher than that of micro-Pred and miPred. At the end, they also used their classifier on the most recently published microRNA dataset and the authors claim that the performance was satisfactory.

2.5.13 Sample Selection for Classification

As noted in many previous methods, class imbalance is one of the problems which should be considered when designing a classifier for microRNA identification. Han [20] proposed an approach which only focuses on solving the class imbalance problem. The author believes that an unbalanced dataset should be first manipulated in order to reduce the negative effects of the unbalanced data.

As mentioned earlier, the imbalance problem arises since the number of samples in the negative class outnumbers samples in the positive class. Han [20] proposed a method which reduces the number of samples in the negative class by clustering methods. Thus, he proposed to cluster positive and negative training samples based on their stem similarity and their distribution in high dimensional sample space, respectively. This approach results in having a dataset which is quite balanced and can be used for classification. The author uses an SVM classifier.

Han [20] claims that the proposed approach is around 12% more accurate than *microPred* which means that it can achieve a performance of nearly 100%. The result the author claims is surprisingly good and the way it is described all the microRNA precursors can be classified accurately from non pre-microRNA sequences. However, further analysis should be done in order to guarantee that reducing the number of samples in this way will not generate a biased dataset.

Chapter 3

Dimensionality Reduction and Explicit Mapping

Pattern recognition is a research area that has attracted many researchers. It is mostly an interdisciplinary field of study covering computer science, statistics, engineering, artificial intelligence and many other subjects. It has been widely used in different applications such as classifying cancerous genes, identifying spam emails, image recognition and credit card fraud detection [11]. In recent years, significant progress has been made specially where the research domain overlaps with probability and statistics and many improvements have been achieved both in application side as well as in methodology.

A particular active area of pattern recognition is the application of algorithms and techniques in solving bioinformatics problems. Analyzing large biological datasets requires employing pattern recognition and machine learning techniques in order to extract useful knowledge out of them. Examples of application of this field in bioinformatics include identifying clusters of gene expression data, distinguishing different types of protein-protein interactions, cancer classification based on microarray data and many others. Therefore,

pattern recognition methods have been well used in bioinformatics problems.

3.1 Dimensionality Reduction

The complexity of the most learning algorithms relies on the number of input dimensions, d and number of input data samples, N. For reducing the complexity, decreasing the number of dimensions of the input data is desirable. In addition, simpler models are more robust and less dependent on noise and outliers. Also, when fewer dimensions are used in learning methods without loss of relevant information, data can be visually analyzed and interpreted [2].

Generally, there are two methods for reducing the dimensionality of learning systems feature selection and feature extraction [2]. In *feature selection* methods, the goal is to select k of the d dimensions (features) of the dataset that can be representative of the classes and can give the most information about the original dataset. Once k dimensions are selected, the other $k-d$ dimensions will be ignored and considered useless.

In *feature extraction* methods, it is desired to find a set of k dimensions, which are combinations of original features. Depending on whether we use the class labels of the samples or not, these methods are categorized into *supervised* and *unsupervised*. Well-known feature extraction methods are *Principle Component Analysis* and *Linear Discriminant Analysis* (LDA) which are unsupervised and supervised algorithms, respectively. In this study, LDA is used for reducing the dimensions of the input data.

3.1.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a supervised approach for dimensionality reduction for classification problems originally developed by R.A. Fisher in 1936. LDA is a well-studied topic in pattern recognition. This is one of the methods available for linear dimension reduction. The advantage of using a linear transformation is that, although the derivation of the underlying transformation may be slower, the classification is extremely fast as it performs linear-time operations to reduce the dimensions, typically, much lower than the original one. There are different schemes for finding transformation matrix $\mathbf{A}$ which can project the data into lower dimensions in a way what the new classes are as separate as possible while classes are as compact as possible. In this work, we consider three different LDA schemes; the well-know Fisher's discriminant analysis (FDA) [11, 14], the heteroscedastic discriminant analysis (HDA) approach [25], and the Chernoff discriminant analysis (CDA) approach [33]. All these three methods propose different approaches for finding a "good" transformation matrix $\mathbf{A}$. A brief discussion of these three schemes is given in the next three sections.

We consider two classes, ω_1 and ω_2 (positive and negative classes), represented by two normally distributed random vectors $\mathbf{x}_1 \sim N(\mathbf{m}_1, \mathbf{S}_1)$ and $\mathbf{x}_2 \sim N(\mathbf{m}_2, \mathbf{S}_2)$, respectively, with p_1 and p_2 the a priori probabilities. After the LDA is applied, two new random vectors $\mathbf{y}_1 = \mathbf{A}\mathbf{x}_1$ and $\mathbf{y}_2 = \mathbf{A}\mathbf{x}_2$, where $\mathbf{y}_1 \sim N(\mathbf{A}\mathbf{m}_1;\ \mathbf{A}\mathbf{S}_1\mathbf{A}^t)$ and $\mathbf{y}_2 \sim N(\mathbf{A}\mathbf{m}_2;\ \mathbf{A}\mathbf{S}_2\mathbf{A}^t)$ with $\mathbf{m}_i$ and $\mathbf{S}_i$ being the mean vectors and covariance matrices in the original space, respectively. The aim of LDA is to find a linear transformation matrix $\mathbf{A}$ in such a way that the new classes ($\mathbf{y}_i = \mathbf{A}\mathbf{x}_i$) are as separable as possible.

3.1.1.1 Fisher's Discriminant Analysis

Let $\mathbf{S}_W = p_1\mathbf{S}_1 + p_2\mathbf{S}_2$ and $\mathbf{S}_E = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^t$ be the within-class and between-class scatter matrices respectively. The well-known FDA criterion consists of maximizing the Mahalanobis distance between the transformed distributions by finding $\mathbf{A}$ that maximizes the following function [11]:

$$J_{FDA}(\mathbf{A}) = tr\left\{(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}(\mathbf{A}\mathbf{S}_E\mathbf{A}^t)\right\}. \tag{3.1}$$

The matrix $\mathbf{A}$ that maximizes (3.1) is obtained by finding the eigenvalue decomposition of the matrix:

$$\mathbf{S}_{FDA} = \mathbf{S}_W^{-1}\mathbf{S}_E, \tag{3.2}$$

and taking the d eigenvectors whose eigenvalues are the largest ones. Since $\mathbf{S}_E$ is of rank one, $\mathbf{S}_W^{-1}\mathbf{S}_E$ is also of rank one. Thus, the eigenvalue decomposition of $\mathbf{S}_W^{-1}\mathbf{S}_E$ leads to only one non-zero eigenvalue, and hence FDA can only reduce to dimension $d = 1$.

3.1.1.2 Heteroscedastic Discriminant Analysis

HDA was proposed as a new LDA technique for normally distributed classes [25], which takes the Chernoff distance in the original space into consideration to minimize the error rate in the transformed space. It can be seen as a generalization of FDA to consider heteroscedastic classes, and the aim is to obtain the matrix $\mathbf{A}$ that maximizes the function:

$$\begin{aligned} J_{HDA}(\mathbf{A}) = tr\Big\{&(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}\,[\mathbf{A}\mathbf{S}_E\mathbf{A}^t \\ &-\mathbf{A}\mathbf{S}_W^{\frac{1}{2}}\frac{p_1\log(\mathbf{S}_W^{-\frac{1}{2}}\mathbf{S}_1\mathbf{S}_W^{-\frac{1}{2}})+p_2\log(\mathbf{S}_W^{-\frac{1}{2}}\mathbf{S}_2\mathbf{S}_W^{-\frac{1}{2}})}{p_1p_2}\mathbf{S}_W^{\frac{1}{2}}\mathbf{A}^t\Big]\Big\} \end{aligned} \tag{3.3}$$

where the logarithm of a matrix $\mathbf{M}$, $\log(\mathbf{M})$, is defined as:

$$\log(\mathbf{M}) \triangleq \Phi \log(\Lambda) \Phi^{-1}. \tag{3.4}$$

with Φ and Λ representing the eigenvectors and eigenvalues of $\mathbf{M}$, respectively.

The solution to this criterion is given by computing the eigenvalue decomposition of:

$$\begin{aligned} &\mathbf{S}_{HDA} = \mathbf{S}_W^{-1} \\ &\left[\mathbf{S}_E - \mathbf{S}_W^{\frac{1}{2}} \frac{p_1 \log(\mathbf{S}_W^{-\frac{1}{2}} \mathbf{S}_1 \mathbf{S}_W^{-\frac{1}{2}}) + p_2 \log(\mathbf{S}_W^{-\frac{1}{2}} \mathbf{S}_2 \mathbf{S}_W^{-\frac{1}{2}})}{p_1 p_2} \mathbf{S}_W^{\frac{1}{2}} \right] \end{aligned} \tag{3.5}$$

and choosing the d eigenvectors whose corresponding eigenvalues are the largest ones.

3.1.1.3 Chernoff Discriminant Analysis

CDA is an LDA method that has been recently proposed, and its aim is to maximize the separability of the distributions in the transformed space, measured by the Chernoff distance between the two classes. CDA assumes that the classes are normally distributed (in the original and transformed spaces), maximizing the following function [33]:

$$\begin{aligned} J_{CDA}(\mathbf{A}) = tr\{ & p_1 p_2 \mathbf{A} \mathbf{S}_E \mathbf{A}^t (\mathbf{A} \mathbf{S}_W \mathbf{A}^t)^{-1} \\ & + \log(\mathbf{A} \mathbf{S}_W \mathbf{A}^t) - p_1 \log(\mathbf{A} \mathbf{S}_1 \mathbf{A}^t) - p_2 \log(\mathbf{A} \mathbf{S}_2 \mathbf{A}^t) \} \end{aligned} \tag{3.6}$$

where $\mathbf{S}_W = p_1 \mathbf{S}_1 + p_2 \mathbf{S}_2$, $\mathbf{S}_E = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^t$.

It has been shown in [33] that for any normally distributed random vectors, $\mathbf{x_1}$ and $\mathbf{x_2}$, there always exists an orthogonal matrix $\mathbf{Q}$, where $\mathbf{Q}\mathbf{Q}^t = \mathbf{I}$, such that $J_{CDA}(\mathbf{A}) = J_{CDA}(\mathbf{Q})$ for any matrix $\mathbf{A}$ or rank d. Thus, without loss of generality, here, we assume that $\mathbf{A}$ is an orthogonal matrix. In [33], a gradient-based algorithm was proposed, which maximizes the

function (3.6) in an iterative way. The algorithm starts with an arbitrary orthogonal matrix $\mathbf{A}^{(1)}$, and at step $k+1$, $\mathbf{A}^{(k+1)}$ is computed as follows:

$$\mathbf{A}^{(k+1)} = \mathbf{A}^{(k)} + \alpha_k \nabla J_{CDA}(\mathbf{A}^{(k)}) \tag{3.7}$$

where the gradient for J_{CDA} is:

$$\begin{aligned}
&\frac{\partial J_{CDA}}{\partial \mathbf{A}} = \nabla J_{CDA}(\mathbf{A}) = 2p_1p_2\left[\mathbf{S}_E\mathbf{A}^t(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}\right.\\
&\left.-\mathbf{S}_W\mathbf{A}^t(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}(\mathbf{A}\mathbf{S}_E\mathbf{A}^t)(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}\right]^t\\
&+2\left[\mathbf{S}_W\mathbf{A}^t(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1} - p_1\mathbf{S}_1\mathbf{A}^t(\mathbf{A}\mathbf{S}_1\mathbf{A}^t)^{-1}\right.\\
&\left.-p_2\mathbf{S}_2\mathbf{A}^t(\mathbf{A}\mathbf{S}_2\mathbf{A}^t)^{-1}\right]^t
\end{aligned}$$

For this gradient algorithm, a learning rate, α_k needs to be computed. In order to ensure that the gradient algorithm converges, α_k needs to be maximized. In [33], the secant method is used for this, and the aim is to maximize the function:

$$\phi_k(\alpha) = J_{CDA}(\mathbf{A}^{(k)} + \alpha\nabla J_{CDA}(\mathbf{A}^{(k)})) \tag{3.8}$$

Starting with two initial values $\alpha^{(0)}$ and $\alpha^{(1)}$, the value of $\alpha^{(j+1)}$ at time $j+1$ is iteratively found as follows:

$$\alpha^{(j+1)} = \alpha^{(j)} + \frac{\alpha^{(j)} - \alpha^{(j-1)}}{\frac{d\phi_k}{d\alpha}(\alpha^{(j)}) - \frac{d\phi_k}{d\alpha}(\alpha^{(j-1)})}\frac{d\phi_k}{d\alpha}(\alpha^{(j)}) \tag{3.9}$$

where

$$\frac{d\phi_k}{d\alpha}(\alpha) = [\nabla J_{CDA}(\mathbf{A}^{(k)} + \alpha\nabla J_{CDA}(\mathbf{A}^{(k)}))] \cdot \nabla J_{CDA}(\mathbf{A}^{(k)}) \tag{3.10}$$

The operator "$\cdot$" is the dot product between two matrices, and is computed, for any two

matrices $\mathbf{C}$ and $\mathbf{D}$, as follows: $\mathbf{C} \cdot \mathbf{D} = tr\{\mathbf{C}\ \mathbf{D}\}$. The value of $\nabla J_{CDA}(\mathbf{A}^{(k)} + \alpha \nabla J_{CDA}(\mathbf{A}^{(k)}))$ is computed by replacing $\mathbf{A}$ by $\mathbf{A} + \alpha \nabla J_{CDA}(\mathbf{A})$ in the Equation (3.8).

Finally, with the definition of $\frac{d\phi_k}{d\alpha}(\alpha)$, Equation (3.9) can be solved, and the gradient algorithm continues with the next iteration. The complete algorithm can be found in [33]. One of the keys in this algorithm is the initialization of the matrix $\mathbf{A}$, and in this work, we have performed ten different initializations and then chosen the solution for $\mathbf{A}$ that gives the maximum Chernoff distance.

3.2 Feature Selection

As mentioned earlier, feature selection is a very important task for a variety of reasons [2] increasing the generalization performance, speeding up the training and testing processes, improving classification performance such as predictive accuracy, and result comprehensibility [42]. Feature selection algorithms can be widely categorized into two groups: *filter* and *wrapper* methods. Filter methods evaluate the "goodness" of the feature subset by using the intrinsic characteristics of the data. They are computationally cheap, since they do not involve the induction algorithm. However, they also take the risk of selecting subsets of features which may not match the chosen induction algorithm. Wrapper methods, on the contrary, directly use the induction algorithm to evaluate the feature subsets. They generally outperform filter methods in terms of prediction accuracy, but they are computationally more intensive [19]. Brute-force search is a method that evaluates the performance of the classifier based on different subsets of features. In this method, the performance of all possible combination sets of features are compared with each other. In other words, the performance of all possible two-feature-pairs are compared with the performance of all possible subsets of three features and so on. Despite the fact that brute-force search guar-

antees the highest accuracy, it is extremely time-consuming and impractical – brute-force search should find the best subset of features among 2^d subsets of features, where d is the number of features (dimensions). Thus, the search space is extremely large that it is not possible to run this method for more than a few features. Another feature selection method is forward search which is a greedy algorithm to find a sub-optimal subset of features [35]. This algorithm starts with the null set and selects features to be added to the set one at a time, based on the performance of the classifier with the currently selected feature in addition to a potential selected feature. This algorithm is very fast and usually has an acceptable performance, but does not guarantee the best subset of features.

In this study, we introduce a systematic feature selection method that is based on floating forward search and aims to improve the performance of the basic algorithm. The improvement relies on searching a larger feature space compared to the basic forward search approach. In our approach, the best 10 pairs of features among all the pairs (2-tuples) of features are selected. Then, all combinations of pairs with a third feature are evaluated and stored in a database, and again, the best 10 triplets (3-tuples) of features are selected. This procedure is continued with k-tuples, $k = 4,5,\ldots$ until a criterion is satisfied. The criterion can be a certain number of features being selected or selecting a new feature that does not improve the performance significantly. In our approach, the feature selection process is continued until a certain number of features are evaluated (11 in our case). The formal definition of the algorithm is given in Algorithm 1.

As mentioned earlier, since our dataset is unbalanced, G_m is used for comparison between the performance of the classifiers to ensure that class imbalance does not mislead feature selection algorithms to select the best subset of features, regardless of different proportions between the number of samples in each class.

Algorithm 1 Feature selection algorithm.

1: Let m be the number of feature subsets to be considered at each step and K_{MAX} be maximum number of features we want to consider.
2: Let $P(\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\})$ denotes the performance of the classifier using feature subset $\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}, 1 \leq i_1 \neq \ldots \neq i_k \leq d, 1 \leq k \leq d$ and d is number of dimensions.
3: Let $S_j(F_k)$ denotes the feature subset which achieves the jth performance in set F_k which is defined bellow.
4: $F_2 \leftarrow \left\{ \left(P(\{x_{i_1}, x_{i_2}\}), \{x_{i_1}, x_{i_2}\} \right) \mid 1 \leq i_1 \neq i_2 \leq d \right\}$
5: $k \leftarrow 2$
6: **repeat**
7: $\quad F_{K+1} \leftarrow \left\{ \left(P(\{x_{i_{k+1}}, S_j(F_k)\}), \{x_{i_{k+1}}, S_j(F_k)\} \right) \mid 1 \leq i_{k+1} \leq d, x_{i_{k+1}} \notin S_j(F_k), 1 \leq j \leq m \right\}$
8: $\quad k \leftarrow k+1$
9: **until** $k \leq k_{MAX}$
10: The best feature subset can be obtained by searching for highest performance through $F_k : 1 \leq k \leq k_{MAX}$.

3.3 Classification

In pattern recognition, classification is the problem of identifying the class which an observation belongs, based on a set of features that are extracted from the input dataset. This type of machine learning problem is considered as *supervised learning* as the class labels of each sample is known prior to classification. There are several types of classifiers developed in machine learning that some of them are introduced as follows: linear classifier, quadratic classifier, SVM, decision trees and etc. In this work, linear and quadratic classifiers are used for classifying the transformed data ($\mathbf{y} = \mathbf{A}\mathbf{x}$) that is projected by LDA.

There are many ways to represent classifiers in pattern recognition. One of the most worthwhile approaches is by using *discriminant functions* $g_i(\mathbf{x})$, $i = 1, \ldots, c$, where c is number of classes. The classifier, assigns a sample to class ω_i if

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \text{ for all } j \neq i, \tag{3.11}$$

where $\mathbf{x}$ is the input feature vector.

A Bayes classifier can be represented by $g_i(\mathbf{x}) = P(\omega_i|\mathbf{x})$ so that the maximum discriminant function corresponds to the maximum *posterior probability*. Since the choices of the discriminant function, $g_i(\mathbf{x})$, can be always replaced by $f(g_i(\mathbf{x}))$, assuming that $f(.)$ is a monotonically increasing function, the following function is a common choice of discriminant function in *minimum-error-rate* classification which makes understanding and computation much easier:

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln P(\omega_i). \tag{3.12}$$

The structure of Bayes classifier relies on the conditional densities, $p(\mathbf{x} \mid \omega_i)$, as well as prior probabilities. One of the most popular density functions that has received more attention is the multivariate normal or Gaussian density. The general multivariate Gaussian density in d dimensions has the following form

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}} \mid \boldsymbol{\Sigma} \mid^{\frac{1}{2}}} \exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})] \tag{3.13}$$

where $\mathbf{x}$ is a d-dimensional feature vector, $\boldsymbol{\mu}$ is d-dimension mean vector, $\boldsymbol{\Sigma}$ is d-by-d covariance matrix, $\mid \boldsymbol{\Sigma} \mid$ and $\boldsymbol{\Sigma}^{-1}$ are its determinant and inverse, respectively.

If we assume that the densities $p(\mathbf{x} \mid \omega_i)$ are multivariate normal, that is $p(\mathbf{x} \mid \omega_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, then from Eq. (3.12) we would have:

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln \mid \boldsymbol{\Sigma}_i \mid + \ln P(\omega_i). \tag{3.14}$$

The linear and quadratic classifiers can be obtained by examining the discriminant function and resulting classification in special cases which will be described as follows.

3.3.1 Linear Classifier - $\Sigma_i = \sigma^2 I$

The simplest case occurs when the features are statistically independent, and when each feature has the same variance, σ^2. Simplifying Eq. (3.14) results in:

$$g_i(\mathbf{x}) = -\frac{(\mathbf{x}-\boldsymbol{\mu}_i)^t(\mathbf{x}-\boldsymbol{\mu}_i)}{2\sigma^2} + \ln P(\omega_i). \tag{3.15}$$

By expanding the $(\mathbf{x}-\boldsymbol{\mu}_i)^t(\mathbf{x}-\boldsymbol{\mu}_i)$ term, we have:

$$g_i(\mathbf{x}) = -\frac{1}{2\sigma^2}[\mathbf{x}^t\mathbf{x} - 2\boldsymbol{\mu}_i^t\mathbf{x} + \boldsymbol{\mu}_i^t\boldsymbol{\mu}_i] + \ln P(\omega_i), \tag{3.16}$$

where the term $\mathbf{x}^t\mathbf{x}$ can be omitted due to the fact that it is the same for all i and can be ignored. Therefore, the Eq. (3.16) can be summarized to the following equivalent *linear discriminant function*:

$$g_i(\mathbf{x}) = W_i^t\mathbf{x} + w_{i0}, \tag{3.17}$$

where

$$W_i = \frac{1}{\sigma^2}\boldsymbol{\mu}_i, \tag{3.18}$$

and

$$w_{i0} = -\frac{1}{2\sigma^2}\boldsymbol{\mu}_i^t\boldsymbol{\mu}_i + \ln P(\omega_i). \tag{3.19}$$

In the case of two classes, ω_1 and ω_2:

$$g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x}) = \frac{1}{\sigma^2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^t\mathbf{x} + \frac{1}{2\sigma^2}(\boldsymbol{\mu}_1^t\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^t\boldsymbol{\mu}_2) + \ln\frac{P(\omega_1)}{P(\omega_2)}. \tag{3.20}$$

The classifier will return ω_1 if $g(\mathbf{x}) > 0$, or else ω_2. A classifier which uses a linear discriminant function is called a *linear classifier*.

3.3.2 Quadratic Classifier - $\mathbf{\Sigma}_i$ = arbitrary

In a more general case, we assume that covariance matrices, $\mathbf{\Sigma}_i$, have arbitrary values for each class. Thus, by simplifying Eq. (3.14) the resulting discriminant function would be:

$$g_i(\mathbf{x}) = \mathbf{x}^t \mathbf{W}_i \mathbf{x} + W_i^t \mathbf{x} + w_{i0}, \tag{3.21}$$

where

$$\mathbf{W}_i = -\frac{1}{2}\mathbf{\Sigma}_i^{-1}, \tag{3.22}$$

$$w_i = \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i \tag{3.23}$$

and

$$w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^t \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\ln |\, \mathbf{\Sigma}_i \,| + \ln P(\omega_i)]. \tag{3.24}$$

This discriminant function is quadratic and the decision surfaces could have various shapes such as a hyperplane, a hypersphere, a hiperellipsoid, a hyperparaboloid, a hyperhyperboloid, etc. In the case of two classes, the classifier predicts ω_1 if $g_1(\mathbf{x}) > g_2(\mathbf{x})$, otherwise it predicts, ω_2.

Normally, $\mathbf{\Sigma}_1 \neq \mathbf{\Sigma}_2$ (in the case of two classes). This implies that the classifier is *quadratic Bayesian* (QB). I also consider a linear classifier by *forcing* the covariances to be the same by obtaining a new covariance as follows: $\mathbf{\Sigma} = \frac{\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2}{2}$. I call this classifier *linear Bayesian* (LB).

3.4 Non-linear Mapping of LDA

While LDA has been widely used in machine learning and pattern recognition due to their simplicity, there are some drawbacks in using linear transformations, especially when the data is non-linear and complex. Linear classifiers are usually inefficient when compare to more sophisticated classifiers such as SVM.

The kernels trick is used extensively in pattern recognition methods such as SVM, PCA and others. Also it has been shown that FDA using kernels significantly improves the performance of LDA [22, 26, 27]. The main idea of kernel based methods is to implicitly map the input data to a higher dimension hoping the data become linearly separable. For some methods, the kernel trick allows solving the problem of mapping and classifying without explicitly mapping data to a higher dimensional space. However, this approach is not possible for LDR methods such as CDA and HDA for which an implicit solution is far from trivial.

On the other hand, explicit mapping is a good alternative in some scenarios. The scenario which we face in this research work is that the data has a few dimensions (features), and a large number of samples. In fact, if implicit mapping solutions were available, the kernel matrices would be in the order of the number of samples. Generally, methods that work on a kernel matrix (Gram matrix) of the input dataset scale poorly with the size of the training dataset [32]. In our dataset, it would be around 10,000 $\times$ 10,000 in which basic algebraic operations on matrices are time and space consuming, and in many cases impossible.

Another advantage of using LDA methods is also related to dealing with a large number of samples on a lower dimensional space. Even though LDA methods are usually affected by the problem of singular matrices, it is not the case in our datasets. As a result of doing

feature selection, we deal with a very small number of features, and even after mapping to a higher dimensional space, the number of new features does not increase to more than one hundred (depending on the choice of parameters), which is still low, compared to the number of samples (a few thousands).

Therefore, we propose to explicitly map the data onto a higher dimensional space and then applying LDA methods on the mapped data. Explicitly mapping the data can be a challenging task since finding the actual mapping function $\phi(\mathbf{x})$ of kernels could be far from trivial, especially for the radial basis function (RBF) kernel, since it implicitly maps the data to an infinite dimension in the Hilbert space. In this study, mapping functions that are extracted from polynomial and RBF kernels are used.

3.4.1 Mapping with Linear Functions

Polynomial kernels are not as popular and widely used as RBF kernels which map the dataset to an infinite dimensional space. This is because a polynomial kernel might not achieve high accuracy as RBF kernels under similar training and testing cost. But the polynomial kernels have been very popular in natural language processing (NLP) [15].

Training and testing large datasets is usually time and space consuming. Chang *et al.* [9] propose a method in which they apply a linear-SVM classifier to the explicit form of polynomially mapped data. They claim that the approach has faster training and testing process comparing to using the kernel SVM while it achieves a good accuracy when compared using non-linear kernels. They also show that their algorithm is useful for large scale NLP datasets.

The polynomial kernel has the following general form

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + r)^n \tag{3.25}$$

where γ and r are the parameters, n is the degree of the polynomial and $\mathbf{x} \in \mathbb{R}^d$. The product of two mapping functions $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ is the polynomial kernel. By setting $d = 2$, $r = 1$ and $\gamma = 1$ and simplifying the result, the mapping function results in:

$$\phi(\mathbf{x}) = [1, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2, x_1 x_2, \ldots, x_{d-1} x_d]^T. \tag{3.26}$$

where $\phi(\mathbf{x})$ is of dimension $C(d+n, n)$.

In this work, the classifier uses the polynomially mapped data which is explicitly mapped. In other words, first the data is explicitly and polynomially mapped using the mapping function (3.26), $\phi(\mathbf{x})$ and then it is fed into the LDA-based classifier. As an example, if the original input dataset has two features (dimensions), the mapping function would transform it into $C(2+2, 2) = 6$ dimensions and LDA classifier will use the mapped dataset as its training and test sets.

In this work, I only consider polynomial mapping function of $d = 2$. Larger degrees of polynomial mapping function results to having large number of dimensions that causes *singularity problem*.

3.4.2 Mapping with the Gaussian Radial Basis Function

The Gaussian radial basis function kernel or simply the RBF kernel is one of the most popular kernels functions that is widely used in machine learning and pattern recognition. This kernel has the following form

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\gamma||\mathbf{x}_i - \mathbf{x}_j||^2}{2}\right) \tag{3.27}$$

As mentioned earlier, finding the mapping function of RBF is not as straightforward as finding the mapping function of polynomial kernel. Rahimi and Recht [32] proposed a randomized mapping function that maps input samples to a low-dimensional Euclidean inner product space, in which the inner product of any two mapped data is equivalent to a Gaussian radial basis function kernel of two input data. Therefore, we can simply transform the data using the mapping function, and then apply an LDA classifier on the mapped data.

The randomized map is composed of sinusoids randomly drawn from the Fourier transform of the Gaussian radial basis function. Basically, the randomized map projects data points onto a randomly chosen line, and then passes the resulting scalar through a sinusoid. The random lines are drawn from a distribution so it can guarantee that the inner product of two transformed points approximates the value of the actual mapping function (see Algorithm 2).

Algorithm 2 Explicit mapping with Gaussian RBF

Require: A positive definite shift-invariant kernel $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathcal{K}(\mathbf{x}_i - \mathbf{x}_j)$
Ensure: A randomized feature map $\phi(\mathbf{x}) : \mathcal{R}^d \to \mathcal{R}^D$ so that $\phi(\mathbf{x}_i^T)\phi(\mathbf{x}_j) \approx \mathcal{K}(\mathbf{x}_i - \mathbf{x}_j)$.
Compute the Fourier transform p of the kernel $\|$: $p(\omega) = \frac{1}{2\pi}\int e^{j\omega'\Delta}\|(\Delta)d\Delta$.
Draw $\boldsymbol{D}$ iid samples $\omega_1, \cdots, \omega_D \in \mathcal{R}^d$ from p.
Let $\phi(\mathbf{x}) \equiv \sqrt{\frac{1}{D}}\left[\cos(\omega_1' X)\ldots\cos(\omega_D' X)\sin(\omega_1' X)\cdots\sin(\omega_D' X)\right]'$.

Scikit-learn [30] provides a package written in Python for approximating the radial basis mapping. The package takes two parameters, $\boldsymbol{D}$ that is the number of dimensions of the transformed feature space, and γ, which is the parameter of the RBF kernel. We use their package in our implementation for approximating the value of the mapping function,

$\phi(\mathbf{x})$.

3.5 *K*-fold Cross-validation

Cross-validation is a technique for assessing the performance of a given classifier when using independent data. It is mainly used for estimating how well the predictive model, which is trained with the training data, will perform to classify the unseen testing data. There are different types of cross-validation method such as K-fold, leave-one-out and repeated random sub-sampling. In this research work, K-fold cross-validation is used for performance evaluation of the classifier.

In K-fold cross-validation, initially, the original dataset is partitioned into K different sub-datasets. Out of the K sub-datasets, one will be used for validation as testing set and the other $K-1$ sub-datasets will be used for training of the classifier. In this way, none of the samples which belong to the testing set is involved in the training process and is only used in testing the performance of the classifier.

One thing which should be stressed here is that, when the data is unbalanced, random splitting the dataset into K folds might lead to testing sub-datasets which has none or very few samples from the smallest class. Thus, it is crucial that the ratio between the number of samples in any of the partitions should be kept fixed when splitting the original dataset randomly and to be equal to ratio of samples in the original dataset.

3.6 Class Imbalance Problem and Performance Evaluation Challenge

Rare objects had not taken much attention in the context of machine learning and data mining until a decade ago [37]. However, real-world problems needed methods for handling rarity and addressing the problem of imbalanced data. Examples which data is unbalanced include credit card fraud detection, detecting oil spills from satellite images, detecting spam emails, etc [37]. One type of rarity is *rare classes*, or more generally, class imbalance. The class imbalance problem occurs when there is a major difference in prior class probabilities or when the disproportionate numbers of samples in positive and negative classes lead to poor performance of the classifier with respect to the smallest class [1].

There are some problems which may arise when working with a dataset that is imbalanced [37]. Here, some of these problems are listed below:

- Improper evaluation metrics
- Relative lack of data
- Lack of data
- Noise

Here, we only describe the *improper evaluation metrics* issue. *Improper evaluation metrics* refers to the problem that conventional evaluation metrics which do not value rarity, are more likely not to be able to properly handle the imbalance data [37]. As an example, *classification accuracy* which represents the ratio of data that is properly classified, cannot handle unbalanced data. It is well known that the rare class has less impact on the accuracy. For example, in a two-class classification problem if the ratio between two classes are 99:1,

the classification performance of a classifier that returns the class label of the majority class would be 99% which is misleading. Generally, when the number of samples in the negative class heavily outnumbers the number of samples from the positive class and the classifier always classifies samples as negative, the accuracy is high, although the classifier is useless. Hence, other indicators are required for analysis of the classification performance.

There are some methods for dealing with class imbalance problem. One of these methods is using evaluation metrics that take rarity into consideration by better guiding the search process and better evaluating the performance of the classifier [37]. ROC analysis and associated area under the ROC curve are two metrics that are used to evaluate the classification performance. Other examples are recall, precision and F-measure. Another evaluation measures that is employed in the context of imbalance data is the *geometric mean*, $G_m = \sqrt{SE \times SP}$, where sensitivity (SE) and specificity (SP) are defined as follows [1, 23]:

$$SP = \frac{TN}{TN + FP}, \tag{3.28}$$

$$SE = \frac{TP}{TP + FN}. \tag{3.29}$$

Here TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives and FN is the number of false negatives. Using LDA as the classifier is a good strategy to overcome the class imbalance problem. This is because LDA methods take the distribution of the whole data into account, in order to optimize the prediction function. This is in contrast with the criterion followed by the SVM that uses only the "support vectors" to find the most efficient prediction function. Although some SVM schemes have improved this by incorporating the concept of soft margin, the SVM

still relies on the vectors on (or next to) the margin, ignoring the contribution of the other samples to a more efficient classification rule. The latter feature is indeed intrinsic to the LDR techniques used in this work.

As mentioned in Chapter 2, the number of positive samples in comparison to the number of negative samples is small, with a ratio of about 1:13. In this case, standard classifiers have tendency to classify well the largest class, while ignoring the smallest class. Thus, in order to tackle this problem, G_m is used for performance evaluation of the classifier.

Part II

Methods

Chapter 4

Proposed Methodology

4.1 Dataset

The proposed classifier should be able to distinguish human pre-microRNAs from both pseudo hairpins and other non-coding RNAs. Therefore, the training dataset should include pre-microRNA sequences as the positive class and pseudo hairpins and other non-coding RNA sequences as the negative class. This dataset containing sequences and extracted features, is publicly available in [4] as *supplementary material*, and can be freely downloaded from `http://www.cs.ox.ac.uk/people/manohara.rukshan.batuwita/microPred.htm` . Detailed information about the datasets is presented here.

4.1.1 Positive dataset

Known human pre-microRNAs: This dataset includes 691 non-redundant human pre-microRNA sequences, which are obtained from `http://microrna.sanger.ac.uk/sequences/` [17, 18]. At first, 695 sequences were downloaded from miRBase and then after removing

redundant sequences, 691 sequences are obtained which fold into hairpin structures. Some of these sequences fold into multi-branched loops at default parameters, which show sequences with multi-branched secondary structures can also be identified by the proposed classifier. I do not make any assumption about pre-microRNA secondary structures.

4.1.2 Negative Dataset

Pseudo hairpins: The negative dataset is composed of 8,494 human pseudo hairpin sequences which were previously used in Triplet-SVM, MiPred, miPred and microPred. These sequences were obtained originally from RefSeq genes [31].

Other non-coding RNAs: This dataset contains all the non-coding RNA sequences, except microRNA sequences to make the classifier able to distinguish microRNA from other kinds of RNAs. This dataset contains 754 non-redundant sequences which are no longer than 150 nt. This dataset is known to be the best available ncRNA dataset for the human genome according to the authors of [16]. It includes 334 snoRNAs, 327 tRNAs, 53 snRNAs, 32 YRNAs, 5 5S-rRNAs, and three more sequences from other kinds of RNAs.

4.2 The Features

One of the most important aspects of designing a classifier is to extract the most relevant features which empower the classifier to distinguish between classes of data. Therefore, extracting the set of appropriate features from the dataset is very momentous. In addition, since the datasets used in the training and testing phases contain multi-branched sequences in addition to hairpin secondary structure, the extracted features should make classifier to be able to succeed.

Table 4.1: Primary structure features.

Symbol(s)	Description	Number of Features
$\%AA$, $\%AC$, ..., $\%UU$	Dinucleotide frequencies	16
$\%(C+G)$	C+G content	1

In this study, 48 features were used in which 29 of them were previously introduced in miPred and 19 features were proposed in microPred. Here, I provide a brief description [4] about the features. These 48 features can be categorized into four different groups *primary structure*, *secondary structure*, *energy-related* and *information-theoretic*.

4.2.1 Primary Structure

Primary structure features are those features that can be deducted by simply looking at the nucleotide sequence of the pre-microRNA. Table 4.1 summarizes 17 primary structure features. Let L be the length of the pre-microRNA sequence.

Dinucleotide frequencies, $\%XY$, where $X,Y \in \{A,C,G,U\}$ are calculated as follows:

$$\%XY = \frac{|XY|}{L-1} \times 100, \tag{4.1}$$

where $\mid XY \mid$ is number of denucleotide XY in the pre-microRNA sequence.

(C+G) content $\%(C+G)$ is calculated as follows:

$$\%(C+G) = \frac{|C|+|G|}{L} \times 100, \tag{4.2}$$

where $\mid C \mid$ and $\mid G \mid$ are number of nucleotides C and G in the sequence, respectively.

Table 4.2: Secondary structure features.

Symbol(s)	Description	Number of Features
nAUb/L, *nGCb/L*, *nGUb/L*	Number of each base pair normalized to sequence length	3
nAU/n_stems, *nGC/n_stems*, *nGU/n_stems*	Average number of each base pair per stem	3
BP/n_stems	Average number of base pairs per stem	1
dP	Normalized base-paring propensity	1
D, *dD*	Base-pair distance and its normalized variants by *L*	2
dF	The second (the Fielder) eigenvalue	1

4.2.2 Secondary Structure

Secondary structure features are more complex comparing to simple primary structure features in the sense that these features can only be calculated once the secondary structure of the sequence is predicted. These structural features are calculated based on secondary structure of the pre-microRNA sequence that is predicted by the *RNAfold* software package at the default temperature of 37°C. *RNAfold* predicts the secondary structure having the minimum free energy (MFE) of folding from the primary sequence of pre-miroRNA. Secondary structure features are listed in Table 4.2.

Let *n_stem* be number of stems in the secondary structure; stem is a structural motif of the secondary structure, which contains more than three contiguous stack of base pairs. Also, let *tot_bases* be the total number of base pairs in the secondary structure of a sequence.

The number of each base pair normalized to sequence length, *nXYb/L* where $(X-Y) \in \{(A-U),(G-C),(G-U)\}$, is defined as

$$nXYb/L = \frac{|\,X-Y\,|}{L}, \tag{4.3}$$

where $\mid X-Y \mid$ is the number of (X-Y) base pairs in the secondary structure.

The average number of each base pair per stem, *nXY/n_stems* where $(X-Y) \in \{(A-U),(G-C),(G-U)\}$, is defined as

$$nXY/n\_stems = \frac{\%(X-Y)}{n\_stems}, \tag{4.4}$$

where $\%(X-Y) = \frac{|X-Y|}{tot\_bases}$.

The average number of base pairs per stem, BP/n can be simply calculated as

$$BP/n = \frac{tot\_base}{n\_stems}. \tag{4.5}$$

The normalized base-paring propensity, *dP*, can be obtained using the following equation:

$$dP = \frac{tot\_bases}{L}. \tag{4.6}$$

The base-pair distance (*Diversity*), *D*, is defined as the average of all base pair distances between any two structures $\mathbf{S}_\alpha$ and $\mathbf{S}_\beta$ on sequence x as the number of base pairs not shared by the secondary structure $\mathbf{S}_\alpha$ and $\mathbf{S}_\beta$. Also, *dD* is defined as the normalized variant of *D* that is normalized by the sequence length (*L*).

The second (the Fielder) eigenvalue, *dF*, of the Laplacian matrix of the tree-graph structure is used as a similarity measure between secondary structure of RNA sequences. The idea is based on the fact that each pre-microRNA can be represented by a tree-graph *G* where the vertices represent loops and the edges represent stems. The Laplacian matrix *L(G)* is a mathematical representation of a tree-graph *G*. The second eigenvalue (*dF*) of *L(G)* measures the compactness of a tree *G* which can be used for measuring the degree of similarity between pre-microRNA sequences.

Table 4.3: Energy related features.

Symbol(s)	Description	Number of Features
dG	Normalized minimum free energy of folding	1
$MFEI_1$, $MFEI_2$, $MFEI_3$, $MFEI_4$	Variations of minimum free energy	4
$NEFE$	Normalized Ensemble Free Energy	1
$Freq$	Frequency of the MFE structure	1
$Diff$	Difference between MFE and EFE	1
Tm, Tm/L	Melting energy of the structure and its normalized form by length	2
dH, dH/L	Structure enthalpy and its normalized value by length	2

4.2.3 Energy Related

Among 12 energy related features, minimum free energy (MFE) is the most important feature and many variation from this feature have been derived such as normalizing MFE by length (L), number of stems (n_stems), etc. From a thermodynamical point of view, Boltzmann distribution can be used for probabilistically modelling an RNA molecule that exists in an assembly of structures. This information can be captured by the ensemble free energy (EFE), the ensemble diversity and other variant features. Table 4.3 shows all the 12 features that are related to energy of the pre-microRNA sequence.

Normalized minimum free energy of folding, dG, can be achieved using the following formula:

$$dG = \frac{MFE}{L} \tag{4.7}$$

$MFEI_1$, $MFEI_2$, $MFEI_3$ and $MFEI_4$ are different variants of minimum free energy and

can be calculated as follows:

$$MFEI_1 = \frac{dG}{\%(C+G)}, MFEI_2 = \frac{dG}{n\_stems}, MFEI_3 = \frac{dG}{n\_loops}, MFEI_4 = \frac{dG}{tot\_bases}, \quad (4.8)$$

where dG is defined in Eq. (4.7), n_loops is the number of loops in the secondary structure, n_stems is the number of stems and tot_bases is the total number of base pairs in the secondary structure of a sequence.

Normalized ensemble free energy, *NEFE*, is calculated by normalizing the ensemble free energy (EFE) by Length (L) and *NFE* can be calculated as below:

$$NEFE = \frac{EFE}{L}, EFE = -RT\ln(\mathbf{Z}), \quad (4.9)$$

where $\mathbf{Z}$ is free energy of a sequence, $R = 8.31451\frac{J}{molK}$ and T is the temperature taken as 310.15°K (37°C).

The frequency of the MFE structure, *Freq* can be obtained as $Freq = \exp(\frac{EFE-MFE}{RT})$. Also, *Diff* can be found as follows: $Diff = \frac{|MFE-EFE|}{L}$.

4.2.4 Information Theoretic

Information theoretic features offer a way of measuring the diversity of the possible structures of the pre-microRNA sequences. These features measure the entropy of the base pairing profile (structure entropy) and of the ensemble (Shannon entropy). In Table 4.4 all features in this category are listed.

Normalized Shannons entropy, *dQ*, models the distribution of pre-microRNA sequences that have different structures with a Boltzmann distribution of free energy. *dS* and *dS/L* are Structure Entropy and normalized Structure Entropy by L, respectively.

Table 4.4: Information theoretic features.

Symbol(s)	Description	Number of Features
dQ	Normalized Shannon entropy	1
dS, *dS/L*	Structure entropy and its normalized value by length	2

Table 4.5: Normalized features.

Symbol(s)	Description	Number of Features
zG, *zP*, *zQ* *zD*, *zF*	**Z**-value for *dG*, *dP*, *dQ*, *dD* and *dF*	5

4.2.5 Normalized Values

These features are normalized versions (**Z**-value) of some of the features which are described previously. Table 4.5 list all the 5 features.

4.3 Model Flowchart

In order to identify human pre-microRNA sequences, a system was designed to automatically manage large amount of experiments which should be done and store the intermediate results for further analysis and comparisons. The flowchart of the proposed system is illustrated in Figure 4.1 and all the components of the system are highlighted in the figure. In this section, the different components are introduced and described.

4.3.1 Feature Selection

The feature selection algorithm (see Algorithm 1) as described before is used for selecting the best subset of features in this work. As mentioned earlier, the evaluation metric that is

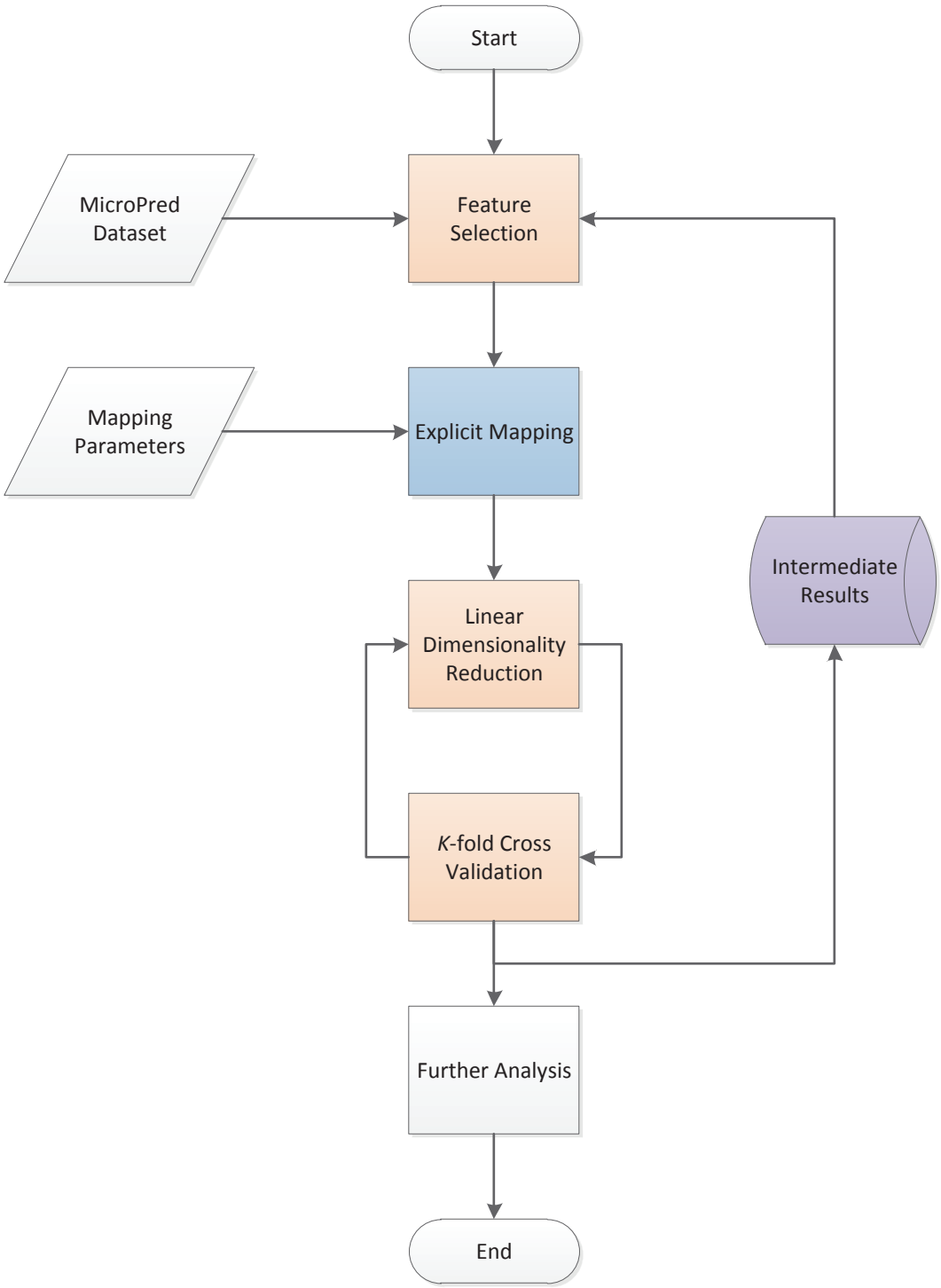

Figure 4.1: Overall flowchart of the proposed system.

used for evaluating the "goodness" of a feature set is geometric mean, G_m.

4.3.2 Explicit Mapping

In this component, the selected feature set is explicitly mapped onto higher dimensions. As mentioned earlier, in this work I use two types of mapping functions in order to map the data: *polynomial* and *RBF*. The polynomial mapping function does not have any parameter which should be set. However, the RBF has two parameters which are to be set prior to mapping to higher dimensions, which are $\boldsymbol{D}$ and γ. These parameters are fed into this component through another component responsible for deciding on these parameters.

All the coding in this part is done in Matlab except the algorithm for RBF mapping which is implemented in Python. The following trick is proposed to pass the data between Matlab and Python. First, the program saves the feature set on the local disk within Matlab. Then, the *scipy.io* Python package is used for reading the data and loading the feature set. After this step the *Scikit-learn* package is used for RBF explicit mapping and the result is saved again on the local disk. Once saving is done, the Matlab code will load the mapped data to its workspace and the data is ready to be processed further.

4.3.3 Mapping Parameters

Depending on the experiment that is going to be performed, this component should choose the parameters. In some experiments only default parameters will be passed to the Explicit Mapping component. However, when optimization of the parameters is the goal, then optimization algorithms shall be used for choosing the best parameters.

4.3.4 LDA classifiers and *K*-fold Cross-validation

This component is responsible for classifying the mapped data which is fed into it from the *Explicit Mapping* component. As mentioned earlier, there are six different combinations of LDA methods and classifiers which are FDA, HDA and CDA coupled with LB and QB. Thus, for each mapped feature set all these six LDA-based classifiers should be evaluated. Evaluating each of the classifiers is done by using *10*-fold cross validation. Simply, the dataset is divided into the 10 nearly-same-size groups and then, at each step nine of these sample groups is selected for training the classifiers, and one of them is used for testing purposes. This process is done until all sample groups are selected once for testing. It is very important that the sample group which is selected for testing should not be used in the training process. Once performances of the classifier for all the 10 different testing sets are evaluated, the average of the *separability* and *sensitivity* is used for calculating the average G_m. All the programming codes in these two components are implemented purely in Matlab. LDA-based classifiers were obtained by the implementation of the *Pattern Recognition and Bioinformatics* lab at *University of Windsor*, and in this work, a modified and simplified version of this code is used.

4.3.5 Intermediate Results

The design of this component is very important and there are some issues which should be considered carefully when designing it. An important issue is that the program should be able to recover the intermediate results in case of problems in the application which is inevitable during run-time of Matlab. Crashing happens due to many reasons such as limited memory of the computer, power outage, many others. Therefore, storing the intermediate results in the work space of Matlab is not an option. Thus, the results should be stored

on a local hard drive in order to prevent data lost. There are two options which were considered: using *flat file* databases such as plain text files and using *relational* databases such as Microsoft SQL Server.

On the other hand, the number of experiments which should be performed is large. Moreover, the experiments are not dependent on other experiments at each stage, there is a need for using paralleling ability of Matlab for reducing the execution time of the experiments. For this reason, the Matlab Parallel Computing Toolbox is used on a single machine in order to utilize all the CPU cores of a machine. Another important issue which should be considered is that, each thread of the parallel program should not lock the storage media at any point of time as another parallel thread might need to write data onto the storage media. Therefore, using flat file database might be troublesome in this sense unless a handler mechanism manages accessing each thread to the storage media which might be quit complex. For this reason, the solution that was employed in this research work is to use relational databases for storing the intermediate results. The relational database employed here is Microsoft SQL Server. Here are some of the benefits of using relational databases in this research work:

1. Concurrent requests are managed by the database server
2. Recoverable after program crashing
3. Generating reports easily
4. Writing queries for getting information
5. Easy to backup

Connecting Matlab to Microsoft SQL Server can be done in a few ways. In this work, Matlab Database Toolbox and jTDS JDBC Driver are used for connecting to the Microsoft

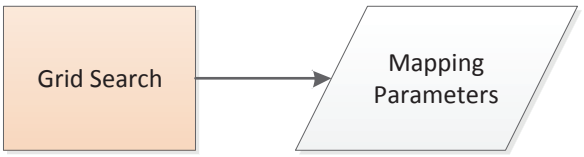

Figure 4.2: Optimizing Mapping Parameters.

SQL Server database. The proposed design allowed me to even perform experiments on different machines but still have all the results stored in another machine which is very beneficial as researchers should not be worried about moving data from one machine to another machine.

4.4 Optimizing Mapping Parameters

There are many optimization algorithms that can be used for optimizing the parameters of the explicit RBF. One of the algorithms that is used in this work is *grid search*. Grid search is an exhaustive search algorithm through a hyperparameter space for finding suboptimal parameters which are $\boldsymbol{D}$ and γ. Due to the fact that exhaustive search is highly time consuming, it is not feasible to perform optimization for all subset of features. The flowchart of the system changes, if the optimization is needed to be performed on the *Mapping Parameters* (Figure 4.2).

Part III

Results and Discussion

Chapter 5

Result and Discussion

In this research work, different experiments have been conducted on a human microRNA precursor dataset which is described in Chapter 4.1. As noted earlier, a 10-fold cross-validation is used for evaluating the classifier performance and also G_m is used as the performance metric. We have evaluated the performance of the proposed methodology in different experiments and results are shown in tables and figures in this chapter. In addition, comparisons have been made between the proposed method and previously proposed methods. At the end of this chapter, it is discussed that how the proposed methodology has advantages over previous methods as well as a discussion of features which the feature selection method has selected as the representative feature subset.

5.1 Experimental Results

As explained in the Proposed Methodology chapter, the original dataset has 48 features and our goal is to select a subset of these 48 features which can potentially be representative of the original dataset.

	Linear Classifier								
	FDA			HDA			CDA		
Mapping	*SE*	*SP*	G_m	*SE*	*SP*	G_m	*SE*	*SP*	G_m
None	52.53	99.27	72.21	51.81	99.18	71.68	52.39	99.27	72.12
Polynomial	38.64	99.85	62.11	51.81	99.65	71.85	56.01	99.85	74.78
RBF	61.22	99.60	78.08	80.32	96.57	88.07	80.32	96.57	88.07
	Quadratic Classifier								
	FDA			HDA			CDA		
Mapping	*SE*	*SP*	G_m	*SE*	*SP*	G_m	*SE*	*SP*	G_m
None	70.33	98.90	83.41	69.17	98.94	82.73	70.04	98.87	83.22
Polynomial	75.11	98.14	85.86	76.99	96.98	86.41	76.99	96.98	86.41
RBF	78.44	97.68	87.53	85.24	92.91	**88.99**	85.53	92.67	**89.03**

Table 5.1: Classification performance for different combinations of LDA methods coupled with linear and quadratic classifiers. Each row represents the best performance in term of G_m of the classifier when using none, polynomial and RBF mapping function.

As described in Algorithm 1 in Chapter IV, the feature selection method starts with evaluating the classifier with all possible combinations of two features which is a set of $\binom{48}{2} = 1,128$ different pairs of features. Table 5.1 lists the performance of the classifiers for all six combinations of FDA, HDA and CDA with linear Bayesian (LB) and quadratic Bayesian (QB) classifiers when the data (with two features) is not mapped as well as when the data is mapped using polynomial and RBF mapping.

As it is clear from the table, LDA methods coupled with QB performs better than linear classifiers. As an example polynomial+CDA+Q achieves G_m around 12% higher than polynomial+CDA+L. It can be concluded that the QB classifier significantly outperforming LB classifier at least in this Table 5.1.

In addition, by comparing different mapping functions while the LDA method and classifier is the same, it can be concluded that the data that is mapped using RBF leads to a higher G_m which is desirable. For instance, considering CDA coupled with QB classifier, and that when the data is mapped with RBF, the G_m is 89.03% compared to 86.41% and

Number of Features	*SE*	*SP*	G_m
2	85.24	92.91	88.99
3	85.53	97.51	**91.32**
4	85.82	93.59	89.62
5	84.23	96.16	90.00
6	83.94	97.13	90.29
7	86.54	96.91	**91.58**
8	90.59	91.24	90.92
9	86.83	92.68	89.71
10	87.12	93.18	90.10
11	87.84	92.90	90.33

Table 5.2: Performance of the classifier at different stages of Algorithm 1 with different numbers of features.

83.22% for polynomially mapped data and original data, respectively. A Comparison between the 18 different G_m metrics shows that that when the data is mapped with RBF, the classifier performs better and even in some cases the difference is quite large such as CDA coupled with LB classifier. In addition, the best performance is for the case in which the data is mapped with RBF and when the LDA criterion is CDA or HDA coupled with the quadratic classifier (with a slight difference around 0.04%).

As mentioned earlier, it is crucial for one to find a subset of features which could be representative of the whole dataset. Once the classifier examines all possible pairs, it continues with the procedure described in Algorithm 1 in Chapter IV. Table 5.2 lists the best performance of the six different combinations of three LDA methods coupled with two Bayesian classifiers for different numbers of features at different stages of the feature selection algorithms. The data is mapped with RBF and the parameters are $\gamma = 1.5$ and $\mathbf{D} = d + 15$, where d is the number of features. Figure 5.3 visualizes the performance of the classifier at different stages of the feature selection algorithm. As it is clear, when the classifier uses

seven features, it achieves its best performance with $G_m = 91.58\%$. In addition, when the classifier uses three features, it also achieves a very good performance ($G_m = 91.32\%$). This result is very important considering the fact that it only uses three features.

Deciding on the default parameter values: The values of γ and $\mathbf{D}$ were decided by analysis on pairs of features. For finding these values a grid search was done on different pairs of features with different values of γ and $\mathbf{D}$. Figures 5.1 and 5.2 shows the performance, G_m of the classifiers with $\gamma \in \{0.25, 0.5, 1, 2, 4\}$ and $\mathbf{D} \in \{5, 10, 15, 20\}$. The regions which are red indicateS higher performance and blue color indicate lower performance. As it is clear from Figures 5.1 and 5.2, the area around $\gamma = 1.5$ and $\mathbf{D} = 15$ is red, which shows higher performance. This is also true for other feature pairs, when $\gamma = 1.5$ and $\mathbf{D} = 15$. Based on this, I decided to choose $\gamma = 1.5$ and $\mathbf{D} = d + 13$, where d is the number of features in the original feature space.

Table 5.3 shows the performance of the classifier when different numbers of features are used. This table reveals very important observations about the classification and feature selection method that are used in this research work. The feature numbers are given in Appendix A. First, I discuss the advantage of using the proposed feature selection methods. As mentioned earlier, the feature selection algorithm keeps all the results when evaluating a certain number of features and then uses the top $m = 10$ feature subsets at its next step for adding a new feature. As seen from the table, at row 11, the classifier achieves $G_m = 91.32\%$ using three features, $\{20, 25, 32\}$, which is the second best performance in the whole table after row 51 in which the classifier achieves $G_m = 91.58\%$. The three features, $\{20, 25, 32\}$, are evaluated because at the previous stage of feature selection, features 25 and 32 are selected as candidates for the next stage. This means that the strategy of using the best 10 feature sets at each stage is beneficial and may lead to finding feature sub-

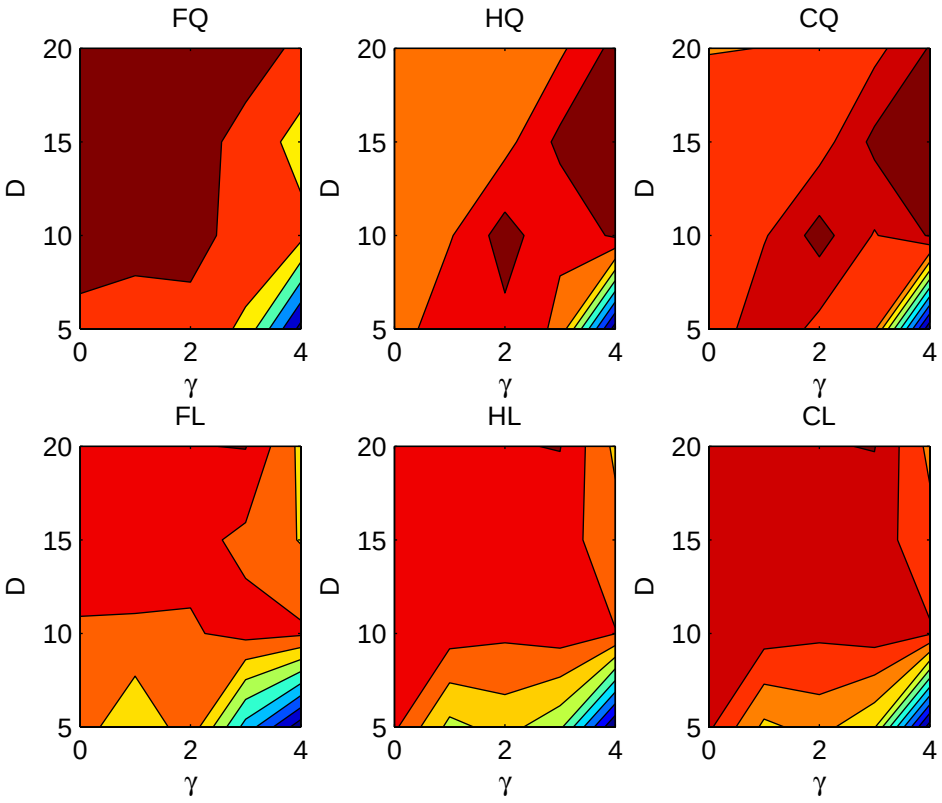

Figure 5.1: Performance of the classifiers with different RBF parameters with features 21 and 25.

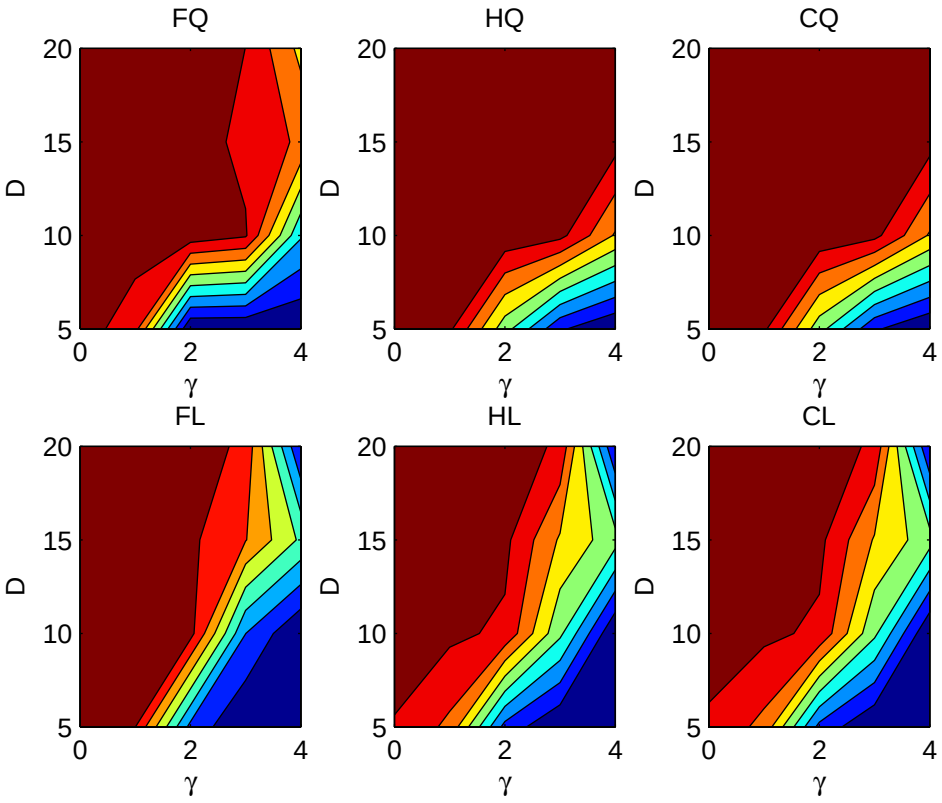

Figure 5.2: Performance of the classifiers with different RBF parameters with features 25 and 26.

sets that are more representative. If another feature selection algorithm such as sequential forward search or floating forward search were used, one would have missed the feature subset $\{20, 25, 32\}$. On the whole, I can conclude that the proposed feature selection algorithm has advantages over sequential forward search or floating forward search since it explores a larger search space that is more probable to reach a better answer. The second observation is that the proposed classifier is not affected dramatically by features that might be irrelevant and could be potentially ignored. For example, by comparing row 64 and 65, the performance of the classifier using those features are approximately equivalent even though one feature is different. Another example are rows 77 and 78. This observation shows that the classifier is robust and the noisy data do not affecting the performance of the classifier, significantly. However, this claim should be further investigated.

Table 5.3: Classifier performance for the top 10 subset of feature.

No.	Feat. #1	Feat. #2	Feat. #3	Feat. #4	Feat. #5	Feat. #6	Feat. #7	Feat. #8	Feat. #9	Feat. #10	G_m
1	18	22									88.99
2	18	39									88.90
3	18	28									88.81
4	25	26									87.43
5	21	28									87.08
6	25	32									87.07
7	21	23									86.97
8	20	42									86.92
9	27	42									86.64
10	18	23									86.62
11	20	25	32								91.32
12	20	32	42								90.97
13	18	21	22								89.60
14	18	22	23								89.17
15	20	22	42								88.99
16	18	21	23								88.99

17	25	31	32								88.83
18	18	21	28								88.79
19	20	27	42								88.57
20	18	23	35								88.57
21	18	21	22	32							89.62
22	18	21	23	44							89.35
23	18	20	22	23							89.30
24	18	21	23	35							88.97
25	18	21	23	32							88.95
26	18	21	22	35							88.95
27	18	21	22	43							88.94
28	18	21	22	42							88.93
29	18	21	22	44							88.84
30	18	20	23	35							88.80
31	18	21	22	32	43						90.00
32	18	20	22	23	42						89.39
33	18	21	22	43	44						89.30
34	18	21	22	42	44						89.15
35	18	21	22	23	42						89.10
36	18	21	22	42	43						89.08
37	18	21	22	23	32						89.07
38	18	20	21	22	23						89.06
39	18	20	21	22	23						89.00
40	18	21	22	23	44						88.90
41	18	21	22	23	32	44					90.29
42	18	21	22	32	43	44					90.23
43	18	20	21	22	23	32					90.21
44	18	20	21	22	23	32					90.21
45	18	21	22	30	33	35					90.18
46	18	21	22	23	32	43					89.86
47	18	21	22	23	33	43					89.83
48	18	21	22	32	42	44					89.78
49	18	20	21	22	43	44					89.75
50	18	21	22	30	33	44					89.74

51	18	20	21	22	32	43	44				91.58
52	18	21	22	23	31	32	43				91.30
53	18	21	22	31	32	43	44				91.14
54	18	20	21	22	23	32	43				91.11
55	18	21	22	23	32	35	43				90.66
56	18	21	22	23	32	42	43				90.12
57	18	21	22	32	42	43	44				90.03
58	18	21	22	32	33	35	42				89.48
59	18	20	21	22	42	43	44				89.47
60	18	21	22	23	33	42	43				89.46
61	18	21	22	23	33	42	43	44			90.92
62	18	21	22	32	33	35	42	44			90.44
63	18	21	22	23	33	35	43	44			90.33
64	18	21	22	23	33	35	43	44			90.33
65	18	21	22	23	33	35	42	44			90.30
66	18	21	22	32	33	42	43	44			90.29
67	18	21	22	23	33	35	42	43			90.06
68	18	21	22	32	33	41	42	44			89.78
69	18	21	22	23	33	34	35	42			89.67
70	18	21	22	23	33	34	43	44			89.44
71	18	19	21	22	23	32	33	42	43		89.71
72	18	21	22	23	33	41	42	43	44		88.94
73	18	19	20	21	22	23	33	42	43		88.69
74	18	21	22	23	33	35	41	42	43		88.41
75	18	21	22	23	33	35	41	42	43		88.41
76	18	19	21	22	23	30	33	42	43		88.36
77	18	21	22	32	33	35	42	43	44		88.33
78	18	21	22	23	33	35	42	43	44		87.97
79	18	21	22	25	32	33	42	43	44		87.94
80	18	21	22	23	33	34	35	42	44		87.92
81	18	19	20	21	22	23	32	33	42	43	90.10
82	18	19	21	22	23	25	32	33	42	43	89.72
83	18	19	21	22	23	32	33	35	42	43	89.40
84	18	19	21	22	23	30	33	35	42	43	89.24
85	18	19	21	22	23	32	33	42	43	44	89.23

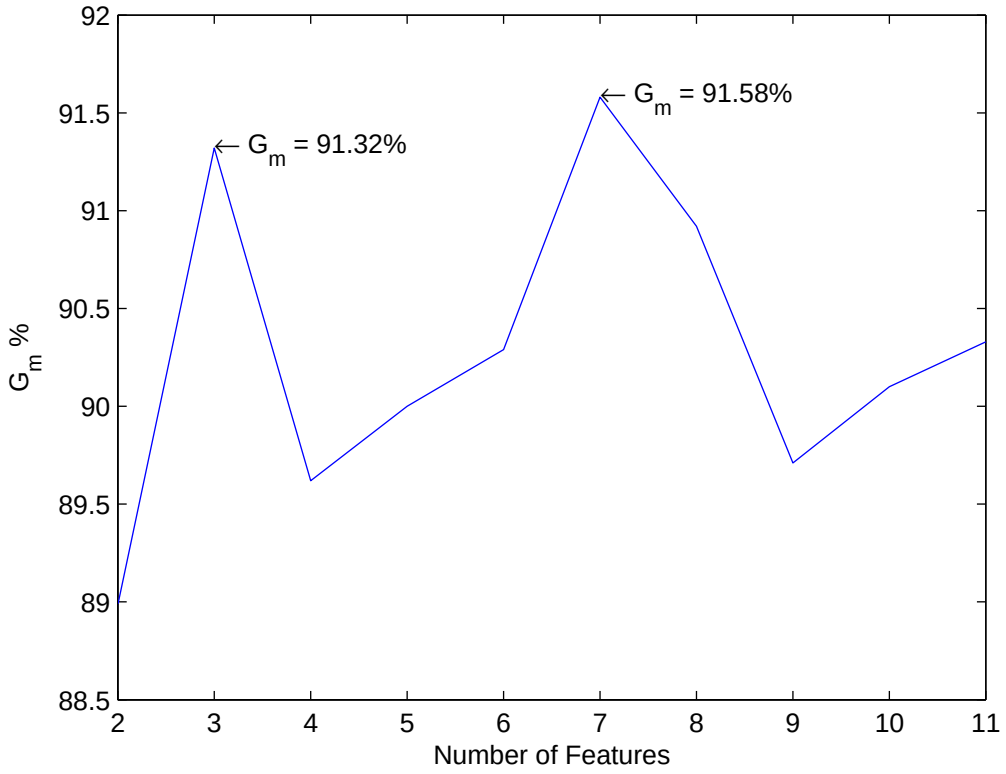

Figure 5.3: Performance of the classifier at different stages of Alg. 1 with different number of features.

86	18	19	20	21	22	23	33	41	42	43	88.97
87	18	19	20	21	22	23	30	33	42	43	88.95
88	18	19	21	22	23	30	32	33	42	43	88.92
89	18	21	22	23	25	33	34	35	42	44	88.75
90	18	19	20	21	22	23	33	35	42	43	88.71

As mentioned earlier, two parameters of the RBF are equal to default values for all the results which have been listed so far. However, since finding the optimal parameters for each classifier is not feasible, the parameters are only optimized for the best two results of

Number of Features	γ	$\boldsymbol{D}$	*SE*	*SP*	G_m
3	4.00	25	88.13	96.45	92.20
7	0.50	20	89.15	96.84	92.91

Table 5.4: Performance of the classifier after optimization of mapping parameters for three and seven features.

Table 5.2, three and seven features. I selected γ to be $\gamma \in \{0.25, 0.50, 1.00, 2.00, 4.00, 8.00\}$ and $\boldsymbol{D}$ (number of dimensions of the target space) to be $\boldsymbol{D} \in \{10, 15, 20, 25, 30, 35\}$ when having three features and $\boldsymbol{D} \in \{15, 20, 25, 30, 35, 40\}$ when having seven features. Table 5.4 shows the performance when the optimization is done as well as the values of the parameters. As seen in the table, the performance of the classifier after optimization is equal to $G_m = 92.20\%$ when using three features which is very good in spite of the fact that the classifier uses only three features of the original dataset, which has 48 features. Also, with seven features, the classifier can achieve $G_m = 92.91\%$.

5.2 Discussion and Comparison

I compared the proposed approach, LDA classification with RBF mapping which I call ***miLDR-EM***, which yields the highest performance when using three features, with some of the previously proposed methods such as *miRabela*, *MiPred*, *miPred*, *microPred* and *Triplet-SVM*. As can be seen in Table 5.5, the performance of miLDR-EM is slightly lower than microPred.

MicroPred uses a few feature selection algorithm to find the best subset of features. As a result of this, 21 features are selected as the best feature subset with $G_m = 90.84\%$. In this study, *miLDR-EM* achieves $G_m = 92.20\%$ with only three features, which is slightly lower than *microPred* but still higher than other methods. This may provide not only an

Method	Number of Features	*SE*	*SP*	G_m
Triplet-SVM	32	93.30	88.10	90.66
miRabela	40	71.00	97.00	82.99
MiPred	34	89.35	93.21	91.26
miPred	29	84.55	97.97	91.01
microPred	21	90.02	98.28	93.58
miLDR-EM	**3**	88.13	96.45	**92.20**

Table 5.5: Comparison between miLDR-EM with just three features and previously proposed methods.

Feature Selection Methods	Number of Features	*SE*	*SP*	G_m
JeffriesMatusita distance	21	83.36	99.00	90.84
Divergence and Transformed Divergence	8	67.59	99.44	81.99
miLDR-EM	3	88.13	96.45	92.20

Table 5.6: Comparison the performance of miLDR-EM and different feature selection algorithms used in microPred.

improvement on the computational tasks for classification, but also an insight on the RNA structural properties that are suitable for prediction of pre-microRNA and pseudo hairpins. Comparison between different feature selection methods that are used in microPred and the feature selection algorithm that is used in this research work is presented in Table 5.6. By comparing these results, one can state that the proposed method has a very good performance even without using any kind of imbalance learning methods. MicroPred uses few imbalance learning techniques for improving the performance after feature selection method.

The three features that miLDR-EM uses for classification are as follows: *dG*, *zG* and *NEFE*. As described previously, *dG* represents the normalized free energy of folding in sequences, normalized by the length of the sequence, *zG* is the normalized variant (z-score) for feature *dG* and *NEFE* measures the normalized ensemble free of the sequence normal-

ized by the length of the sequence. In a nutshell, it can be inferred that the identification of microRNA is merely based on minimum free energy (and its normalized z-score) and the normalized ensemble free energy. The other 45 features are much less relevant, if not irrelevant, in the prediction problem, and could eventually be disregarded. This is a very important discovery in this research work that using only three features pre-microRNA sequences can be classified.

Part IV

Conclusions and Perspectives

Chapter 6

Conclusions and Perspectives

In this thesis, it is shown that LDA classifiers can be successfully used for the classification of microRNA precursors especially when the dataset is explicitly mapped via RBF. On the whole, when kernelizing a method is not feasible or the kernelized version is not available it is shown that, the data can be explicitly mapped and then the classification can be done on the mapped data. This statement is supported by the results the classifier achieves. Also, the feature selection algorithm utilized in this method, selects only *three* features which allows the classifier to achieve a high G_m compared to previously proposed methods, while those methods use larger numbers of features. The three selected features are all related to *minimum free energy* of the microRNA precursor sequences. Also, it can be concluded that designing and implementing a proper framework for automating the experiments as well as using relational databases for storing the results are crucial, especially parallelizing the program which is desirable for researchers.

6.1 Contributions

The main contributions of this thesis are:

- Proposing a new classification scheme that combines LDA methods (FDA, HDA and CDA) with quadratic and linear classifiers for the classification of microRNA precursors.
- Comparison of using explicitly mapped data fed into the classifier and using the original data. These methods have never been used with LDR classifiers.
- Utilizing feature selection algorithm for selecting fewer features.
- Designing and implementing a framework for automating and handling a large number of experiments and using a database server for storing the results.

6.2 Future Works

There is room for continuing this research topic in the future. A few options for future work are described below:

- Generating a dataset containing new microRNA precursors as well as adding newly identified non microRNA sequences to the negative class.
- Evaluating the performance of the proposed method on a dataset containing microRNAs of other species.
- Creating a web service in which researchers can provide the potential pre-microRNA sequences and the service can decide whether the sequence is a microRNA or not using three features.

- Conducting an empirical research on effectiveness of the feature selection algorithm which is a modified version of existing feature selection methods.
- Investigating the limitations of explicit feature mapping in terms of maximum size of the dataset it can handle and comparing it with the kernelized version of some machine learning algorithms in which the kernelized version exists, such as SVM.
- Using other datasets of different applications in order to compare the idea of using explicit feature mapping.
- Working on other kernel functions rather than RBF or polynomial kernel such as hyperbolic tangent, additive Chi squared kernel and skewed Chi squared kernel.
- Investigating the possibility of using imbalance learning methods after the feature selection process.

Part V

Appendices

Appendix A

Feature Indices

Table A.1: Indices of all features in the dataset

Index	Feature Symbol	Index	Feature Symbol	Index	Feature Symbol
1	*%(C+G)*	17	*%UU*	33	*D*
2	*%AA*	18	$MFEI_1$	34	*Freq*
3	*%AC*	19	$MFEI_2$	35	*Diff*
4	*%AG*	20	*dG*	36	*dH*
5	*%AU*	21	*dP*	37	*dH/L*
6	*%CA*	22	*dQ*	38	*dS*
7	*%CC*	23	*dD*	39	*dS/L*
8	*%CG*	24	*dF*	40	*Tm*
9	*%CU*	25	*zG*	41	*Tm/L*
10	*%GA*	26	*zP*	42	*nAUb/L*
11	*%GC*	27	*zQ*	43	*nGCb/L*
12	*%GG*	28	*zD*	44	*nGUb/L*
13	*%GU*	29	*zF*	45	*BP/n_stems*
14	*%UA*	30	$MFEI_3$	46	*nAU/n_stems*
15	*%UC*	31	$MFEI_4$	47	*nGC/n_stems*
16	*%UG*	32	*EAFE*	48	*nGU/n_stems*

Appendix B

How to Set Up the Classifier

For setting up the classifier and conducting the experiments, the following instructions should be followed. Let us denote by `$root` , the absolute path of the miLDR_ER folder (eg: `C:\miLDR _ER`). `$matlabroot` denotes the location of the folder in which your Matlab is installed (eg: `C:\Program Files \Matlab \R2011b`).

Connecting Matlab to Microsoft SQL Server: Once Microsoft SQL Server is installed, you can follow the instructions for connecting Matlab to the database server.

- Adding database driver path to Matlab's classpath. This can be done by adding the following line to `classpath.txt` in `$matlabroot \toolbox \local` folder (Note: You need administrative permissions for modifying this file):

  `$root/req/jtds-1.2.5.jar` (replace `$root` with the folder path of miLDR_ER eg. `C:/miLDR _ER`)

- Choose and copy `ntlmauth.dll` from `$root \req \32bit` or `$root \req \64bit` to `$matlabroot \sys \java \jre \win64 \jre \bin` depending on your system.

Input data format The program expects the input data to have the following format:

<label><feature_1><feature_2>...<feature_d>

Creating the Tables in Database Depending on the data you want to store, a table should be created in the database which matches the result format. The source code for creating the tables is included in `$root \req\demo` .

Running the program Once Matlab is set up to be able to connect to the tables in the database, you can run the miLDR_ER program. The program is well documented and it provides the other instructions necessary for running it. The instructions include specifying the tables' schemas, specifying the tables name, loading the input dataset, how to use the parallel version, etc.

Bibliography

[1] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, Machine Learning: ECML 2004, volume 3201 of Lecture Notes in Computer Science, pages 39–50. Springer, 2004.

[2] Ethem Alpaydin. Introduction to Machine Learning. The MIT Press, 2nd edition, 2010.

[3] David P. Bartel. Micrornas: Genomics, biogenesis, mechanism, and function. Cell, 116(2):281 – 297, 2004.

[4] Rukshan Batuwita and Vasile Palade. micropred: effective classification of pre-mirnas for human mirna gene prediction. Bioinformatics, 25(8):989 – 995, 2009.

[5] Isaac Bentwich, Amir Avniel, Yael Karov, Ranit Aharonov, Shlomit Gilad, Omer Barad, Adi Barzilai, Paz Einat, Uri Einav, Eti Meiri, Eilon Sharon, Yael Spector, and Zvi Bentwich. Identification of hundreds of conserved and nonconserved human micrornas. Nat Genet, 37(7):766–770, 07 2005.

[6] Markus Brameier and Carsten Wiuf. Ab initio identification of human micrornas based on structure motifs. BMC Bioinformatics, 8(1):478, 2007.

[7] Yimei Cai, Xiaomin Yu, Songnian Hu, and Jun Yu. A brief review on the mechanisms of mirna regulation. Genomics, Proteomics & Bioinformatics, 7(4):147–154, 12 2009.

[8] Darby Tien-Hao Chang, Chih-Ching Wang, and Jian-Wei Chen. Using a kernel density estimation based classifier to predict species-specific microRNA precursors. BMC bioinformatics, 9 Suppl 12:S2, January 2008.

[9] Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. Training and testing low-degree polynomial data mappings via linear svm. Journal of Machine Learning Research, 11:1471–1490, 2010.

[10] Kevin Chen and Nikolaus Rajewsky. The evolution of gene regulation by transcription factors and microrna. Nat Rev Genet, 8(2):93–103, 02 2007.

[11] R. Duda, P. Hart, and D. Stork. Pattern Classification. John Wiley and Sons, Inc., New York, NY, 2nd edition, 2000.

[12] A. Esquela-Kerscher and F. J. Slack. Oncomirs - microRNAs with a role in cancer. Nat Rev Cancer, 6:259–269, 2006.

[13] Aurora Esquela-Kerscher and Frank J Slack. The age of high-throughput microrna profiling. Nat Meth, 1(2):106–107, 11 2004.

[14] R. Fisher. The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics, 7:179–188, 1936.

[15] Yoav Goldberg and Michael Elhadad. splitsvm: fast, space-efficient, non-heuristic, polynomial kernel computation for nlp applications. In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human

Language Technologies: Short Papers, HLT-Short '08, pages 237–240, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[16] Sam Griffiths-Jones. The microrna registry. Nucleic Acids Research, 32(suppl 1):D109–D111, 01 2004.

[17] Sam Griffiths-Jones, Russell J. Grocock, Stijn van Dongen, Alex Bateman, and Anton J. Enright. mirbase: microrna sequences, targets and gene nomenclature. Nucleic Acids Research, 34(suppl 1):D140–D144, 2005.

[18] Sam Griffiths-Jones, Harpreet Kaur Saini, Stijn van Dongen, and Anton J. Enright. mirbase: tools for microrna genomics. Nucleic Acids Research, 36(suppl 1):D154–D158, 01 2008.

[19] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. J. Mach. Learn. Res., 3:1157–1182, 2003.

[20] K Han. Effective sample selection for classification of pre-miRNAs. Genetics and molecular research : GMR, 10(1):506 – 518, January 2011.

[21] Peng Jiang, Haonan Wu, Wenkai Wang, Wei Ma, Xiao Sun, and Zuhong Lu. Mipred: classification of real and pseudo microrna precursors using random forest prediction model with combined features. Nucleic Acids Research, 35(suppl 2):W339 – W344, 2007.

[22] Seung-Jean Kim, Alessandro Magnani, and Stephen Boyd. Optimal kernel selection in kernel fisher discriminant analysis. In Proceedings of the 23rd international conference on Machine learning, pages 465–472. ACM, 2006.

[23] Miroslav Kubat and Stan Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In In Proceedings of the Fourteenth International Conference on Machine Learning, pages 179–186. Morgan Kaufmann, 1997.

[24] Harvey Lodish, Arnold Berk, Chris A. Kaiser, Monty Krieger, Matthew P. Scott, Anthony Bretscher, Hidde Ploegh, and Paul Matsudaira. Molecular Cell Biology. W. H. Freeman, 6th edition, June 2007.

[25] M. Loog and P.W. Duin. Linear Dimensionality Reduction via a Heteroscedastic Extension of LDA: The Chernoff Criterion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(6):732–739, 2004.

[26] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.R. Mullers. Fisher discriminant analysis with kernels. In Neural Networks for Signal Processing IX, 1999, pages 41–48, aug 1999.

[27] Sebastian Mika, Gunnar Rtsch, and Klaus-Robert Muller. A mathematical programming approach to the kernel fisher algorithm. In Proc. Neural Information Processing Systems, pages 591–597. MIT Press, 2001.

[28] Jin-Wu Nam, Ki-Roo Shin, Jinju Han, Yoontae Lee, V. Narry Kim, and Byoung-Tak Zhang. Human microrna prediction through a probabilistic co-learning model of sequence and structure. Nucleic Acids Research, 33(11):3570 – 3581, 2005.

[29] Kwang Loong Stanley Ng and Santosh K. Mishra. De novo svm classification of precursor microrna from genomic pseudo hairpins using global and intrinsic folding measures. Bioinformatics, 23(11):1321 – 1330, 2007.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . Journal of Machine Learning Research, 12:2825–2830, 2011.

[31] Kim D. Pruitt and Donna R. Maglott. RefSeq and LocusLink: NCBI gene-centered resources. Nucleic Acids Research, 29(1):137–140, 2001.

[32] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20, pages 1177–1184. MIT Press, 2008.

[33] L. Rueda and M. Herrera. Linear Dimensionality Reduction by Maximizing the Chernoff Distance in the Transformed Space. Pattern Recognition, 41(10):3138–3152, 2008.

[34] Alain Sewer, Nicodeme Paul, Pablo Landgraf, Alexei Aravin, Sebastien Pfeffer, Michael Brownstein, Thomas Tuschl, Erik van Nimwegen, and Mihaela Zavolan. Identification of clustered micrornas using an ab initio prediction method. BMC Bioinformatics, 6(1):267, 2005.

[35] S. Theodoridis and K. Koutroumbas. Pattern Recognition. Elsevier Academic Press, third edition, 2006.

[36] Yanqiu Wang, Xiaowen Chen, Wei Jiang, Li Li, Wei Li, Lei Yang, Mingzhi Liao, Baofeng Lian, Yingli Lv, Shiyuan Wang, Shuyuan Wang, and Xia Li. Predicting human microRNA precursors based on an optimized feature subset generated by GA-SVM. Genomics, 98(2):73 – 78, August 2011.

[37] Gary M. Weiss. Mining with rarity: a unifying framework. SIGKDD Explor. Newsl., 6(1):7–19, June 2004.

[38] Yunpen Xu, Xuefeng Zhou, and Weixiong Zhang. MicroRNA prediction with a novel ranking algorithm based on random walks. Bioinformatics (Oxford, England), 24(13):i50–i58, July 2008.

[39] Chenghai Xue, Fei Li, Tao He, Guo-Ping Liu, Yanda Li, and Xuegong Zhang. Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine. BMC Bioinformatics, 6:310, January 2005.

[40] Malik Yousef, Segun Jung, Louise C. Showe, and Michael K. Showe. Learning from positive examples when the negative class is undetermined- microrna gene identification. Algorithms for Molecular Biology, 3:2, 2008.

[41] Malik Yousef, Michael Nebozhyn, Hagit Shatkay, Stathis Kanterakis, Louise C. Showe, and Michael K. Showe. Combining multi-species genomic data for microrna identification using a naïve bayes classifier. Bioinformatics, 22(11):1325 – 1334, 2006.

[42] Zexuan Zhu, Yew-Soon Ong, and M. Dash. Wrapper-filter feature selection algorithm using a memetic framework. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 37(1):70–76, Feb. 2007.

Vita Auctoris

Navid Shakibapour Tabrizi was born in Tehran, Iran, on 1987. He graduated from Ferdowsi University of Mashhad in July 2005 with a B.Sc. degree in computer engineering. He joined computer science program at the University of Windsor in September 2010 and earned his M.Sc. degrees in September 2012. respectively.