



Faculty of Cognitive Sciences and Human Development

Prediction of Enhancer Regions using Deep Learning Techniques

Wong Yu Shiong

**Doctor of Philosophy
2018**

Prediction of Enhancer Regions using Deep Learning Techniques

Wong Yu Shiong

A thesis submitted

In fulfilment of the requirements for the degree of Doctor of Philosophy (Intelligent System)

Faculty of Cognitive Sciences and Human Development
UNIVERSITI MALAYSIA SARAWAK
2018

ACKNOWLEDGEMENT

First and foremost, I would like to deliver my uttermost gratitude to my supervisor, Dr. Lee Nung Kion, for his professional supervision and technical supports during my Ph.D. journey. It will be impossible to have my thesis done without his research experience and personal help. I really appreciate his contributions and the time spent to read this thesis. Throughout this journey, I've learned many valuable lessons from my supervisor on how to conduct research, write papers in an academic way, and enhance my understandings on academic ethics. He has inspired me the great possibilities possessed by Deep Learning. All these definitely will help me to develop as a professional in the future. This thesis work contribute as part of the Minister of Education Malaysia, Fundamental Research Grant Scheme-FRGS/SG03(01)/1134/2014(01). I would like to thank Prof. Dato' Dr. Mohd Tajuddin Abdullah, Director, Kenyir Research Institute, Universiti Malaysia Terengganu for the support and initial discussion on the Proboscis Monkey Genome annotation project. We also thank him for kind sharing of the genome data for our analysis.

I'm truly blessed by my parents who are very supportive throughout my Ph.D. course. They are always there for me and listen to me when I faced difficulties in conducting this study. I wouldn't be where I am today without my parents. Their unconditional love always encourages me to be a better person and help others in need. It is my great pleasure as well to serve Logos Hope as a port volunteer to bring knowledge, help, and hope during my studies here in Kuching. Making friends from all around the world who share the same faith as me is definitely an unforgettable memory.

Lastly, I would like to express my sincere thanks to my lab mates: Fong Pui Kwan, Kok Weiying, Shekli, best friends, and EE2 cell group for always being supportive and show

positive energy to me for the past three years. I really give thanks for all their prayers and encouragement, it meant a lot to me. I would never complete my studies with joy without them. Never forget all these achievements accomplished from my bachelor degree until the Ph.D. degree is to glorify the name of the Lord. For the Lord is able and faithful to all His servants. All glory to you, my Father in Heaven. Your Words and promises are my sources of strength throughout my Ph.D. journey. Below are three verses I kept to remind myself, nothing is impossible in the name of the Lord.

Proverbs 16:3

Commit to the Lord whatever you do,

And he will establish your plans.

Jeremiah 29:11

For I know the plans I have for you,” declares the Lord, “plans to prosper you and not to harm you, plans to give you hope and a future.

Psalms 32:8

I will instruct you and teach you in the way you should go;

I will counsel you with my loving eye on you.

ABSTRACT

Identification of transcription factor proteins binding locations (motifs) genome-wide and their specificities is a critical task for the understanding of gene regulation and associated diseases. Computational methods for motif prediction have been well studied and many tools have been proposed to target different categories of motifs. With more labelled datasets are available, machine learning techniques have become more useful for model construction. Deep-learning neural network has become a very prominent machine learning technique because of its capability to model complex datasets in different problem domains. This thesis aspires to develop machine learning systems to predict the locations and specificities of transcription factor binding sites (TFBSs) in DNA enhancer regions. In addition, the deep-learning is also employed to learn features associated with ChIP-datasets. For that, this thesis work has developed two models called DeepFinder and GMFR-CNN for motif prediction. First deep learning model called DeepFinder, which formulates the problem as enhancer classification, where an initial subset of the whole sequences can be performed by motif discovery tools and these tools select most potential candidate motifs as seed motif profile to predict TFBSs in the remaining sequences. The second deep learning model is called GMFR-CNN, which considers the importance of gapped motif features and integrates with deep convolutional neural networks (CNNs) that are able to compose higher level features from lower level ones. This thesis contributes to the problem solving in the following aspects: (1) a framework called three-stage method is proposed to tackle large search space task by using seed motif to search for TFBSs in the remaining sequences and this framework is employed in design of enhancer prediction using stacked autoencoders classifier; and (2) to take into account the significance of gapped motif feature, we proposed a novel representation to describe DNA sequence as two-dimensional gapped motif feature representation (GMFR) and implements with CNNs as

classifier to obtain highly significant gapped features. A thorough assessment of the proposed frameworks has been done using five-fold cross validation method. Empirical results indicate that DeepFinder and GMFR-CNN have good potential in identifying the locations of TFBSs and specifying the enhancer binding regions in DNA sequences. Genes and enhancer regions annotation protocol for proboscis monkey genome was introduced. Analysis of coverage has been carried among different enhancer prediction tools where GMFR-CNN has shown highest coverage as compared to the other tools. Other than that, the prior knowledge regarding motifs and enhancer sequences is advantageous to improve prediction performance.

Keywords: motif prediction, deep-learning neural network, enhancer, classification.

Penggunaan Teknik Pembelajaran Mendalam untuk Ramalan Rantau Penggalak

ABSTRAK

Pengenalpastian lokasi dan spesifikasi protein tapak mengikat faktor transkripsi (motif) dalam genom merupakan satu tugas yang kritikal untuk memahami regulasi gen dan penyakit yang berkaitan. Kaedah pengiraan untuk ramalan motif telah dikaji dan banyak alat telah dicadangkan untuk menyasarkan kategori motif yang berlainan. Melalui penggunaan set data yang banyak dan berlabel, teknik pembelajaran mesin telah menjadi lebih bermanfaat untuk pembinaan model. Rangkaian neural pembelajaran mendalam telah menjadi teknik pembelajaran mesin yang sangat popular kerana keupayaannya untuk membina model set data kompleks dalam berlainan masalah domain. Tesis ini bertujuan untuk membina sistem pembelajaran mesin untuk meramalkan lokasi dan spesifikasi tapak mengikat faktor transkripsi (TFBSs) di kawasan-kawasan DNA yang terletaknya rantau penggalak. Di samping itu, pembelajaran mendalam juga digunakan untuk mempelajari ciri-ciri yang berkaitan dengan ChIP-set data. Untuk ramalan motif, tesis ini menyampaikan dua model, iaitu DeepFinder dan GMFR-CNN. Model Pembelajaran Mendalam pertama, DeepFinder, merumuskan masalah seperti klasifikasi penggalak, dimana subset awal yang berasal daripada data DNA boleh diproses dengan menggunakan alat penemuan motif dan calon motif yang paling berpotensi akan dipilih sebagai profil tapak mengikat faktor transkripsi untuk meramalkan TFBSs dalam data yang selebihnya. Model Pembelajaran Mendalam kedua, GMFR-CNN, dimana kepentingan ciri-ciri jarak motif dititikberatkan dan diintegrasikan dalam 'deep convolutional neural networks' (CNNs), untuk pembinaan ciri-ciri yang berguna dalam klasifikasi. Tesis ini memberikan penyelesaian masalah dalam aspek-aspek berikut: (1) Rangka kerja yang dinamakan sebagai kaedah tiga peringkat telah dicadangkan untuk mengendalikan pencarian data yang lebih besar. Kaedah tiga peringkat melibatkan penggunaan profil tapak mengikat

faktor transkripsi untuk mencari TFBSs yang selebihnya dan kaedah ini direka berasaskan ramalan penggalak dengan menggunakan klasifikasi 'stacked autoencoders'; (2) Perwakilan baharu yang mengambil kira kepentingan ciri-ciri jarak motif diperkenalkan untuk menggambarkan urutan DNA sebagai dua dimensi penggambaran ciri-ciri jarak motif (GMFR) dan melaksanakan pembelajaran dengan menggunakan CNNs sebagai alat klasifikasi untuk memperolehi ciri-ciri jarak motif yang penting. Penilaian yang wajar telah dijalankan untuk rangka kerja berikut dengan menggunakan lima validasi silang. Hasil kajian menunjukkan bahawa DeepFinder dan GMFR-CNN mempunyai potensi yang baik dalam pengenalan lokasi TFBSs dan penentuan rantau mengikat penggalak dalam urutan DNA. Protokol anotasi gen dan kawasan penggalak dalam genom kera bekantan diperkenalkan. Analisis liputan telah dijalankan dalam kalangan alat ramalan penggalak dan GMFR-CNN telah menunjukkan liputan tertinggi berbanding dengan alat lain. Selain itu, pengetahuan sedia ada mengenai motif dan urutan penggalak adalah satu kelebihan untuk meningkatkan prestasi ramalan.

Kata kunci: ramalan motif, rangkaian neural pembelajaran mendalam, penggalak, klasifikasi.

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iv
ABSTRAK	vi
TABLE OF CONTENTS	viii
LIST OF TABLES	xiii
LIST OF FIGURES	xv
LIST OF ABBREVIATIONS	xx
CHAPTER 1: INTRODUCTION	1
1.1 Overview	1
1.2 Motif Prediction in Enhancer Regions	2
1.2.1 Background	4
1.2.2 A Brief Introduction to Deep Learning	9
1.2.3 TFBS Identification and Challenges	10
1.3 Motivation	11
1.4 Problem Formulation	13
1.5 Objectives	15
1.6 Technical Contributions	17
1.7 Scope of the Study	19

1.8	Thesis Structure.....	20
CHAPTER 2: LITERATURE REVIEW		23
2.1	Overview	23
2.2	Gene Expression.....	23
2.2.1	From DNA to RNA (Transcription).....	24
2.2.2	From RNA to Protein (Translation)	26
2.2.3	Transcription Factor (TF).....	27
2.2.4	Motif Representations	29
2.3	A Brief Introduction to Deep Learning	34
2.3.1	Artificial Neural Networks (ANNs).....	36
2.3.2	Multilayer Backpropagation Neural Networks	38
2.3.3	Deep Learning Neural Network (DLNN)	42
2.3.3.1	Stacked Autoencoders	43
2.3.3.2	Convolutional Neural Network (CNN)	49
2.4	Using Whole vs Partial Input Set	54
2.4.1	Word-based Approaches	57
2.4.2	Probabilistic Approaches.....	63
2.5	Ensemble Approaches	66
2.6	Deep Learning Approach	70
2.7	Summary	74

CHAPTER 3: DNA MOTIF PREDICTION USING FEATURED-BASED AND

DEEP LEARNING APPROACHES	76
3.1 Introduction	76
3.2 Description of Proposed Framework.....	77
3.2.1 Motif Features	81
3.2.2 Motif Similarity Measures	82
3.2.3 Motif Clustering Technique	84
3.2.4 Deep Learning Classifier Model	84
3.3 Performance Evaluation Metrics	87
3.3.1 Datasets	89
3.3.2 Experiment Setup	90
3.3.3 Results and Comparisons	91
3.4 Discussion	96
3.5 Conclusion.....	98

CHAPTER 4: AN INTEGRATION OF GAPPED MOTIF FEATURE

REPRESENTATION AND DEEP LEARNING APPROACH FOR	
ENHANCER PREDICTION	99
4.1 Introduction	99
4.2 Description of Proposed Framework.....	100
4.2.1 Sequence Encoding	101
4.2.2 CNN Architecture	104

4.3	Performance Evaluation Metrics	106
4.3.1	Datasets	107
4.3.2	Experiment Setup	107
4.3.3	Results and Comparisons	109
4.4	Discussion	120
4.5	Conclusion.....	121
 CHAPTER 5: GENES AND ENHANCER REGIONS ANNOTATION		
	PROTOCOL FOR PROBOSCIS MONKEY GENOME.....	122
5.1	Introduction	122
5.2	Background	123
5.3	Methods.....	126
5.3.1	Annotation Using MAKER.....	129
5.3.2	Enhancer Prediction	130
5.3.3	Intergenic Regions for Enhancer Identification	130
5.4	Results	133
5.4.1	Analysis of Coverage	134
5.5	Discussion	136
5.6	Conclusion.....	138
 CHAPTER 6: CONCLUSION AND FUTURE WORKS		
6.1	Contributions of Thesis	139
6.2	Summary	141

6.3	Limitations	143
6.4	Future Works.....	145
REFERENCES		147
APPENDICES		173

LIST OF TABLES

	Page
Table 2.1 IUPAC nucleotide code for eleven possible subsets of the four bases.	32
Table 2.2 Regular expression of aligned sites (Bailey, 2008).....	33
Table 3.1 TFBSs obtained from UCSC Genome Table Browser.....	90
Table 3.2 Sequence logo for top 3 motifs using MEME, BioProspector, MDscan, and MotifSampler.....	92
Table 3.3 Comparisons of precision, recall, and f -measure between MAST, FIMO, and DeepFinder (DF).	93
Table 3.4 Comparisons of false discovery rate (FDR), accuracy, and Matthews correlation coefficient (MCC) between MAST, FIMO, and DeepFinder (DF). ...	94
Table 3.5 One way ANOVA test of MAST, FIMO, and DeepFinder (DF).....	94
Table 4.1 Comparison in terms of AUC between different parameters for configuring GMFR-CNN using ELK1 as an example.....	106
Table 4.2 Ten ChIP-seq datasets, their DeepBind model ID, and the number of sequences.....	107
Table 4.3 Parameters used by GMFR-CNN, DeepBind, and LS-GKM.	109
Table 4.4 Average precision rates of LS-GKM, DeepBind, and GMFR-CNN.	110
Table 4.5 Average recall rates of LS-GKM, DeepBind, and GMFR-CNN.	111
Table 4.6 Average f -measure rates of LS-GKM, DeepBind, and GMFR-CNN.	112
Table 4.7 Average accuracy of LS-GKM, DeepBind, and GMFR-CNN.	112
Table 4.8 Average AUC for LS-GKM, DeepBind, and GMFR-CNN.....	113
Table 4.9 One way ANOVA test of LS-GKM, DeepBind, and GMFR-CNN.....	113
Table 5.1 Parameters used by different prediction tools.	133

Table 5.2	The number of predicted enhancers obtained from five prediction tools.....	133
Table 5.3	The number of overlapped and merged regions by a different number of enhancer prediction tools for Chromosome 18 of the proboscis monkey.....	134
Table 5.4	A comparison of coverage between CSI-ANN, DeepBind, LS-GKM, GMFR- CNN, and iEnhancer-2L.....	135

LIST OF FIGURES

	Page
Figure 2.1	Gene expression. (National Institutes of Health) 24
Figure 2.2	Regulatory elements of transcriptional regulation. (adapted from the webpage https://www.boundless.com/biology/textbooks/boundless-biology-textbook/gene-expression-16/eukaryotic-gene-regulation-113/transcriptional-enhancers-and-repressors-45811684/)..... 25
Figure 2.3	Translation. (adapted from the webpage http://www.bbc.co.uk/bitesize/higher/biology/cell_biology/rna/revision/2/) 27
Figure 2.4	Enhancer boosts transcriptional regulation, bound by transcription factors and mediator proteins. (adapted from the webpage https://www.boundless.com/biology/textbooks/boundless-biology-textbook/gene-expression-16/eukaryotic-gene-regulation-113/transcriptional-enhancers-and-repressors-458-11684/)..... 29
Figure 2.5	PWM and LOGO representation of the nine motifs (Bailey, 2008). 31
Figure 2.6	A nonlinear model of a neuron..... 37
Figure 2.7	The layout of a multilayer neural network for the case of a single hidden layer..... 39
Figure 2.8	An architecture of multilayer neural network with two hidden layers where the signal flow is in the forward direction..... 40
Figure 2.9	A simple autoencoder with three layers (input, hidden, and output layer). ... 45
Figure 2.10	First denoising autoencoder is trained to obtain primary features. 47
Figure 2.11	Second denoising autoencoder. 48
Figure 2.12	Secondary features as raw input to a logistic classifier..... 48

Figure 2.13	Combined all three layers together to become stacked autoencoder.	49
Figure 2.14	(a) A sparse connectivity between neurons in adjacent layers. (b) A full connectivity between neurons in adjacent layers. (c) A feature map that comprised of three hidden neurons.	50
Figure 2.15	A simple illustration of convolution.....	51
Figure 2.16	A convolutional neural network for image processing such as digit recognition.....	52
Figure 3.1	An overview of DeepFinder framework. (a) We randomly partitioned the dataset into five subsets. (b) Apply four <i>de novo</i> motif discovery tools on the partial input sequences E to predict putative motifs and respective binding sites. The top three motifs of four tools were clustered according to the location of the putative binding sites. Those binding sites that are overlapped by at least 1 bp or located next to the other are considered belonging to the same cluster. The pairwise similarity score is computed among the clusters, <i>k</i> -medoid clustering generates two clusters, with each cluster, added one hundred bases to both directions of binding sites and randomized with negative controls. The sequences are then converted into representation using the 76 designed features. The illustrated figure in details is presented in Section 4.2.2. (c) Stacked autoencoder, a deep learning technique is used to train the normalized cluster and only one cluster is chosen to perform the next task based on the size of the clusters generated. Those with bigger size cluster is constructed as a classifier to predict in the remaining DNA sequences.....	79

Figure 3.2	An overview of clustering technique and motif similarity measure. (a) The binding sites of top three motifs are accumulated after obtained from the four <i>de novo</i> motif discovery tools. (b) Those overlapped binding sites by at least 1 bp or next to one another based on their locations are grouped under the same cluster. (c) The similarity measure is computed among the clusters and populates their scores into distance matrix for the purpose of <i>k</i> -medoid clustering later on.....	83
Figure 3.3	An example of how similarity score is calculated using three motif pairs. ...	83
Figure 3.4	The neural network resides within DeepFinder.	85
Figure 3.5	A visual description of <i>TP</i> , <i>FN</i> , <i>FP</i> , and <i>TN</i> on TFBS binding region and non-TFBS binding region prediction.....	89
Figure 3.6	Precision values for each of the subsets.	95
Figure 3.7	Recall values for each of the subsets.....	95
Figure 3.8	Overall <i>f</i> -measure, precision, recall, and accuracy for ten data sets.	96
Figure 3.9	Average precision, recall, <i>f</i> -measure, false discovery rate (FDR), and accuracy for MAST, FIMO, and DeepFinder.	96
Figure 4.1	The computation of probabilistic weight values.	103
Figure 4.2	GMFR convolved by 3×3 kernel size motif detector.....	103
Figure 4.3	An architecture of GMFR-CNN.....	105
Figure 4.4	Summary graph for the ten datasets using LS-GKM, DeepBind, and GMFR-CNN.....	114
Figure 4.5	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CEBPB.	115

Figure 4.6	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CEBPD.	115
Figure 4.7	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CTCF.	116
Figure 4.8	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for ELK1.	116
Figure 4.9	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for EP300.	117
Figure 4.10	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for FOXA2.	117
Figure 4.11	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for HNF4A.	118
Figure 4.12	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for TBP.	118
Figure 4.13	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for TFAP2A.	119
Figure 4.14	Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for YY1.	119
Figure 5.1	Genome sequencing and assembly pipeline of the proboscis monkey genome.	125
Figure 5.2	Enhancer annotation pipeline.	126
Figure 5.3	BEDTools merge and intersect. (a) Merging two sequences that overlapped by at least 1 bp or next to the other. (b) The intersection of DeepBind to LS-GKM and GMFR-CNN.	129

Figure 5.4	Both genes are in forward strand.....	131
Figure 5.5	Both genes are in reverse strand.....	132
Figure 5.6	Both genes are in opposite directions where the gene on the left is in reverse strand and the gene on the right is in forward strand.....	132
Figure 5.7	An example of the calculation of coverage percentage.....	135

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
ChIP-Chip	Chromatin Immunoprecipitation with DNA Microarray
ChIP-Seq	Chromatin Immunoprecipitation with Massively Parallel DNA Sequencing
CNN	Convolutional Neural Network
CSI-ANN	Chromatin Signature Identification by Artificial Neural Network
DL	Deep Learning
DLNN	Deep Learning Neural Network
DNA	Deoxyribonucleic Acid
EST	Expressed Sequence Tag
FIMO	Find Individual Motif Occurrences
GMFR-CNN	Gapped Motif Feature Representation-Convolutional Neural Network
LS-GKM	Large Scale-Gapped <i>K</i> -mers
MAST	Motif Alignment and Search Tool
MEME	Multiple Expectation Maximization for Motif Elicitation
ML	Machine Learning
PWM	Position Weight Matrix
RNA	Ribonucleoc Acid
SAE	Stacked Autoencoder
TF	Transcription Factor
TFBS	Transcription Factor Binding Site

CHAPTER 1

INTRODUCTION

1.1 Overview

Transcription factor binding motifs are short DNA segments which have degenerative feature patterns that defined the binding preference of transcription factors. The defined motifs such as positive weight matrices can be used to predict longer DNA sequences in the genome. Thus, motif discovery or identification is a crucial task in unravelling the mechanism of gene expression, notably enhancer prediction. It is an extremely difficult and expensive task, especially when conducted in wet-lab despite its high accuracy (Leung et al., 2010). In this study, an improved three-stage method known as DeepFinder and DNA motif representation known as gapped motif feature representation-convolutional neural network (GMFR-CNN) were introduced where both methods implemented deep learning techniques for motif discovery in enhancer prediction.

Computational approaches provide alternatives to tackle the degenerative sequence patterns of binding motifs in an efficient way. In this thesis, *deep learning* (DL) techniques were implemented to improve the search outcome because it can extract meaningful features and learn the relationship between complex data (Najafabadi et al., 2015). Raw data is used as input feature without being heavily engineered by human experts. This raw information will eventually produce meaningful features that are important to differentiate between enhancer and non-enhancer for example. All these learning processes are carried out by the deep learning neural network. In this study, we found out that features such as dinucleotide repeat motifs can be used in combination with k -mer to perform the classification and achieved 97% in terms of

accuracy. We also addressed the issues of binding sites prediction in enhancer sequences and how we improved the prediction using deep learning techniques. *Deep learning* (DL) is a branch of *machine learning* (ML), that has been introduced with the purpose of progressing machine learning towards to one of its original goals which are *Artificial Intelligence* (AI) (Bengio, 2009). In this way, a deep learning classification model of motif prediction can be built automatically.

This chapter first presents the background and limitations of current TFBSs prediction in enhancer regions. After this, the research motivation is introduced. Next, the problem formulation and objectives are conferred. The chapter is then followed by a list of main contributions developed throughout the study. Finally, the thesis structure is briefly described at the end of this chapter.

1.2 Motif Prediction in Enhancer Regions

In molecular biology and computer science, motif discovery or also known as motif identification in DNA sequences is one of the most challenging problems among the practitioners (Yu, Huo, & Feng, 2016). Thus, the identification of motifs, especially transcription factor binding sites (TFBSs) in DNA sequence for transcription factors (TFs) is a crucial task in understanding the mechanism of gene expression. TFBS is referring to the binding site where it bound by TF in enhancer regions. It can be referred as the binding site of enhancer-related binding factors such as P300 or CREBBP. The motif can be considered as a representation of a set of TFBSs aligned together. The term enhancer prediction and identification of TFBSs can be used interchangeably. The challenge in motif discovery can be formulated as finding an unknown pattern that occurs repeatedly in a given set of sequences.

For example, a pattern of l letters that occurs in every sequence, therefore, a list of all l -letter patterns that appear in the sequence provides the alternative to motif identification. Unfortunately, with the mutation, insertion or deletion of nucleotides, the task of finding the motif patterns is even more challenging (Makalo, 2016).

DNA functional elements which include promoter, enhancer, and silencer located at gene regulatory regions orchestrate transcriptional regulation in multicellular eukaryotes (Kleftogiannis, Kalnis, & Bajic, 2015). They are key elements of different biological processes which play a central role in determining physiological signals, organ development and tissue differentiation, nutrient and environment stress responses. The enhancer is a cis-regulatory element that works collaboratively with promoters and silencers to control gene transcription (Jin, Li, Ren, & Natarajan, 2011). In order to determine enhancer function, a description of specific internal sequence features is needed to fully understand the mechanism. Previous studies used sequence conservation, colocalization of previously characterized TFBSs from databases such as TRANSFAC (Matys et al., 2003) or JASPAR (Bryne et al., 2008) or a combination of both to predict enhancers. The assessment by Su et al. (2010) have shown that some of the existing approaches such as MSCAN (Johansson, Alkema, Wasserman, & Lagergren, 2003) and MCAST (Bailey & Noble, 2003) were successful in identifying enhancers in *Drosophila*, however, only a few generalized to mammalian systems. Other than that, detecting novel enhancers using word frequency profile similarity between pairs of sequences can be susceptible to noise when training on small numbers of enhancers (Leung & Eisen, 2009). In this study, we focus on enhancers that are situated at a great distance from the gene they regulate. Hence, the identification of distal regulatory elements is an extremely difficult task in the bioinformatics field, especially in large vertebrate genomes. This is due to the regulatory complexes formed by what number, order, orientation, and spacing, thus,

increases the level of difficulty (Guo & Huo, 2017). Enhancers are conventionally identified by an inserted DNA sequence that can activate transcription regardless of its orientation or location relative to the promoter using a reporter gene system. These elements are operated by acquiring sequences-specific transcription factors (TFs) and co-activator complexes (Maston, Evans, & Green, 2006; Tjian & Maniatis, 1994). This method of enhancer identification involved multiple experiment procedures that may be constrained by the different cellular conditions. With the advent of high-throughput technologies, such as chromatin immunoprecipitation followed by microarray (ChIP-chip) or massive parallel sequencing (ChIP-Sequencing or ChIP-Seq), computational approaches using supervised or unsupervised machine learning (ML) algorithms have been developed for genome-wide enhancer prediction. These DNA motifs are short, recurring sequence patterns in a set of input DNA sequences that are presumed to have a biological function. These patterns indicated sequence-specific binding sites for proteins such as nucleases and transcription factors. Diseases and cancer are closely related to genetic variation or disruption in enhancers (Mansour et al., 2014; Lovén et al., 2013). Therefore, the identification of such motifs can be benefited to medical research especially in genomics area.

1.2.1 Background

The human genome contains every information needed to understand the biological properties and how it could manifest itself in every biological stage until it is fully grown as a human being and till the very last breath of the organism. In different cellular conditions, regulatory proteins might activate or inhibit particular functions in order to sustain the biological environment of the living organism. Currently, there are at least 26 billion base pairs (bp) representing the various genomes available in the server of the National Center for

Biotechnology Information (NCBI) (Ezziane, 2006). Besides the human genome with about 3 billion bp, many other species have their complete genome available there. There is a need to carefully study the genome available in order to unveil the properties involved in every mechanism such as gene expression. This is because the ultimate goal is to determine how genes and the proteins they encode function in the intact organism and one of the most direct ways is to find out what a gene does by seeing what happens to the organism when that gene is missing, repressed, or altered. Cohen (2004) explained the needs of biologists to utilize and help interpret the vast amounts of data that are constantly being gathered in genomic research. Bioinformatics and computational biology are concerned with the use of computation to understand biological phenomena and to acquire and exploit biological data, increasingly large-scale data (Gusfield, 2004). The advantages of computational techniques are promising and this is why DNA sequence analysis is attractive to computer scientists because of the availability of digital information as well as the effectiveness of computer algorithms.

Regulation of gene expression in eukaryotes requires precise spatial and temporal coordination of a multitude of general and specific transcription factors at cis-regulatory elements, including enhancers, silencers, insulators, and promoters (Heintzman & Ren, 2007). The recognition and binding of these sequences by TFs occur within the context of chromatin, whose dynamic structural characteristics play a significant role in regulating gene expression (Mellor, 2005). Sequence-specific transcription factors play a critical role in regulating transcription by recruiting coactivators and promoting the formation of the Pre-Initiation Complex (PIC) (Heintzman & Ren, 2007). The enhancer can be predicted through the binding of TF on the sequence at a specific area called TFBS that triggered the transcription process. Unfortunately, TFs bound enhancer regions are distal from the location where it initiates the biological process which makes the prediction even more challenging compared to promoter

prediction where promoters are always located near transcription start sites of genes and upstream in the DNA. Consequently, many investigations have focused on the discovery of direct targets of TF binding. TF consensus binding motifs are highly degenerate and caused sequence-based computational methods to predict many thousands of binding sites for a given TF, however, only a fraction of TFBSs may be biologically relevant. Indeed, even binding sites for which the cognate TF has a very high affinity *in vitro* are not necessarily bound *in vivo*, consistent with our understanding of mechanisms underlying tissue-specific programs of gene expression. Conversely, TF targets may not contain consensus binding motifs, the TFs are binding to uncharacterized motifs or through cooperation with additional factors (Heintzman & Ren, 2007). The development of technologies like ChIP-chip enabled the rapid and direct biochemical purification of DNA sequences bound by TFs in the genome *in vivo* and the subsequent generation of target maps and transcriptional regulatory networks (Heintzman & Ren, 2007).

This also proved that computational identification and annotation of motifs in the genome, including genes and the regulatory elements, is one of the active research areas in bioinformatics due to the high efficiency of the algorithms used. It is always a major challenge in genomics and computational biology especially in identification and discovery of regulatory elements using computational algorithms. This is due to the variable length of the motifs which is fairly short in between 5 to 20 bp. They are known to reappear in different genes or several times within a gene (Rombauts, Déhais, Van Montagu, & Rouzé, 1999). The use of *artificial neural networks* (ANNs) and machine learning (ML) techniques are starting to get popularity in this area due to their effectiveness and efficiency of the algorithms. For example, in the medical field, ANNs were used as a data-mining tool to predict and diagnose the occurrence of breast cancer (Ezziane, 2006). In addition to that, the implementation of machine learning

algorithms also can be seen in cancer diagnosis especially breast cancer and colorectal cancer (Karplus, 2012).

In the post-genomic era, research in bioinformatics has been overwhelmed by the experimental data (Tan & Gilbert, 2003). The huge amount of data can be processed using machine learning techniques to obtain the meaningful information or patterns that can be used to describe its nature. Tan and Gilbert (2003) performed an empirical comparison of rule-based learning systems (decision trees, one rule, decision rules), statistical learning systems (naive Bayes, instance based, support vector machine (SVM), and artificial neural networks) and ensemble methods (stacking, bagging and boosting) on some available data of *Escherichia coli*, Yeast, Promoters, and HIV. They have reported a comparison of different supervised machine learning techniques in classifying biological data. However, they conferred that none of the single methods could consistently perform well over all the datasets even today the work by Klepper and Drabløs in the year of 2013 also supported their findings. Moreover, their work also concluded that combined methods practically perform better and more reliable than the individual ones. This has shown that there is indeed a need for a better prediction tool since the biological data are widely available for experimental purposes. Other than that, the importance of gapped motif features in motif prediction has not been taken seriously in deep learning prediction model, thus, the work by Lee (2016) has inspired the implementation of gapped features in deep learning approach. In bioinformatics, the transformation of biomedical big data into valuable knowledge has become one of the most important challenges especially in the era of big data. The promising results of deep learning beat benchmark performances in many areas such as image and speech recognition. Hence, the popularity of deep learning in bioinformatics has begun to rise up demonstrates state-of-art performance in various fields (Min, Lee, & Yoon, 2017).

Deep learning plays a vital role that helps to bring artificial neural network closer to its original goal which is Artificial Intelligence (AI). The deep learning algorithm is a procedure that is capable of making an accurate prediction by learning past observations using deep learning techniques. This can be seen where the model is able to “see the world” which also known as inputting raw data and their option is highly problem-specific. One of the examples is MNIST digit classification. This includes some processing steps that involve a computer or machine to learn the features of a class or multiple class from a set of training data that comprised of positive and negative controls. Next, after the learning process, the learned model can predict new incoming data based on the data previously learned in the training process. For example, the model is able to differentiate between the digit “1” and “7” accurately when there is a new incoming data that it has never before seen. The characteristics and knowledge of the training data are obtained throughout the learning process. Therefore, this makes our prediction incredibly accurate and promising. It even fully implemented in reading cheque and handwritten signature in the bank industry (Miah, Yousuf, Mia, & Miya, 2015). Deep learning is a powerful technique for processing raw data in a more abstraction way where conventional machine learning techniques might lack this capability. Major advances have been made by deep learning, for example, Google DeepMind succeed in beating its opponents using AlphaGO in the game of GO (Gibney, 2016). Besides, an advancement in healthcare technologies also built by DeepMind Health to improve the efficiency of the medical field. The popularity of deep learning allows us to gain insight from the data in order to extract important features that are a matter of the case in hand.

1.2.2 A Brief Introduction to Deep Learning

Deep learning has gained popularity in image classification and voice recognition. This can be shown by Facebook photo tagging system as well as Apple Siri voice recognition technology (Jones, 2014). The use of this technology has brought advancement to the accuracy and efficiency of image classification and speech recognition. This is why we are now interested in implementing this technology in bioinformatics field and there are some previous works have proved successes in this area. The series of tasks involved deep learning is carried out by the algorithms where the algorithms are able to do the learning process as well as the classification and prediction part effectively.

Unlike any other conventional machine learning techniques, deep learning (DL) is a division of machine learning (ML) that is renowned for its learning ability by modelling high-level abstractions features based on lower level ones with the help of a set of algorithms (Bengio, 2009). An artificial neural network that performed the task is called deep learning neural network (DLNN). The neural network is capable of learning simple and non-linear modules in the multi-level data representation, in other words, deep learning can process raw data more efficiently compared to other machine learning methods. The representation is amended at every level to obtain higher level or more abstract features from lower level features in the form of raw data (LeCun, Bengio, & Hinton, 2015). Conventional machine learning approaches are not robust for raw data transformation because they required careful engineering and consideration of domain expertise to design a feature extractor. Hence, this issue is handled by deep learning that the features in each layer were learned using a general-purpose learning procedure and do not need to be designed by human engineers (LeCun, Bengio, & Hinton, 2015).

The capability of deep learning techniques is a complete breakthrough in discovering complex structures in high-dimensional data. This has brought major improvement in solving problems that are applicable to science, medicine, and business. Recently, deep learning has shown incredible performance in drug discovery (Gawehn, Hiss, & Schneider, 2015) and breast cancer detection (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013). There are few examples of deep learning such as Deep Belief Networks (DBN), Convolutional Neural Network (CNN), Deep Boltzmann Machines (DBM), Recurrent Neural Networks (RNN), Stacked Autoencoders (SAE), and etc.

1.2.3 TFBS Identification and Challenges

The first challenge is the degenerative feature of TFBSs, which naturally occurred between transcription factors and their respective binding sites. This feature is often adapted through evolutionary reshuffling. Highly-degenerate TFBSs are unlikely to bind TFs with sufficient affinity to attain specificity. However, this feature is not well understood until today. For example, the recognition of binding site by the transcription factor is understood, however, the reason why these potential target sites for each transcription factor occur many times in the genome is not known. Also, the accurate identification of the functional binding sites by the proteins in a cellular environment.

The second challenge is related to the length of TFBSs. A TFBS is usually 5-20 nucleotides or bases in length and randomly occur in enhancer regions that are relatively long, from 50 to 1500 nucleotides. Moreover, TFs are sequence tolerant despite variations at many positions of the TFBSs in most cases. The accuracy of genome-scale prediction is therefore affected by this tolerance of sequence variations. These two challenges have made the enhancer

prediction approaches generated a high number of false positives. This means predicted TFBSs that are similar in terms of their constituents are being treated as real TFBSs even though they have no biological function in gene expression.

Next, the conservation in certain positions of the TFBSs throughout species evolution contributes to the third challenge. This can be observed when to align TFBSs together, a pattern or motif is discovered. Due to this reason, motif identification can be viewed as a pattern recognition problem. Unfortunately, it cannot be used as a conventional pattern recognition problem because even a match pattern could not guarantee that it is a putative binding site.

Using computational approach to predict motifs does not guarantee that the identification of enhancers is true positives. Most computationally predicted TFBSs are not accessible for TF binding and this could be caused by chromatin structure or local epigenetic properties. The experimentally validated humans' enhancer set is not large and the methods to assess the correctness of predictions can be done by linking the predicted enhancers to target genes.

1.3 Motivation

The prediction of TFBSs that reside within enhancer region plays a vital role in determining the enhancer location. TF that binds to the TFBS entails that enhancer is not far from the bound region. This contributes to the motivation of this research by introducing deep neural network that is well-known for its classification capability in the previous works such as DeepBind (Schmidhuber, 2015; Alipanahi, Delong, Weirauch, & Frey, 2015). Deep learning has achieved astonishing performance especially in image recognition, speech recognition, and

text document classification (LeCun, Bengio, & Hinton, 2015). It began to shed light in this area which contributes to a higher accuracy level that has never been accomplished this far.

Thus, in our first project presented in Chapter 4, we utilize the benefits of the three-stage method where it can avoid exhaustive searching of motifs using only a partial set of data to perform motif search in the remaining sequences. We employed the data partially and used the subset to search respective binding sites in the remaining sequences. This is because a partial dataset is said to be sufficiently used to build seed motifs and more will only increase the predictive rates marginally. Other than that, the current three-stage approach also yields a high number of false positives, it can be enhanced using a computational approach such as k -medoids clustering technique and stacked autoencoders which implement clustering algorithm and deep learning technique to extract only meaningful information from the training set. The clustering technique used is to allow us to obtain two groups of motifs: similar and dissimilar motifs.

As for our second project which stated in Chapter 5, we employ the strength of gapped motif features in motif prediction to convert DNA sequence into two-dimensional representation called gapped motif feature representation (GMFR) and implement it on top of deep convolutional neural network to capture meaningful features that can be used to distinguish enhancer and non-enhancer regions. We have realized some previous works that employed gapped k -mer features have obtained high-performance results, however, the modelling of gapped motif feature has not been practised in any of the deep learning models yet. The implementation of deep convolutional neural networks in motif prediction has shown promising results, presumably due to the capability of composing higher order features from lower order features. This is seen in the work of MNIST digit classification task (Krizhevsky, Sutskever, & Hinton, 2012).

Throughout this thesis, we provide valuable insights about the potential of employing deep learning technique in bioinformatics to whom may want to conduct research in this field or any other related areas in the future. Deep learning is capable of learning the meaningful features that can be used to identify or differentiate between multiple elements in our case is enhancer and non-enhancer regions. The input needed for deep learning is raw data without the need of being heavily engineered by human experts. This technique contributed in many areas such as breast cancer detection or diagnosis (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013) and skin cancer detection (Esteva et al., 2017).

1.4 Problem Formulation

Currently, the comprehensive and experimentally validated enhancer set for humans or other species is not sufficient. It is challenging to identify whether the enhancers are real based on computationally predicted enhancers. The correct enhancers can be validated by linking the predicted enhancers to their target genes, however, this is the most difficult task complementary to computational enhancer prediction.

In this study, the problem of enhancer prediction is formulated as follows. First, the location of enhancers can be in few thousands of bases or hundred thousands of bases to the genes they initiate (He, Chen, Teng, & Tan, 2014). Therefore, the identified enhancer targets based on their relative location to enhancers are likely to be false positives because the methods that perform this task hypothesized that an enhancer interacts with its neighbouring mRNA promoter.

Second, the search space is large and the overlap between putative enhancer identified tends to be very small, thus, an effective general enhancer classifier is unlikely to be created

due to a small number of training set samples used by some of the current computational approaches (Liu, Li, Ren, Bo, & Shu, 2016). It is important to employ sufficient amount of data to be treated as training set because in this way the significant features used to differentiate between enhancer and non-enhancer can be learned.

Third, only a small number of data used by most of the computational approaches which derived from histone modification marks. There are several tools which combine different numbers and combinations of histone modification marks to predict enhancers such as Chromia (Won, Chepelev, Ren, & Wang, 2008), CSI-ANN (Firpi, Ucar, & Tan, 2010), RFECS (Rajagopal et al., 2013), DELTA (Lu, Qu, Shan, & Zhang, 2015), ChromaGenSVM (Fernández & Miranda-Saavedra, 2012), and DEEP-ENCODE (Kleftogiannis, Kalnis, & Bajic, 2015). In spite of the simplicity of features derived from only one type of data and only a finite number of chromatin modifications that have been examined, these approaches are facing challenges in predicting enhancers systematically even though these tools are capable of identifying thousands of enhancers in the human genome. In addition, it is infeasible for enhancers to be fully captured by an exhaustive catalogue of chromatin modifications and transcription factors.

Fourth, the use of the whole dataset in the motif finding is not necessary because only a partial set of input sequences is adequately used for the purpose of motif finding. This has been shown in the work of Zia and Moses (2012). In this way, seed motifs are formed to scan in the remaining sequences where those motifs which approximate the real ones are likely to be found. The traditional three-stage method produced a high number of false positives, therefore, a computational method like clustering technique need to be implemented in the post-processing stage.

Fifth, most of the computational tools are considered only the independent features possessed by the motifs or transcription factor binding sites (TFBSs). It is utmost important to take into account the dependency features as well. It is highly recommended that in order to enhance prediction, it is utmost important to fully employ the benefits of both independent and dependent features possessed by those motifs (Santolini, Mora, & Hakim, 2013). Other than that, the inclusion of gap within motif features too can further improve the predictive rate of the computational tools (Ghandi, Lee, Mohammad-Noori, & Beer, 2014). This is because the transcription factor did not fully bound to the binding sites. The experiments carried out to compare the inclusion of gap and the exclusion of gap has clearly shown that the inclusion of gap definitely benefited to the task of enhancer prediction.

1.5 Objectives

The intention of this study is to develop deep learning classification frameworks for enhancer regions prediction with the goal of improving the overall effectiveness and efficiency. The two different deep learning classification frameworks presented in this thesis serve the purpose to solve the issues stated earlier. The first framework presented in Chapter 4 is called DeepFinder that ensembled four *de novo* motif discovery tools and employed stacked autoencoder for training the classifier model. The second framework as stated in Chapter 5 is called GMFR-CNN that adapted convolutional neural network to learn the features of binding sites regardless of their locations using gapped motif features. Concerning the main goal and related issues presented above, the two carefully engineered frameworks aim to achieve the following objectives:

- To improve the current three-stage framework by employing an ensemble of motif discovery tools for initial prediction of candidate binding sites from a subset of input sequences. The average accuracy achieved by using the improved method is 97% while the other methods used in the experiments are 3.7% and 53.5% respectively. The traditional three-stage method produced a high number of false positives and often required some clustering technique to obtain high-resolution motifs.
- To implement the use of dependent and independent features as part of the feature-based approach where the benefits of both methods can be achieved in this study. The previous method such as position weight matrix (PWM) fully depends on independent features in motif prediction. It is crucial to include dependent features as well where the binding sites are said to possess these features. The designed features as described in Chapter 4 is capable of attaining the important features that used to define the motifs, for example, dinucleotide repeat motifs.
- To build a deep neural network model using the extracted features associated with the most potential candidate binding sites. This model is a feature-based approach integrated with stacked autoencoder (SAE) that served to find out the specificity of the seed motifs which is capable of discriminating enhancer from non-enhancer regions. The strength of SAE is in feature discovery, and when noise is introduced as part of input features, robust features can still be learned along the way in the learning process. This is where the current enhancer prediction tools lack and are important to consider the benefits of SAE would bring.
- To design a novel representation of DNA sequences as “images” integrated with convolutional neural networks (CNNs) for the purpose of increasing the predictive power of enhancer prediction. This is because the kernel is able to simulate the length of

the motifs with gaps in between them. Also, the transcription factors often bound to the transcription factor binding sites partially. CNN is capable of extracting higher level features from lower level ones without being heavily engineered the necessary features to distinguish enhancers from non-enhancers. The locality of the features can be attained and suitable for enhancer prediction where the location of enhancers is not fixed.

- To analyze and compare the coverage percentage of GMFR-CNN, DeepBind, LS-GKM, CSI-ANN, and iEnhancer-2L in predicting the enhancers in Chromosome 18 of the proboscis monkey genome.

1.6 Technical Contributions

We introduce a novel framework which integrates stacked autoencoder, one of the deep learning techniques together with a three-stage method and we called this framework as DeepFinder. DeepFinder ensembles the advantages of four motif finding tools (BioProspector, MDscan, MEME, and MotifSampler) to search for putative binding sites from the partial input data, later on, the pre-processed putative binding sites are clustered according to their locations based on their overlapped regions. Similarity measure among the binding site clusters is used to populate the distant matrix and divide up into two main clusters using the k -medoid clustering technique. The clustering method was employed because it did not affect by the outliers. Each of the clusters is then trained in the deep learning model as a classifier and the best cluster is used to predict enhancers in the remaining sequences not used in the motif finding tools. We demonstrate the effectiveness of the framework using ten TFBS conserved datasets downloaded from UCSC Genome Browser in comparison with MAST and FIMO. We have selected MAST and FIMO in this case because these two tools are part of the MEME Suite that is frequently

updated by the developers. They are commonly used by the biologists or bioinformatics practitioners to search for sequences that are matched with a set of motifs. Our approach has shown significant results especially in the recall, f -measure, and accuracy values. Also, consistency of the classifier performance is maintained throughout different datasets used.

We develop a novel representation deep learning framework called GMFR-CNN that employs gapped motif features to represent the DNA sequence in two dimensions on top of the deep convolutional neural network (CNN). The proposed representation is capable of capturing the probabilistic weight values of each of the nucleotide positions. We make use of the strengths of CNN in this framework because it is tolerance in terms of translation, rotation, and shifting. Thus, CNN enables recognition of features regardless of their locations in the incoming data. A comprehensive comparison of the proposed framework with LS-GKM and DeepBind is performed using the datasets from DeepBind's literature. The averaged results of five-fold cross-validation have shown that GMFR-CNN has yielded promising results and outperformed competitor tools when employed with gapped motif features representation. Other than that, when GMFR-CNN was performed on proboscis monkey genome specifically on chromosome 18, it has shown higher coverage as compared to other enhancer prediction tools such as CSI-ANN, iEnhancer-2L, DeepBind, and LS-GKM. The genome sequencing, assembly, and annotation of proboscis monkey genome enable the exploration of primate evolution and genetic contributions to human physiology and disease.

1.7 Scope of the Study

This study focused on the enhancement of three-stage framework, the novel representation of the DNA sequence using deep learning models, and the coverage achieved by each tool in the enhancer prediction of the proboscis monkey genome.

The traditional three-stage method is employed for the purpose of using a partial dataset to be scanned by motif finding tools and use the motif profile to scan in the remaining dataset not used in the beginning. This method can avoid an enumerative search of the motifs in the whole genome, however, it contains a high number of false positives. This is where the improvement of this method came in to employ with deep learning and clustering techniques for the purpose of reduction in terms of false positives and also increase the enhancer prediction accuracy. It is utterly important to assemble the motif finding results from multiple motif finding tools because each tool has its own advantages and algorithms. There is no single tool capable of obtaining all the motifs that approximate the real ones. The datasets used for this study include CREB, ELK1, GATA1, HNF4, MEF2, NFE2, P53, P300, SRF, and STAT1. These datasets can be obtained from UCSC Genome database (GRCh37/hg19 Feb. 2009 assembly).

Next, the novel representation of DNA sequence is introduced in this study. In previous studies, there are different representations used to represent DNA motifs which include position weight matrix, k -mer, etc. We realized the inclusion of gaps in the representation enhances the overall prediction which has been shown in the work by Ghandi and the team (2014). Therefore, we introduce the use of kernel together with the implementation of probabilistic weight values where the matrix can be treated as DNA “images” that later on inputting in the convolutional neural network. We named this representation as gapped motif feature representation (GMFR)

that is capable of simulating gaps in between motifs when a kernel size of three convolved in the matrix. The datasets used for this study include CEBPB, CEBPD, CTCF, ELK1, EP300, FOXA2, HNF4A, TBP, TFAP2A, and YY1. These datasets can be obtained from DeepBind's literature (Alipanahi, Delong, Weirauch, & Frey, 2015).

In the prediction of enhancers for proboscis monkey genome, a protocol for genes and enhancer regions annotation was introduced. The intergenic regions obtained from the gene annotation protocol are where the enhancers are predicted later on by different enhancer prediction tools. The tools involved are DeepBind (Alipanahi, Delong, Weirauch, & Frey, 2015), LS-GKM (Lee, 2016), GMFR-CNN (Wong, Lee, & Omar, 2016), CSI-ANN (Firpi, Ucar, & Tan, 2010), and iEnhancer-2L (Liu, Fang, Long, Lan, & Chou, 2016). These tools used transcription factor binding sites (e.g. CTCF and EP300) and histone modification marks (e.g. H3K4me1, H3K4me3, and H3K27ac) to predict the enhancers in the proboscis monkey genome. The annotated protein and transcript sequences as reference data are collected from six different species which include *gorilla gorilla* (western gorilla) (Sally et al., 2012), *macaca mulatta* (rhesus monkey) (Zimin et al., 2014), *mus musculus* (house mouse) (Church et al., 2011), *pan troglodytes* (chimpanzee) (Sequencing & Consortium., 2005), *homo sapiens* (human) (Lander et al., 2001), and *pongo abelii* (Sumatran orangutan) (Locke et al., 2011) (<ftp://ftp.ncbi.nlm.nih.gov/genomes/>).

1.8 Thesis Structure

This thesis consists of 6 chapters including the current chapter which is the introduction. Chapter 2 confers the biological and deep learning background that is related to our study. It is related to gene expression, transcription factor binding sites, motifs, enhancers, existing

computational representations, deep learning approaches, for the purpose of understanding the remainder of this thesis. This chapter also inspects the existing related works in enhancer prediction.

Chapter 3 presents the first contribution of this thesis called DeepFinder. This chapter describes the proposed framework for enhancer prediction with an overview of the entire framework. Each of the processes that took place in the framework is described. The conclusion of this chapter will end with a discussion on experimental results and robustness of the framework.

Next, Chapter 4 confers the second contribution of this thesis by presenting the newly proposed representation of DNA sequence. This proposed representation is crucially important as the current representation is not robust in terms of its short and degenerate properties. The whole structure of this chapter is similar to Chapter 5. The experimental results regarding the performance and robustness of the framework conclude the chapter.

In Chapter 5, an annotation pipeline on enhancer regions prediction of proboscis monkey genome is conferred systematically. This annotation is using MAKER with few related reference proteins and ESTs aligned together to find the locations of the genes and other regions. Thus, intergenic regions obtained are further analyzed before going through the next step which is enhancer prediction using the selected tools. This chapter is concluded by the comparison of coverage percentage among the five enhancer prediction tools which include GMFR-CNN, CSI-ANN, iEnhancer-2L, DeepBind, and LS-GKM.

In the end, Chapter 6 wrap up the thesis with a summary of the main contributions and a discussion of possible further development. This chapter too discusses the future directions

of deep learning in bioinformatics area to inspire readers regarding uncountable possibilities in pursuing their research in this field.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

In this chapter, biological background especially gene expression and transcription factor binding sites will be explained in Section 2.2 which is utterly important to understand the remainder of this thesis. Motifs and their representations in computational approaches also will be illustrated as well in this section. In order for readers to apprehend the remaining chapters, the introduction of deep learning is conferred in Section 2.3 which contained essential knowledge about the deep learning neural network algorithms. Previous works related to our approaches regarding TFBSs identification, notably enhancer prediction, are presented in Section 2.4 where the existing tools that used whole or partial input sequences for motif prediction are included as well. Ensemble approaches that supplemented the motif models with other genomic and biological features of TFBSs are reviewed in Section 2.5 and previous works related to deep learning approaches are presented in Section 2.6. Finally, the chapter is summarized in Section 2.7.

2.2 Gene Expression

The central dogma of biology is often defined by the relationship between DNA, mRNA, and polypeptides in eukaryotic cells. During transcription, mRNA is made from a DNA template in the nucleus. It is then moving into the cytoplasm and combines with ribosomes to direct protein synthesis. This process is called translation. Gene expression took place when the

information in a cistron is being used to make a functional polypeptide chain by transcription and translation. A cistron is the section of DNA that holds information for one polypeptide chain.

Transcription, RNA splicing, translation, and post-translational modification of a protein are regulated by gene expression. It is where information from a gene is used to synthesize gene product that is functional to the cell. Figure 2.1 details the flow of genetic information from DNA to mRNA to protein in a cell. Transcription and translation are further explained in Subsection 2.2.1 and 2.2.2.

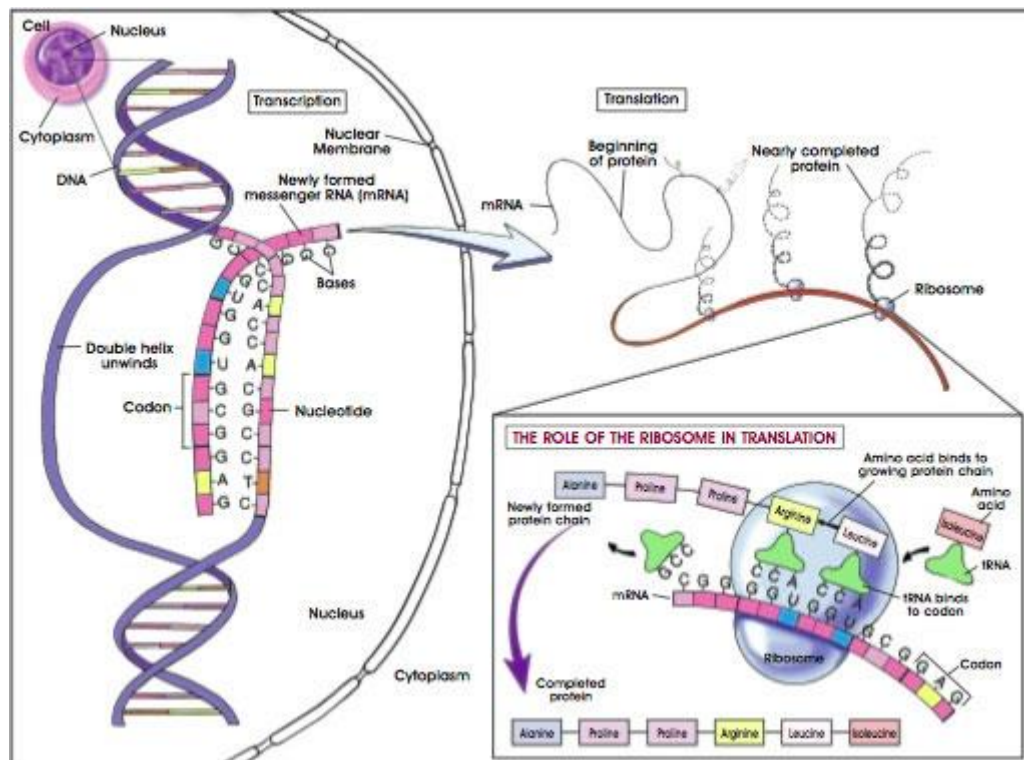


Figure 2.1: Gene expression. (National Institutes of Health)

2.2.1 From DNA to RNA (Transcription)

The variety of activities carried out in a cell requires proteins or enzymes. The information transfer from DNA to mRNA is called transcription. Therefore, transcription is an

important process where it controls how much RNA is created. Helicase is an important enzyme that involved in this matter where it exposes the base in the transcribing DNA strand by breaking up the hydrogen bonds that holding the double helix of DNA. Then, transcription is initiated when polymerase attaches to the promoter site. mRNA is built when complementary nucleotides are added (U to A, C to G, etc.) during the enzyme moves along the cistron in 5' to 3' direction. The double helix of the DNA restores behind it and it is where the 'zip' closes when the enzymes move on to another region of the transcribing DNA strand. The enzyme detaches and the mRNA molecule peels away from the DNA when reaching a 'stop' sequence called a terminator.

Introns are sequences of nucleotides that resides between the cistrons that code for polypeptides. mRNA has to be processed to get rid of the introns before it leaves the nucleus in the eukaryotic cells. After the introns have been removed, the remaining sections of RNA which are called exons are spliced together to form the final strand of mRNA. Figure 2.2 illustrates the regulatory elements involved in the transcription process.

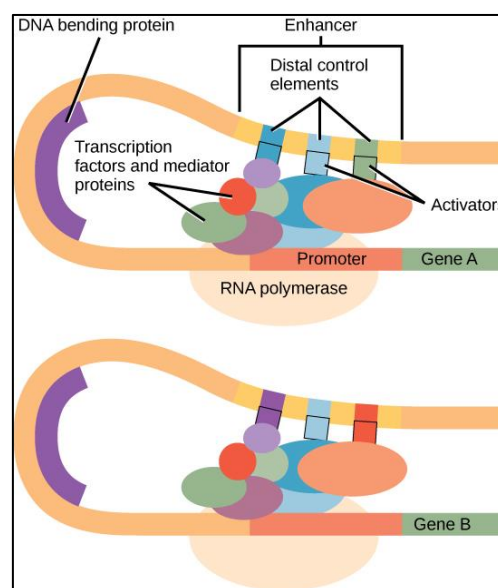


Figure 2.2: Regulatory elements of transcriptional regulation. (adapted from the webpage <https://www.boundless.com/biology/textbooks/boundless-biology-textbook/gene-expression-16/eukaryotic-gene-regulation-113/transcriptional-enhancers-and-repressors-458-11684/>)

2.2.2 From RNA to Protein (Translation)

The process where mRNA is decoded and translated in order to yield a polypeptide is called translation. It is done using mRNA with the help of a ribosome, a number of complex rRNAs, and proteins to carry out the translation. During the process, the mRNA's genetic message is decoded and the new polypeptide chain is assembled. tRNA helps to translate the sequence of codons on the mRNA strands by transferring a free amino acid from the cytoplasm to a ribosome. A single amino acid or also known as a codon is a formation of three bases in a DNA or RNA sequence. They are combined one by one to the growing polypeptide chain. Only two molecules of tRNA are allowed by the ribosome to combine with the mRNA. One of the tRNA molecules is responsible for holding the growing polypeptide chain while the other carries the next amino acid to be added to the chain. A specific enzyme makes sure that it is carrying the correct amino acid specified by its anticodon before a tRNA molecule moves to a ribosome. ATP supplies energy to the attachment of amino acids to tRNA. For example, only tRNA molecule with the anticodon GGA and carrying proline will be attached to the mRNA when the ribosome has arrived at a part of the mRNA strand that contained the codon for proline (CCU). The formation of a peptide bond between the proline and the histidine is catalyzed by peptidyl transferase at the end of the growing polypeptide chain. Therefore, the polypeptide chain is transmitted to proline. The histidine hold by the tRNA molecule is now released from the ribosome, and the tRNA is free to carry another histidine molecule. At the same time, the ribosome moves one codon further along the mRNA strand, and exposes the next codon, for example, (GGU) so that glycine can be added to the chain. This process is terminated when they reach a stop codon on the mRNA. Otherwise, tRNAs will continue to add amino acids to the end of the polypeptide chain. Finally, the completed protein is released to the cell by the ribosome. This translation process is clearly shown in Figure 2.3.

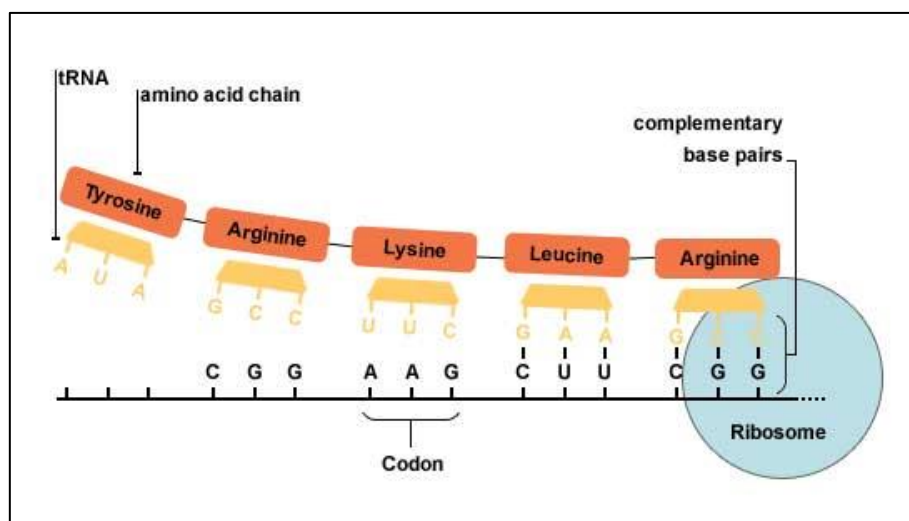


Figure 2.3: Translation. (adapted from the webpage http://www.bbc.co.uk/bitesize/higher/biology/cell_biology/rna/revision/2/)

2.2.3 Transcription Factor (TF)

In molecular biology, a transcription factor has specific recognition of cis-regulatory regions which are called promoters, enhancers, silencers, and insulators that are essential for gene regulation. The binding of some transcription factors on DNA promoter sequence nearby the transcription start site will contribute to the formation of transcription initiation complex which initiates transcription. The RNA polymerase initiates mRNA synthesis by matching complementary bases to the DNA strand. The mRNA molecule is elongated and, once the strand is entirely synthesized, transcription is completed. This well-formed mRNA copies of the gene constitute blueprints for protein synthesis during the translation process. While other transcription factors will either activate or repress the recruitment of RNA polymerase to specific genes especially binding on enhancer or silencer sequences. The location of the regulatory elements such as enhancer or silencer from the gene being transcribed can be up to thousands of base pairs away either in upstream or downstream. Enhancers and silencers only differentiated by the transcription factor that the region binds to because both enhancers and

silencers may be in close proximity to each other or may be located in the same region. A cluster of sites is always required for specific recognition of eukaryotic TFs because eukaryotic TFs display widespread, non-functional binding even though bacterial TFs can recognize a specific DNA site in the genomic background (Wunderlich & Mirny, 2009). The cluster of sites are often motifs that occur repeatedly in the eukaryotic genomes (Todeschini, Georges, & Veitia, 2014).

In this study, our main focus is on the prediction of enhancer, therefore, a description of what is enhancer is conferred. Before this, the study of promoter regions which involved in the cis-regulatory mechanism has gained predominant focus until recently where long-range enhancer clusters were found to play an important role in spatial and temporal gene regulation. The binding of specific transcription factors on enhancer enables the DNA to form a loop in order to allow enhancer-bound regulatory proteins and transcription factors bound to the promoter region. Thus, transcription is activated or repressed through the interactions of the proteins. With the binding of activator and other protein complexes, enhancer has been brought closer to the promoter and hence increase transcription. This shows that enhancer plays a vital role that determines maximal transcription while promoter is crucial whether transcription can occur or not. The maximal increase in transcription is caused by the activators recruiting transcription factors, which helps in the binding of RNA polymerase. Therefore, the rate of transcription can be affected by the binding of regulatory proteins on enhancer which depends on either activators or repressors to the control elements of enhancers.

The location of the enhancer is usually at a distance away, either upstream or downstream (Griffiths, Miller, Suzuki, Lewontin, & Gelbart, 2000). Gene transcription can still be affected by an enhancer, where it may be excised and inserted elsewhere in the chromosome. In addition to that, the forward or reversed pattern of enhancer does not affect the level of

transcription (Narang & Mahato, 2010). Enhancers may show tissue-specific properties and may initiate RNA polymerase II transcription by generating a new group of non-coding RNAs called enhancer RNAs (eRNAs) (Kleftogiannis, Kalnis, & Bajic, 2015). Figure 2.4 illustrates transcription machinery by enhancers bound by transcription factors and mediator proteins.

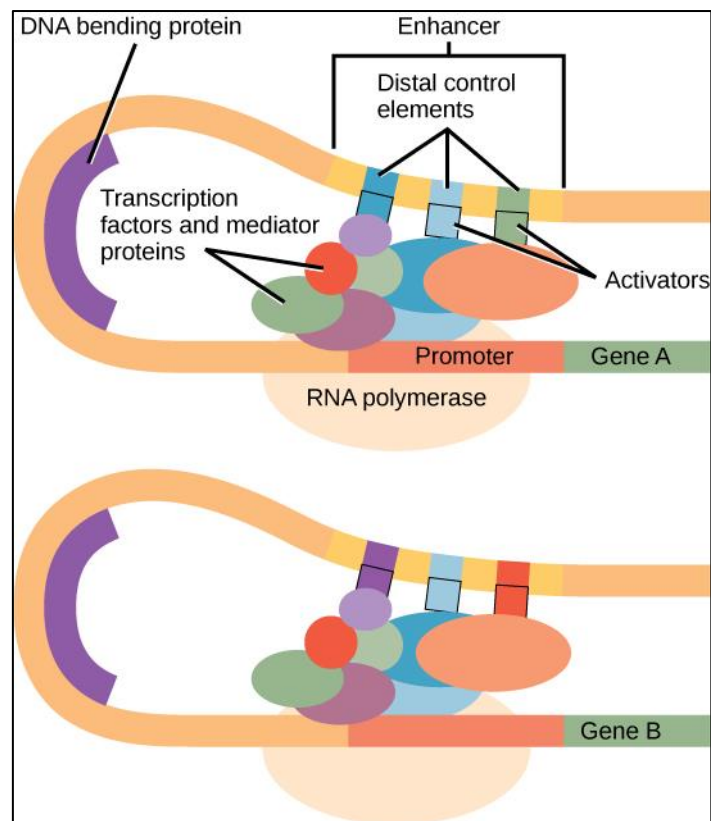


Figure 2.4: Enhancer boosts transcriptional regulation, bound by transcription factors and mediator proteins. (adapted from the webpage <https://www.boundless.com/biology/textbooks/boundless-biology-textbook/gene-expression-16/eukaryotic-gene-regulation-113/transcriptional-enhancers-and-repressors-458-11684/>)

2.2.4 Motif Representations

Transcription factor binding sites or also known as TFBSs are a type of binding site that can be found in DNA sequence where other transcription factors may bind. These binding sites are usually 6 to 20 bases in length, thus, it is short and is often highly degenerate. Multiple TFBSs that occur in a typical enhancer are often conserved sequences with a degree of

degeneracy. The requirements for enhancer activity are commonly related to clustered and/or overlapping TFBSs (Hu, Hu, & Collins, 2007; Gotea et al., 2010; Lusk & Eisen, 2010). Therefore, the binding of these TFBSs is very precise and only significant at a small fraction of sites. The binding specificity is captured by the typical DNA sequence and is often represented by motif. A common or conserved pattern of the motif can be drawn from collections of similar TFBSs that recognized by a TF. According to Sami & Nagatomi (2008), a sequence motif is a nucleotide or amino-acid sequence pattern that is widely spread across the genome and has, or is inferred to have, a biological meaning in genetics.

Currently, there's no existing definition that is fully satisfying and can be employed in finding the definite motifs (Pisanti, Crochemore, Grossi, & Sagot, 2005). Some represent motifs based on statistical representations in the form of PSSM (Position Specific Scoring Matrix) (Gribskov et al., 1987; Hertz & Stormo, 1996; Lawrence & Reilly, 1990; Lawrence et al., 1993). Others represent a motif as a consensus (Brazma et al., 1998; Vanet, Marsan, & Sagot, 1999).

A motif is commonly represented by a position weight matrix (PWM), consensus sequence, and sequence logo (D'haeseleer, 2006). PWM is a profile-based method that describes a motif using quantitative information obtained in a weight matrix (Stormo, 2000); while consensus sequence or sequence pattern confers a motif using a qualitative method such as profiling model that adopts regular expression to characterize a motif (Hertz, Hartzell, & Stormo, 1990); sequence logo is used to represent sequence motifs visually (Schneider & Stephens, 1990). A PWM can be defined by $4 \times l$ matrix $S = (m_{i,j})$ where j^{th} column m_j depicts the probability distribution of the four bases (A, C, T, G) that appears respectively at position j of the aligned common pattern of a binding site of a given TF. Independence between bases was assumed in the input sequences because it scores at each of the position independently from bases at different positions. The sequence score aligned with PWM also

can depicts the log-likelihood of the sequence by using this formula $m_{i,j} = \log \frac{p_{i,j}}{b_i}$, where $p_{i,j}$ is the probability of i at position j of the motif and b_i is the probability of i in a background model. The sequence logo can be graphically plotted using the site-specific $m_{i,j}$ values over sites (Schneider & Stephens, 1990).

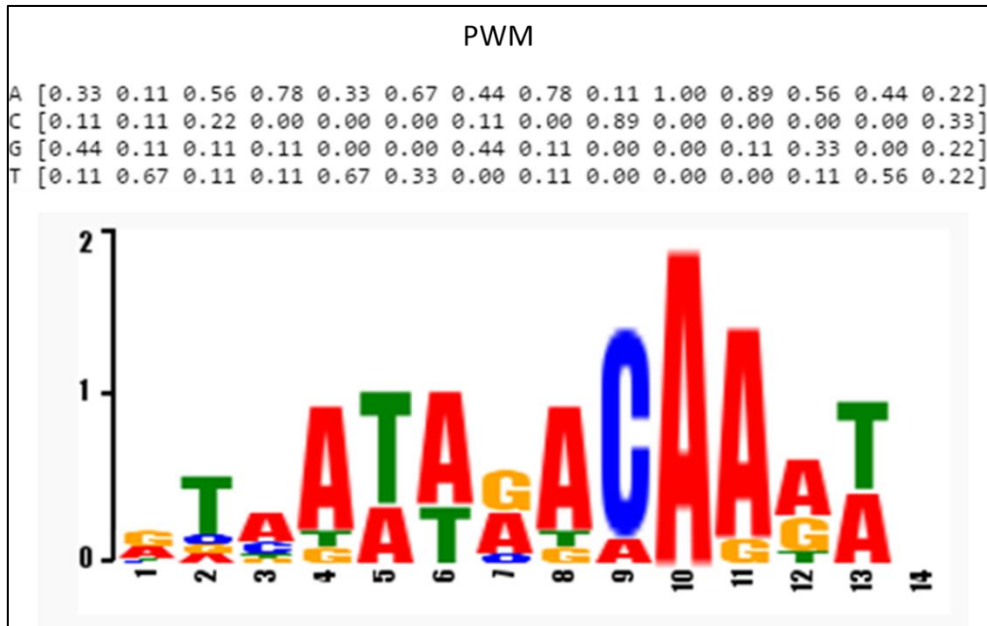


Figure 2.5: PWM and LOGO representation of the nine motifs (Bailey T. L., 2008).

Other than using PWM to represent motif, the consensus sequence is also commonly used to represent aligned binding sites. Conserved sequence motifs are consensus sequences that demonstrate consistency across species or within the species over its evolutionary period and show which residues are conserved or variable. The consensus sequence motif is made up of the 15-letter alphabet, including four nucleotide bases A, C, G, T, and eleven distinct letters of IUPAC nucleotide code for eleven possible subsets of the four nucleotides (Johnson, 2010). The eleven letters are depicted in Table 2.1 as shown below. For example, *Hind*II bind to the sequences GTYRAC, where Y stands for ‘C or T’ (pYrimidine), and R stands for ‘A or G’ (puRine) (D’haeseleer, 2006).

Table 2.1: IUPAC nucleotide code for eleven possible subsets of the four bases.

IUPAC nucleotide code	Base
R	A or G
Y	C or T
W	A or T
S	G or C
M	A or C
K	G or T
B	C or G or T
D	A or G or T
H	A or C or T
V	A or C or G
N	A or C or G or T

Besides, consensus sequence also can be generated from aligned binding sites which uses regular expression to characterize a motif. Regular expressions (REs) are commonly used in the word-based method as it provides a powerful notational algebra to match the DNA sequence patterns (Heger, Lappe, & Holm, 2004). As an example, given a possible RE: “A-G-N(1,2)-C-C”, this RE is matched with any DNA short sequence that starts with AG and ends with CC together with a gap of one or two base pairs in between that can be of any bases. Table 2.2 shows the regular expression obtained from aligned sites.

Table 2.2: Regular expression of aligned sites (Bailey, 2008).

Aligned Sites														
Site1	C	T	A	A	T	T	G	G	C	A	A	A	T	G
Site2	A	T	A	A	T	A	A	A	C	A	A	A	A	C
Site3	G	A	C	A	T	A	G	A	C	A	A	G	A	C
Site4	G	T	C	T	T	T	C	A	C	A	A	A	T	A
Site5	G	T	G	A	A	A	G	A	C	A	A	G	T	T
Site6	A	T	A	A	T	A	A	A	C	A	A	A	A	T
Site7	G	T	T	G	A	A	A	A	C	A	A	T	A	G
Site8	T	C	A	A	A	T	A	T	C	A	A	A	T	C
Site9	A	G	A	A	T	A	G	A	A	A	G	G	T	A
Regular Expression (RE)														
	N	N	N	D	W	W	V	D	M	A	R	D	W	N

For the task of searching new sites, gapped or less conserved sequence motifs, consensus sequence representation may not be sufficient, where the gapped features and relative frequency of each particular base at a given position are totally ignored. The poor description of the motif pattern by consensus sequence allows ambiguity in the consensus that will result in the detection of known motifs and is capable of producing the unacceptably high level of false positives. Other than that, consensus sequence does not take into account insertion and deletion which makes it extremely challenging to detect patterns, for example, bacterial promoter sequences, that consists of conserved region separated by variable distances. In comparison to PWM, the consensus sequence is less sensitive and precise when scanning on new sequences that are not included in the training set. A sequence logo is a visual representation of the conserved regions in DNA or protein strings (Schneider & Stephens, 1990). It is widely used by bioinformaticians to reveal significant features of the alignment in

DNA or protein. Figure 2.5 depicts sequence logo built from WebLogo, a web-based server from aligned binding sites. It basically shows how well nucleotides are conserved at each particular position. The conservation level of the sequence at a particular position is determined by the entire height of the letters. All nucleotides at the same position (A, C, G, T) are scaled according to their respective frequency. The entire height of the sequence logo is scaled by the information content (IC) measured in bits. The term ‘information content’ can be used to describe an estimation of the average specific binding energy to the whole set of known binding sites. The information content or also known as measure of conservation of position i is given by $R_i = \log_2(4) - (H_i + e_n)$. H_i is the uncertainty known as the Shannon entropy of position i : $H_i = -\sum p_{i,j} \times \log_2 p_{i,j}$. $p_{i,j}$ is the relative frequency of base i at position j . e_n is the small-sample correction for an alignment of n letters. The approximation for the small-sample correction e_n is given by: $e_n = \frac{1}{\ln 2} \times \frac{s-1}{2n}$, where $s = 4$ for nucleotides, and n is the number of sequences in the alignment (Schneider & Stephens, 1990).

2.3 A Brief Introduction to Deep Learning

In general, a computational model that is constructed by multiple processing layers which capable of learning the representations of data with multiple levels of abstraction is powered by deep learning methods. These representations are yield by constructing simple but non-linear modules that each transform the representation starting with the raw input into a representation at a higher or more abstract level. Complex functions can be learned with the composition of enough such transformations. For many years, problems that have opposed the best efforts of the artificial intelligence community are taking major advances in deep learning. It is exceptionally good at discovering complex structures in high-dimensional data and is

therefore applicable to many domains of science and technology. Deep learning is one of the most popular techniques used in image and digits recognition, voice recognition, and text document classification ever since it has beaten records in those fields in comparison to machine learning techniques (LeCun, Bengio, & Hinton, 2015). Recently, Google DeepMind's program called AlphaGo which employed deep learning technique has beaten a human expert at Go for the first time in human history (Gibney, 2016). Besides, deep learning techniques also have been implemented in medical and bioinformatics field such as predicting the activity of potential drug molecules (Ma, Sheridan, Liaw, Dahl, & Svetnik, 2015), reconstructing brain circuits (Helmstaedter et al., 2013), and predicting the effects of mutations in non-coding DNA on gene expression and disease (Leung, Xiong, Lee, & Frey, 2014; Xiong et al., 2015). Deep learning has produced extremely promising results for various tasks including the prediction of sequence specificities of DNA- and RNA-binding proteins by Alipanahi and the team (2015), thus, this technique is believed to be benefited to the task of enhancer prediction.

Deep Learning (DL) is a division of machine learning (ML) that is renowned for its learning ability by modelling high-level abstractions features based on lower level ones with the help of a set of learning algorithms (Bengio, 2009). A learning algorithm is a method used to carry out the learning process, the desired design objective is obtained by the function of which is to alter the synaptic weights of the network in an orderly fashion. Deep neural networks can be differentiated from the regular neural networks based on the number of hidden layers where regular neural networks have only one hidden layer and deep neural networks have more than 2 hidden layers. Conventional machine learning approaches are not robust for raw data transformation because they require careful engineering and consideration of domain expertise to design a feature extractor. Hence, this issue is handled by deep learning that the features in

each layer were learned using a general-purpose learning procedure and do not need to be designed by human engineers (LeCun, Bengio, & Hinton, 2015).

The capability of deep learning techniques is a complete breakthrough in discovering complex structures in high-dimensional data as this would allow deep learning practitioners to automatically extract such features without heavily programmed the system and does not require direct human input features (Najafabadi et al., 2015). This has brought major improvement in solving problems that are applicable to science, medicine and business, for example, image recognition (Krizhevsky, Sutskever, & Hinton, 2012), medical image analysis (Xu et al., 2014), and risk analysis (Swaminathan et al., 2015). There are few examples of deep learning such as Deep Belief Networks (DBN) (Boureau & Cun, 2008), Convolutional Neural Network (CNN) (LeCun, Bottou, Bengio, & Haffner, 1998), Deep Boltzmann Machines (DBM) (Salakhutdinov & Hinton, 2009), Recurrent Neural Networks (RNN) (Lukoševičius & Jaeger, 2009), Stacked Autoencoders (SAE) (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010) and etc. In spite of conferring all the available deep learning techniques, we only described the details of Stacked Autoencoders (SAE) and Convolutional Neural Network (CNN) in order for readers to understand the remaining of this thesis since both techniques were implemented in the first and second contribution projects.

2.3.1 Artificial Neural Networks (ANNs)

Artificial neural networks (ANNs) or commonly known as neural networks (NNs) are inspired by how the human brain process incoming signal to perform classification and clustering. They make use of a massive interconnection of simple computing cells known as “neurons” or “processing units”. A neuron is an information-processing unit that forms the basis

of the operation of a neural network by mapping an input vector $\{x_1, x_2, \dots, x_n\}$ to a scalar output y via a weight vector $\{w_1, w_2, \dots, w_n\}$ and a nonlinear activation function A . An artificial neuron is illustrated in Figure 2.6. The graphical representation summarizes the single neuron as $c = \sum_{j=0}^n w_j x_j$ and $y = A(c + b)$, where c is the sum of input signals times with their respective synaptic weights of the neuron ($w_1 x_1 + w_2 x_2 + \dots + w_n x_n$) and y is the output signal of the activation function applied on sum of input signals and bias term. The *bias* term denoted by b can be positive or negative and is able to increase or decrease the sum of inputs of the activation function. The total number of input neurons is denoted by n . The common activation function used here is logistic function and the formulas before and after substitution are defined as below.

$$A(v) = \frac{1}{1 + e^v}, y = \frac{1}{1 + e^{c+b}}.$$

The output of the neuron is determined by the weighted sum c and their respective bias b either less than or greater than the certain threshold value and its output are either 0 or 1.

$$y = \begin{cases} 1 & \text{if } c + b \geq 0 \\ 0 & \text{if } c + b < 0 \end{cases}$$

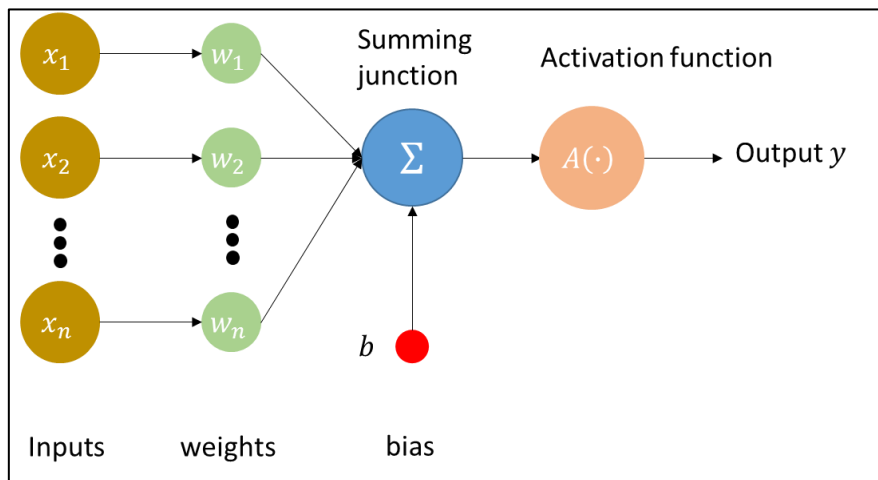


Figure 2.6: A nonlinear model of a neuron.

A neural network is a carefully designed machine that normally consists of three layers (input, hidden, and output layer) which can be used to imitate the brain in performing a specific task such as classification and discover new patterns in data. It is an information-processing system or parallel computer that has the ability to reform structure which is known as neurons to carry out biological computations such as perception, motor coordination, etc. These ‘computations’ are extremely fast compared to any of the supercomputers in existence today.

2.3.2 Multilayer Backpropagation Neural Networks

A multilayer neural network can be distinguished from a simple neural network by the presence of one or more hidden layers, whose neurons are correspondingly called hidden neurons. It is called “hidden” due to the fact that this particular neuron is not seen from either the input or output layer of the neural network. The output of the hidden neurons will act as input to the next hidden layer or output layer. The inclusion of hidden neurons enables the extraction of higher-order features from its input. They begin to discover salient features that characterize and distinguish the training data as the learning process progresses across the neural network. Figure 2.7 illustrates the structure of a multilayer neural network that consists of inputs, one hidden layer, and one output layer. The neural network shown is fully connected as every neuron in each layer is connected to every other neuron in the adjacent forward layer.

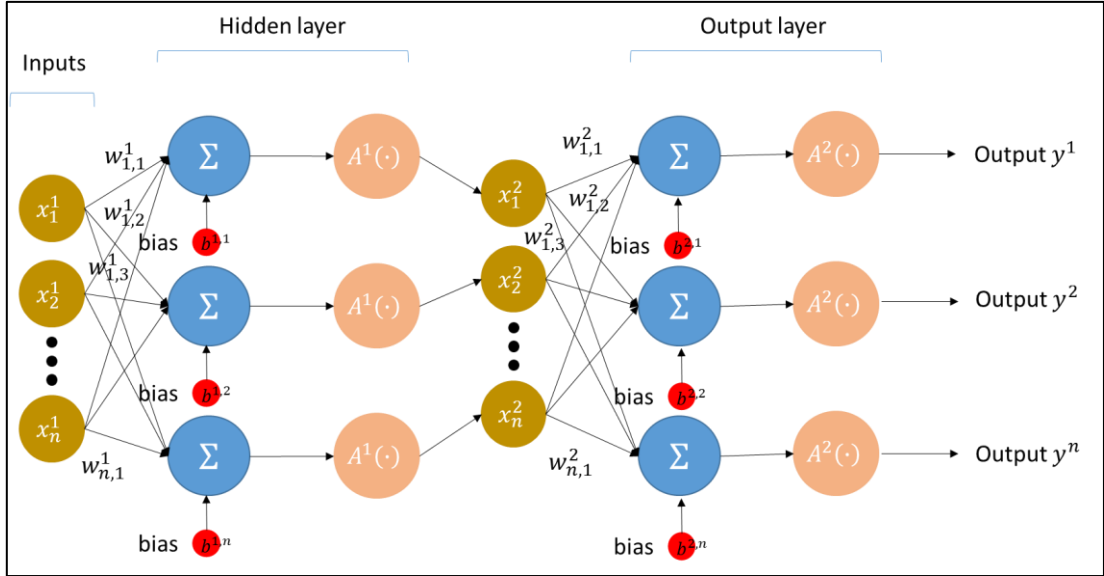


Figure 2.7: The layout of a multilayer neural network for the case of a single hidden layer.

The neural network described previously is a feedforward neural network and this pattern can be observed where the output of one neuron is used as input to the next neuron in the hidden layer or output layer. Bryson and Ho invented a technique called backpropagation or generalized delta rule for learning in multilayer network in 1969 (Bryson & Ho, 1975). The purpose of this technique is to increase the efficiency of partial derivatives computation of a function $A(\bullet)$ recognized by the network with respect to all the neurons of the adjustable weight vector w for an assigned value of input x . The implementation of gradient descent rule in a feedforward network makes backpropagation works by starting off with an initial set of parameter values and constantly progresses toward a set of parameter values that minimize the function. This network that implemented gradient descent is called backpropagation neural networks (BPNNs). BPNN is a multilayer, feedforward network that has at least one hidden layer and its hidden neurons are consists of nonlinear functions (e.g. sigmoid function and hyperbolic tangent function). The network is fully connected between neurons in two consecutive layers, but no connection between neurons within the same layer. Each hidden neuron receives input from all input neurons and sends output to all output neurons for a three-

layer neural network (input, hidden, and output layer). The number of output neurons is not necessarily equal to the number of input neurons and the number of hidden neurons per layer can be more or less than the number of input or output neurons. When the number of hidden neurons increases, the network will tend to converge, however, excessive hidden neurons will lead to poor generalization (Basheer & Hajmeer, 2000). There are many suggestions proposed by researchers with regards to the number of neurons that may not be relevant to every circumstance. For the case of a single hidden layer architecture, it has been suggested to use an upper limit of $2n + 1$ hidden layer neurons, where n is the number of inputs (Hecht-Nielsen, 1987). While for the case of two hidden layer architecture, a 3:1 ratio of neurons between the two layers is suggested (Maier & Dandy, 1996). Other than that, in order to avoid overfitting, it was suggested that the number of hidden neurons should be less than 1/10 the number of training patterns (Weigend, Rumelhart, & Huberman, 1990).

The two main steps involved in the backpropagation algorithm are feedforward and backwards pass operations. Figure 2.8 illustrates the signal flow of the multilayer neural network with two hidden layers in forward and reverse direction.

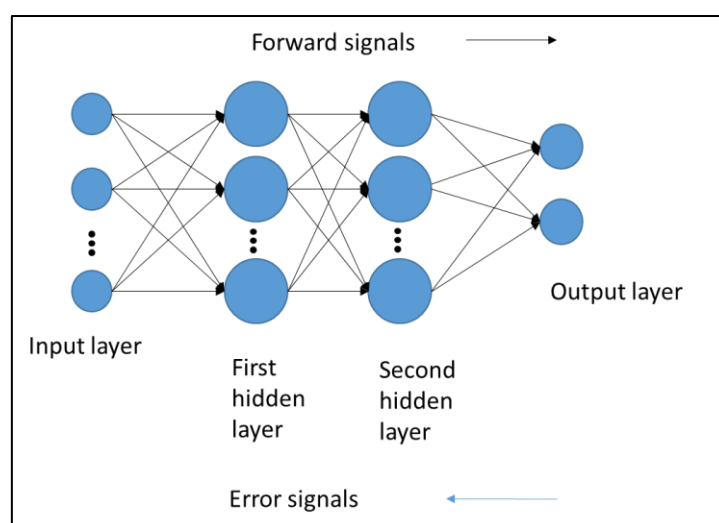


Figure 2.8: An architecture of multilayer neural network with two hidden layers where the signal flow is in the forward direction.

Given a pair of data (p_i, q_i) where p_i is an input signal that travels around in the network and q_i is the actual output or target value, to the network in the feed-forward propagation. Consider a neuron in a hidden layer, r_h and connected with a weight w_{ih} to the neuron in the input layer. By using this equation, $r(out)_h = A(\sum_{i=1}^n w_{ij}p(out)_i + b_{0h})$ is outputted a value to all neurons in the next layer.

While in the backwards pass process, a mean-squared error function is used to compare the final value of p_i and target value, q_i . Thus, a mean-squared error value is computed using $E = \sum_j (U_j(p_i, w) - q_i)^2$ where $U_j(p_i, w)$ is the final output of feed forward pass of j th neuron from input layer until output layer and w is the weights from neurons in the previous layer to j th neuron. A gradient descent rule is implemented to update the connection weights by propagating the neural network in backward direction.

$$\Delta w_{ij} = \frac{-\mu \partial E}{\partial w_{ij}},$$

where Δw_{ij} entails the updated weight from any neuron to neuron and both of which are of different layers. The learning rate parameter in the formula is depicted as μ . So, the weight change from the input layer i^{th} neuron to hidden layer h^{th} neuron is described as follow:

$$\Delta w_{ih} = \mu \cdot \delta_h \cdot x_i \text{ where } \delta_h = r(out)_h(1 - r(out)_h) \sum_k w_{hk} \cdot \delta_k.$$

Next, the weight change from the hidden layer h^{th} neuron to the output layer k^{th} neuron is shown in the equation below. As denoted previously, y_k is the final output of feed forward pass which is the same as $U_j(p_i, w)$ and the target value q_i is denoted as $y_{target,k}$. A backward pass pseudocode is conferred right after the equation below:

$$\Delta w_{hk} = \mu \cdot \delta_k \cdot r(out)_h \text{ where } \delta_k = (y_{target,k} - y_k)y_k(1 - y_k).$$

2.3.3 Deep Learning Neural Network (DLNN)

DLNN has been actively employed in many of the state-of-the-art methods to perform tasks like speech recognition, language translation, object detection (LeCun, Bengio, & Hinton, 2015). These techniques heavily employ the available computation to construct highly complex neural networks even though large datasets were used, and only a small amount of the data is labelled. For more details regarding the neural network models, readers can refer to Schmidhuber's paper (Schmidhuber, 2015), as a complementary of all the available models.

Deep learning refers to machine learning techniques for the learning of hierarchical representations in the network's deep architectures that are used for pattern classification. While for the neural network, is a programming paradigm that imitates the human brain which enables a computer to learn from observational data. The depth or number of hidden layers of the neural networks can be used to distinguish between shallow networks and deep networks. Conventional machine learning such as hidden Markov models (HMMs) and support vector machines (SVMs) depend on shallow nets which comprised of input and output layers. When the number of hidden layers exceeds one, it is considered as "deep" network (Goldberg, 2016). Shallow net architecture is relatively simple because it contains only single layer nonlinear functions that are important in reconstructing raw input or features into desirable feature space.

In general, a set of features is being trained in each layer in the deep neural networks, the deeper the layer, the more complex features can be recognized. This is due to the composition of higher order features from lower order features in the previous layers which are commonly known as feature hierarchy. The main concept of deep learning lies within the complexity and abstraction level of hierarchical representations which increase from one layer to the next layer (Bengio, Courville, & Vincent, 2013). Thus, deep neural networks' capability

in capturing meaningful features is automatically achieved by its ability to deal with high-dimensional datasets together with various parameters. This is a key advantage of deep learning where good feature extractors are not hand-engineered by domain expertise which requires a considerable amount of engineering skills and experience (LeCun, Bengio, & Hinton, 2015). Moreover, deep learning neural network is also an architecture with high adaptation to new problems (Najafabadi et al., 2015). Despite all the strengths possessed by deep learning, here are some limitations presented by Yann LeCun in CVPR conference 2015 which include the missing of theory in deep learning, lack of reasoning or structured prediction, memory (short-term, working, episodic memory), and unsupervised learning that actually functions. In the following subsections, explanations of stacked autoencoders and convolutional neural networks are provided. The subsections are necessary to understand the remainder of this thesis.

2.3.3.1 Stacked Autoencoders

An autoencoder works by mapping the input vectors into a hidden representation that is done by an encoder while the mapping of hidden representation back into the space of input vectors is done by using a decoder. Therefore, the input vectors can be interpreted as target vectors in the training of autoencoders (Amaral et al., 2013). The objective of the autoencoder is to minimize the reconstruction error, which is depicted by a distance between the input vectors and the target vectors. Mean squared error is the most common type of distance used in the algorithm. Autoencoder becomes very useful for feature learning when it has a non-linear form or uses a multi-layered model. This is because a potentially more powerful representation of the data can be learned than its linear form (Deng & Yu, 2014).

A traditional autoencoder used by Bengio and the team in 2007 to construct deep neural networks is depicted in Figure 2.9 and the algorithm is presented as follows (Bengio, Lamblin, Popovici, & Larochelle, 2007). An input vector $x \in [0,1]^d$ is taken by an autoencoder to map it to a hidden representation $y \in [0,1]^{d'}$ by an encoder through a deterministic mapping $y = f_\theta = s(Wx + b)$ and parameterised by $\theta = \{W, b\}$ where W is a $d' \times d$ weight matrix and b is a bias vector. The reconstruction of vector $z \in [0,1]^d$ was mapped back by the resulting latent representation y with a decoder in input space $z = g_{\theta'}(y) = s(W'y + b')$ where $\theta' = \{W', b'\}$. The parameters of this model which include W , W' , b , and b' are optimized for the purpose of average reconstruction error minimisation:

$$\begin{aligned}\theta^*, \theta'^* &= \arg \min_{\theta, \theta'} \frac{1}{n} \sum_{i=1}^n L(x^{(i)}, g_{\theta'}(f_\theta(x^{(i)}))) \\ &= \arg \min_{\theta, \theta'} \frac{1}{n} \sum_{i=1}^n L(x^{(i)}, g_{\theta'}(f_\theta(x^{(i)})))\end{aligned}$$

where L is a loss function such as the traditional squared error $L(x, z) = \|x - z\|^2$. A cross-entropy of the reconstruction can be used when the input is interpreted as either bit vectors or vectors of bit probabilities which also known as Bernoullis:

$$\begin{aligned}L_H(x, z) &= H(B_x || B_z) \\ &= - \sum_{k=1}^d [x_k \log z_k + (1 - x_k) \log(1 - z_k)]\end{aligned}$$

Given the Bernoulli parameters z , $L_H(x, z)$ is a negative log-likelihood for the example x if x is a binary vector. The network is trying to learn the identity function $f_\theta = x$ by appointing the output values to its input values, i.e. $y^{(i)} = x^{(i)}$. It is done using backpropagation algorithm in the most fundamental way. The neurons in hidden layer become feature detectors when the

weights are learned. Learning the identity function can be insignificant when the number of hidden neurons is equal to or larger than the number of inputs, therefore, the number of hidden neurons is a parameter that should be set carefully (Qi, Wang, Zheng, & Wu, 2014). Therefore, for the case of classification, visualization, communication or storage of high-dimensional data, lesser hidden neurons in compared to input neurons is favourable. This is because dimensionality reduction can be achieved (Hinton & Salakhutdinov, 2006). Overfitting may have occurred when the number of hidden neurons is more than inputs and the local minima problem may have deepened as well (Sun, 2012).

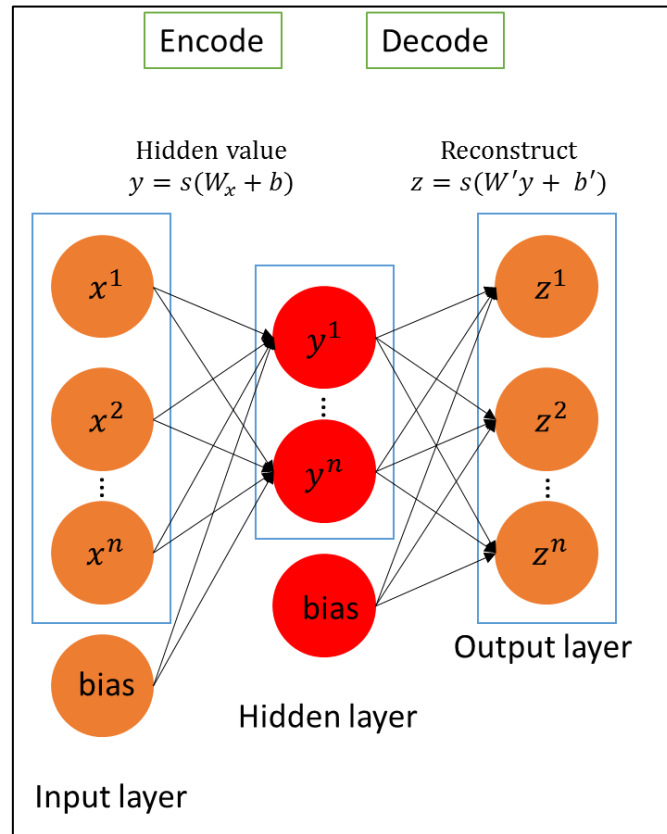


Figure 2.9: A simple autoencoder with three layers (input, hidden, and output layer).

The true meaning of the term ‘stacked’ in stacked autoencoder or SAE is the highlight of the network compared to the traditional autoencoder networks discussed earlier. A stacked autoencoder is arranged in such a way that the output of the first layer is treated as input for the

second layer in the neural network. It normally consists of several layers of sparse autoencoders or denoising autoencoders (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010). Ng (2011) stated that a stacked autoencoder is very powerful in its expression where the network is capable of learning features to build a “part-whole decomposition” of the input. This also means that the autoencoder is capable of producing a more abstract and higher level representation of the observed patterns from the previous layer where lower level features are constructed into more meaningful features. Using the example of an object (e.g. cup) recognition, the first order features obtained from the learning process can be the edges of the cup and the second order features obtained from the composition of first-order features will be more complex such as the cups’ handle. Thus, when higher-level features are learned, the model will be able to recognize the features that formed the structure of a cup. There are two commonly known autoencoders: (1) denoising autoencoder (Vincent, Larochelle, Bengio, & Manzagol, 2008) and (2) sparse autoencoder (Le, 2013). A denoising autoencoder is an extension of the traditional autoencoder that randomly corrupts or introduces noise to the input by forcing it to 0 in the computation in order to make the learned representations robust to the partial corruption of the input pattern. A sparse autoencoder involves a sparseness constraint on hidden units that is enforced by the addition of the Kullback-Leibler (KL) divergence term to the objective function. In this study, denoising autoencoder was chosen over sparse autoencoder because denoising autoencoder is able to automatically learn the statistical difference between inputs and noise which helps in classification task (Lu, Tsao, Matsuda, & Hori, 2013). The idea behind denoising autoencoder is to force the hidden layer to discover more robust features and prevent it from simply learning the identity, thus, the autoencoder is trained to reconstruct the input from a corrupted version of its own. In other words, denoising autoencoder is a stochastic version of the traditional autoencoder.

A denoising autoencoder minimizes the error in reconstructing the input from a stochastically corrupted transformation of the input as depicted in Figure 2.10. After that, primary features are learned on the raw input and the raw input that feed into this trained denoising autoencoder will obtain the primary feature activations for each of the inputs. After the corruption process, these primary features are used as the corrupted “raw input” to another denoising autoencoder to learn secondary features on these primary features as illustrated in Figure 2.11. The primary features are feed into the second denoising autoencoder to obtain secondary feature activations for each of the primary features. As depicted in Figure 2.12, these secondary features will be treated as corrupted “raw input” to a logistic classifier after applying the corruption process, training it to map secondary features to labels. Then, in Figure 2.13, all three layers are combined together to form a stacked autoencoder with two hidden layers and a final classifier layer that are capable of carrying classification task as desired. It is important to note that the corruption process q_D is only used during training and not for propagating representations from the raw input to higher-level representations.

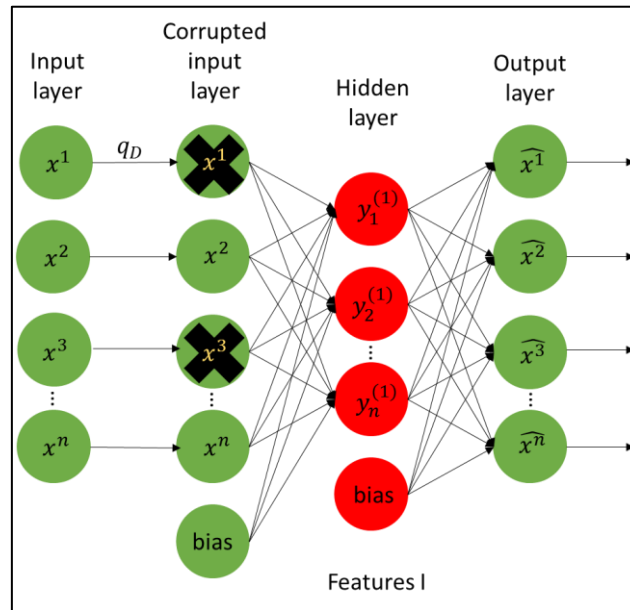


Figure 2.10: First denoising autoencoder is trained to obtain primary features.

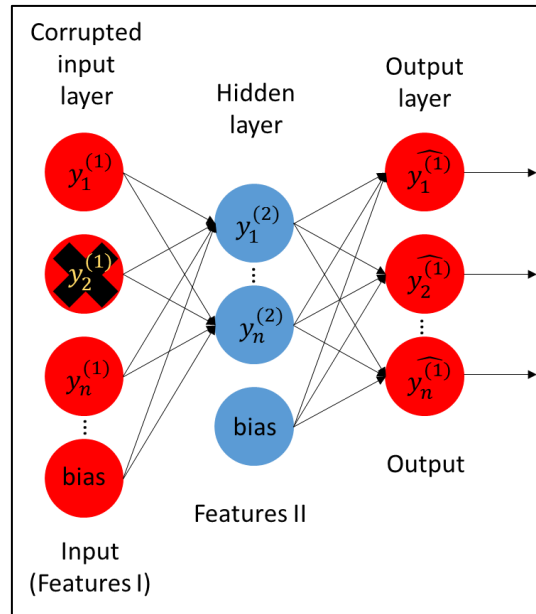


Figure 2.11: Second denoising autoencoder.

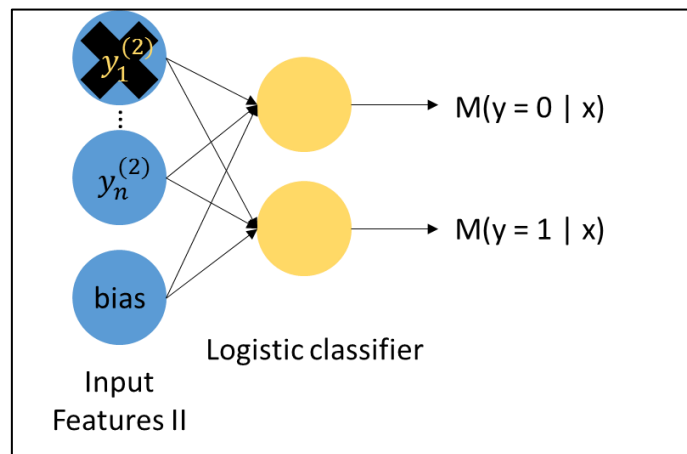


Figure 2.12: Secondary features as raw input to a logistic classifier.

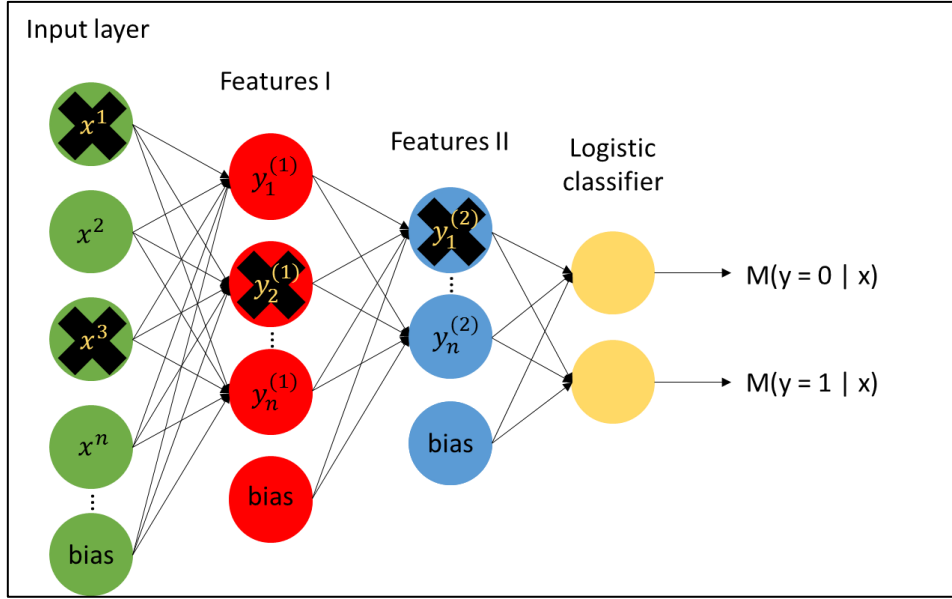


Figure 2.13: Combined all three layers together to become stacked autoencoder.

All the benefits of any deep network are contained in a stacked autoencoder including the reduction of the need for feature engineering and the adaptation to new problems becomes relatively easy (Najafabadi et al., 2015).

2.3.3.2 Convolutional Neural Network (CNN)

A convolution neural network consists of one or more convolution layers and often with a subsampling layer. In standard multilayer neural network, it is often followed by one or more fully connected layers. In the convolution layer for an image recognition as an example, the two-dimensional image acts as input to the network but each of the neurons in the image which is the pixel values are not fully connected to the next layer due to memory intensive computation. For example, a fully-connected network with the input image of size 32×32 and a hidden layer having 1000 features will require an order of 10^6 coefficients, thus, a huge memory is required for the computation. Therefore, CNNs exploit local correlation by

enforcing a local connectivity pattern between neurons of adjacent layers where a local receptive field from the previous layer will compute the synaptic inputs to each neuron, thereby forcing it to extract local features. The exact location becomes less important when a feature has been extracted as long as its position relative to other features is approximately preserved. Figure 2.14 depicts (a) a sparsely connected neurons where the inputs of hidden neurons in layer m are a subsets of neurons in layer $m - 1$ which have spatially contiguous receptive fields, (b) a version of its own with fully connected neurons, and (c) a feature map that consists of three hidden neurons. In sparse connectivity, neurons in layer m have receptive fields of width 3 instead of 5 in the inputs. The learnt “filters” that are able to produce the strongest response to a spatially local input pattern is guaranteed by this architecture. This procedure that produces local features is called feature extraction.

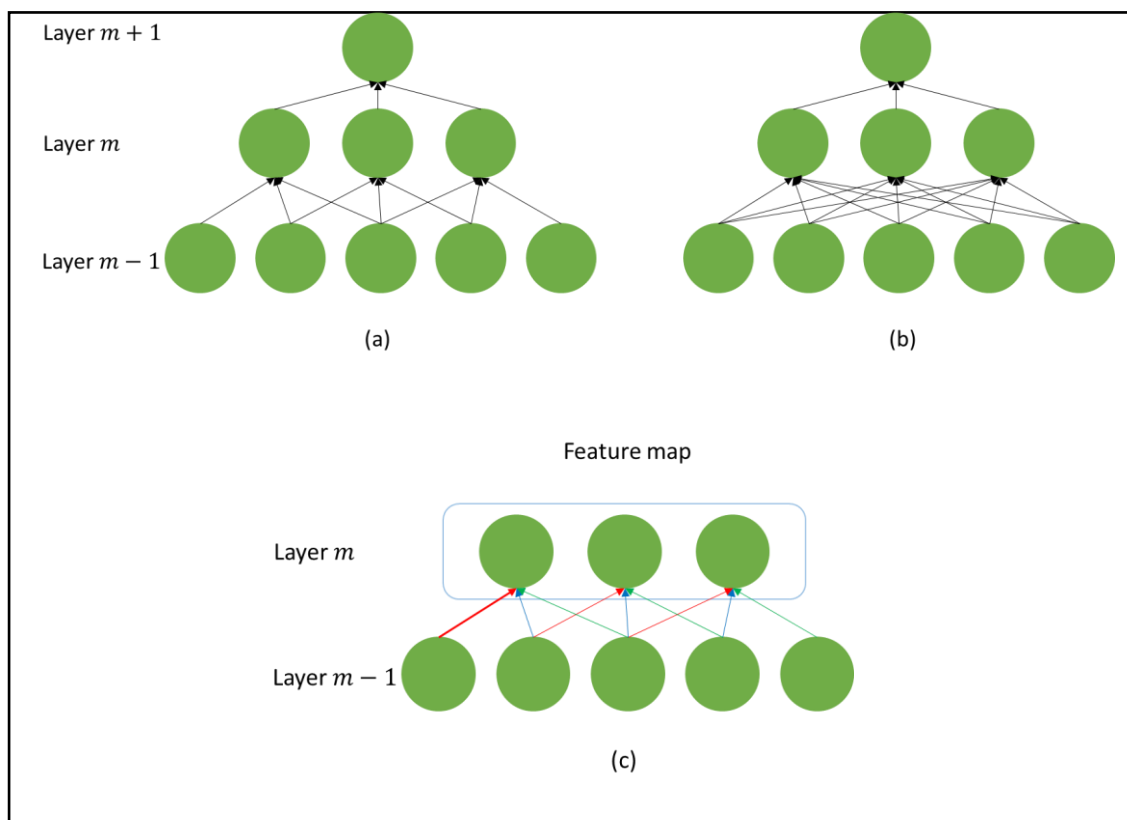


Figure: 2.14: (a) A sparse connectivity between neurons in adjacent layers. (b) A full connectivity between neurons in adjacent layers. (c) A feature map that comprised of three hidden neurons.

Next, the output of one filter which applied to the previous layer is called feature map and each computational layer of the network is comprised of multiple feature maps. Each feature map is in the form of a plane within where the individual neurons are forced to share the same set of synaptic weights. It is obtained by convolution of the input image with a linear filter, adding a bias term and then applying a non-linear function. A simple illustration of convolution is depicted in Figure 2.15.

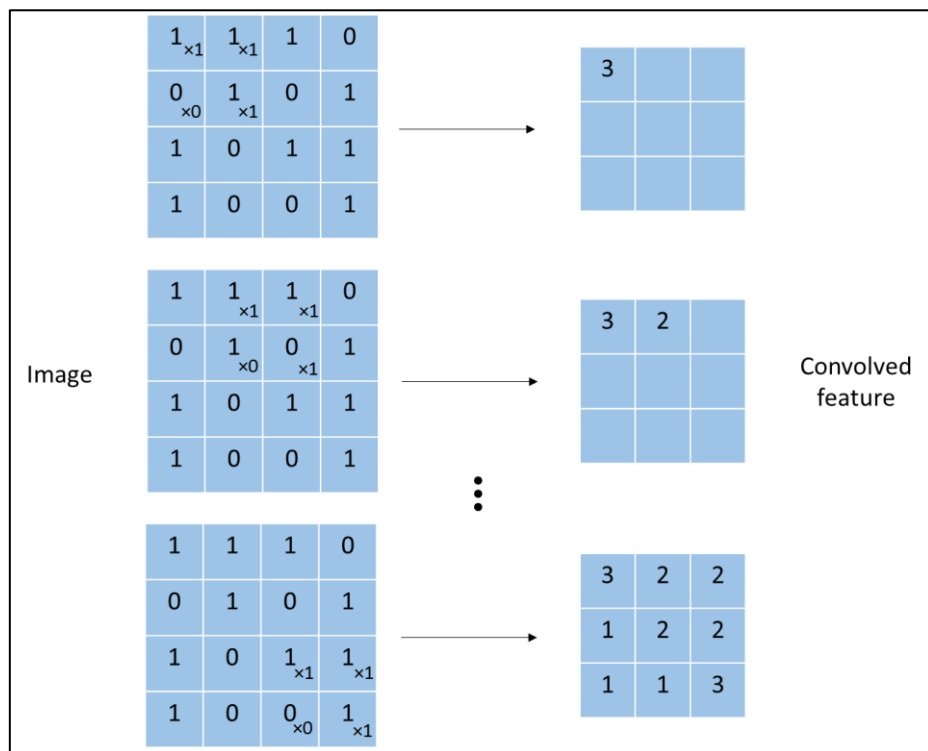


Figure 2.15: A simple illustration of convolution.

After that, a max pooling layer or subsampling layer is needed to pool the convolved features and take the max or mean feature activation over these regions of the image to obtain the pooled convolved features which will be used for classification later on. This is where the resolution of the feature map is diminished and hence reduce the dimension of the image. This process is capable of reducing the feature map's output sensitivity to translation, rotation, and scaling invariance. This is due to parameterization that is capable of learning all weights of a

network equally well. The advantages attained from such constraint including a reduction in the number of free parameters. The reduction of free parameters can be achieved through the use of weight sharing (Haykin, 2009).

Throughout this process, the exact location information will be lost and fortunately, the position relative to other features is mostly retained. Figure 2.16 illustrates the convolution network which consists of 6 layers including input and output layer.

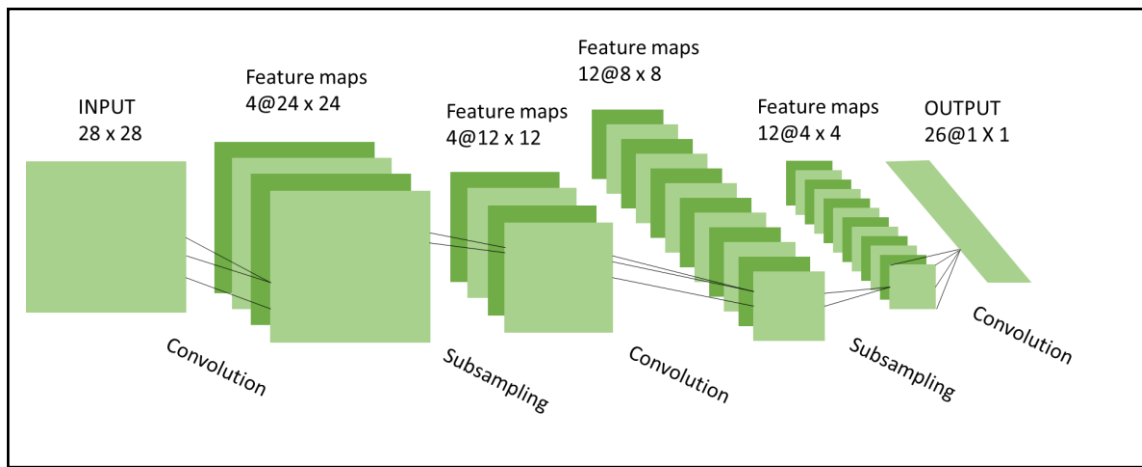


Figure 2.16: A convolutional neural network for image processing such as digit recognition.

Given that the input 2D representation for CNN is a $n \times m$ 2D signal $X = \{x_{ij}\}$ and each convolution layer l has $K^{l-1} * K^l$ weight matrices $W^{l,k^l,k^{l-1}} = \{w_{ij}\}^{l,k^l,k^{l-1}}$ of size $u^l \times v^l$ which is usually known as kernels or filters and K^l intercept biases b^{l,k^l} , where K^{l-1} is the number of output feature maps at layer $l - 1$ while K^l is the number of feature maps at layer l . Then, the activation or output maps A^{l,k^l} of each convolutional layer l , for each kernel k^l of the layer, are generated by convolving the input signals $A^{l-1,k^{l-1}}$ from the previous layer with each kernel, the bias of each kernel is added and a non-linear activation function $g(x)$ is applied (El-Sayed, Estaitia, & Khafagy, 2013):

$$a_{ij}^{l,k^l} = g(h_{ij}^{l,k^l}), h_{ij}^{l,k^l} = \sum_{k^{l-1}} (W^{l,k^l,k^{l-1}} \otimes A^{l-1,k^{l-1}})_{ij} + b^{l,k^l}$$

where $A^{0,1} = X$, and the convolution operator $\otimes$ is as follows:

$$(f_{n_f \times m_f} \otimes g_{n_g \times m_g})_{ij} = \sum_{u=i-n_g/2}^{i+n_g/2} \sum_{v=j-m_g/2}^{j+m_g/2} f_{u,v} * g_{n_g-u, m_g-v}.$$

The dimension reduction of the output features maps is expected for each of the convolution layer using a non-linear down-sampling method. There are two commonly used down-sampling techniques which are *Max Pooling* and *Mean Pooling*. Max pooling functions by partitioning the feature map into small grids and only outputting maximum value of each grid square while mean pooling only outputting the average value of each grid square (LeCun, Bottou, Bengio, & Haffner, 1998). The main disadvantage of average pooling is that all elements in a pooling region regardless of low or high activations may downplay a higher activation value. This is because if the region consists of most negative activations, it will resulting in a near zero activation function even though the most important information was the high activation of one of the feature detectors (Zhang, Zhu, & Jia, 2015). Therefore, max pooling is preferred. The final output of each convolutional layer can be described as $o_{ij}^{l,k^l} = \text{maxpool}(A^{l,k^l})_{ij}$. After this, a fully connected feed-forward networks construct the final layer of CNNs. CNNs are similar to other feed-forward neural networks, where it is trained using backpropagation of error and gradient descent function. The error signals δ^{l,k^l} or matrices are computed as follows for each convolutional layer l and output feature map k^l .

$$\delta^{l,k^l} = \text{upsample}(\sum_{k^{l+1}} W^{l,k^{l+1},k^l} \times \delta^{l+1,k^{l+1}}) * g'(H^{l,k^l})$$

where $H^{l,k^l} = \{h_{ij}\}^{l,k^l}$ is the convolved input feature map that treated as input to compute the activation A^{l,k^l} for layer l and output feature map k^l . Before the convolution layer is pooled, inverse operation is applied to the pooling function used in the convolutional layer, i.e. it upsamples the δ^{l,k^l} matrix to the dimensionality of the output feature map before. Each error signal is written to the position of each unit by the upsampling operation which was chosen as max, while zero is written to the positions of the non-chosen units. It is worth noticing that the forward error signal $\delta^{l+1,k^{l+1}}$ from the fully connected output layer is received by the last convolutional layer. Last but not least, the stochastic gradient related to each kernel $W^{l,k^l,k^{l-1}}$ and each bias b^{l,k^l} is given as:

$$\nabla_{W^{l,k^l,k^{l-1}}} L = A^{l-1,k^{l-1}} \otimes \text{rot90}(\delta^{l,k^l})$$

$$\frac{\partial L}{\partial b^{l,k^l}} = \sum_{i,j} (\delta^{l,k^l})_{ij}$$

where $\text{rot90}()$ rotates 90 degrees of the input matrix and the convolution operator, $\otimes$ (Gülçehre & Bengio, 2016). It is how convolution neural networks are trained like fully connected feed-forward network layers using these gradients after its implementation in a gradient descent optimization procedure.

2.4 Using Whole vs Partial Input Set

The current novel experimental techniques namely, protein-binding microarray (PBM) (Bulyk, 2006), chromatin immunoprecipitation microarray hybridization (ChIP-chip), and high-throughput sequencing (ChIP-seq) (Bailey et al., 2013) enable genome-wide identification of TFBSs (Zambelli, Pesole, & Pavesi, 2012). These techniques are used for the

characterization of the sequence specificities of DNA-protein interactions. However, only a relatively small number of the eukaryotic factors with PBM data is available (Munteanu, Ohler, & Gordân, 2014). Therefore, PBM data is not suitable to be employed for the purpose of eukaryotic motif prediction. The data provided by these techniques are used as inputs to motif discovery tools in the case of locating TFBSs in the regions accurately. ChIP-seq has more advantages in compared to ChIP-chip. The main improvements are in base pair resolution, avoiding non-linearity, saturation of ChIP-chip signal intensity, and limitations from the limited selection of probes on a ChIP-chip array. Besides, it also has the ability to analyze sequence repeat regions (Jayaram, Usvyat, & Martin, 2016). The ChIP-seq dataset contains less noise which makes it a good source for researchers to obtain the dataset for their experiments such as genome-wide characterization of transcription factor binding, polymerase binding, and histone modifications (Zambelli et al., 2012). Other than that, the dataset has a high probability of containing TFBSs. Such sequence is said to have a high resolution where the sequence length is short which is about 200 bp.

In brief, we can group the existing computational prediction tools into two groups which are tools that use the whole set for motif discovery and tools that only use the partial set for motif discovery. The first group are tools that use the entire data as input data for motif prediction such as MEME-Chip (Machanick & Bailey, 2011), RSAT Peak-Motifs (Thomas-Chollier et al., 2012), GimmeMotifs (van Heeringen & Veenstra, 2011), and ChIP-Munk (Kulakovskiy, Boeva, Favorov, & Makeev, 2010). They explore the entire sequence space to search for significant motifs. Heuristics are thus employed due to the complexity of large sequence space for the purpose of search time reduction (Zambelli, Pesole, & Pavesi, 2012; Jia, Carson, & Yu, 2013; Yu et al., 2015). In the second approach, only a subset of data is used for initial data motif prediction. The discovered motifs then are used to perform site scanning in

the remaining sequences. The examples of tools that use a subset of data for motif prediction are anchor based sequence clustering algorithm (ASC) (Liu, Han, Zhou, Yan, & Kosik, 2016) and entropy-based position projection algorithm (EPP) (Zhang, Wang, & Yan, 2016).

When the motif discovery carried out on the binding regions which resulting from the genome-wide techniques of ChIP-seq, using conventional motif discovery programs such as MEME produces ChIP-seq data that contains thousands or more sequences (Jayaram, Usvyat, & Martin, 2016). Tools that use entire data such as MEME-ChIP has a limitation in terms of the time cost of the motif discovery step which depending on the MEME program, thus, the square power of the sequence size increases (Thomas-Chollier et al., 2012). Therefore, the implementation of using partial inputs for motif discovery can avoid this problem. The use of existing motif-finding tools on a partial set of all sequences has become a common practice for performing motif finding on ChIP-seq data such as using the top 500 sequences or top 10% of all sequences (Jothi, Cuddapah, Barski, Cui, & Zhao, 2008). For example, ASC proposed to divide a sequence dataset into multiple smaller clusters where sequences of the same motif will be allocated to the same cluster. After that, the motifs are generated by each individual cluster when applied with an existing motif algorithm. In the end, the results from multiple clusters are merged together as final output. The performance of ASC has shown that the running time of MEME has been greatly reduced from two weeks to less than 10 minutes and discover even more motifs than MEME, for example, ASC has found 24 motifs while MEME has found 20 motifs on a real dataset with 11, 642 protein sequences. However, the proposed algorithm only tackles short sequences and assumed that each sequence contains at most one motif. Therefore, when comes to different datasets with more than one motif in the input sequences, the performance of this method might be deteriorated since they only group sequences with similar motif into the same cluster and may not be applicable for a sequence with two or more motifs.

Another example of motif prediction tool which uses partial input is called EPP. EPP is a projection method that designed to divide the dataset into subsets and refines the subsets by searching all the possible instances, thus, explores the best local optimal solution. In comparisons to other tools such as MEME, GAME (Wei & Jensen, 2006), VINE (Huang, Lee, & Hsieh, 2011), and APMotif (Sun, Huo, Yu, Guo, & Sun, 2015), EPP has shown performance stability with average 0.85 in terms of Precision, Recall, and *F*-score for different datasets (e.g. CREB, CRP, MEF2, MYOD, SRF, and TBP). Nevertheless, motif instances are hard to be differentiated from the background sequences when the width of the motif is short, hence, the accuracy of the performance is affected as well. In addition, when the length of motifs ranges from 7 to 56 bp were used, EPP only able to predict a partial segment with the true motif because of the fixed motif length used. Other than that, EPP did not assemble other algorithms' advantages in enhancing the prediction performance just like EMD (Hu, Yang, & Kihara, 2006) did.

2.4.1 Word-based Approaches

Word-based methods or also known as string-based methods mainly depend on exhaustive enumeration such as counting and comparing oligonucleotide frequencies (Das & Dai, 2007). Global optimality was assured by this method and they are relevant for short motifs, thus, useful for motif finding in eukaryotic genomes. This is because motifs are shorter than prokaryotes in most cases. Other than that, these methods are a good option for total-constrained motif discovery especially when all instances are identical and are high in efficiency when implemented with optimized data structures such as suffix trees (Sagot, M., 1998). Unfortunately, the result of word-based methods often needs to be post-processed with some

clustering system when transcription factor motifs that often have several weakly constrained positions are being used. Another limitation is too many spurious motifs are being produced by word-based methods. The example of motif discovery tools that use word-based methods are gkm-SVM (Ghandi, Lee, Mohammad-Noori, & Beer, 2014), MITRA (Eskin & Pevzner, 2002), Yeast Motif Finder (YMF) (Sinha & Tompa, 2003), Weeder (Pavesi, Mereghetti, Mauri, & Pesole, 2004), and Oligo/Dyad-analysis (Defrance, Sand, & Van Helden, 2008).

The gkm-SVM was extended from the kmer-SVM method (Lee, Karchin, & Beer, 2011) to use gapped k -mers or robust k -mer count estimates as feature sets and introduce tree data structure to compute these new kernels. This method is significantly outperformed a kmer-SVM using both CTCF and EP300 genomic bound regions over a variety of feature lengths. Other than that, gkm-SVM also outperformed the single PWM by identifying necessary co-factors on the ENCODE ChIP-seq datasets. However, gkm-SVM can only support up to approximate 20,000 training sequences. When more training sequences are involved, LS-GKM (Lee, 2016) is introduced instead of gkm-SVM. The training of an SVM in gkm-SVM involves evaluation of a kernel matrix which defined as an n -by- n matrix of all possible inner products or kernel functions between a set of vectors of n training examples. Unfortunately, computation of the kernel matrix becomes impractical when n increases with gapped k -mers as features. Therefore, gkm-SVM implements an efficient tree data structure that computes a full kernel matrix with a runtime that linearly scales with n instead of n^2 for the purpose of solving this issue. The pre-calculated kernel matrix was used to train the SVM classifier. On larger datasets, optimal training of gkm-SVM was hindered by the full kernel matrix evaluation which is required in the original implementation because substantial memory resources that are proportional to n^2 are needed. In order to improve this problem, the implementation of gapped k -mer kernel functions within the LIBSVM framework which is now called LS-GKM is introduced (Chang

& Lin, 2011). LIBSVM is commonly used in most SVM tools where it can utilize decomposition methods for SVM training. Only a partial kernel matrix is needed for LIBSVM to find and solve a small subset SVM problem iteratively. For example, for every problem solving steps, only two columns of the kernel matrix were evaluated by LIBSVM in its sequential minimal optimization algorithm (Fan, Chen, & Lin, 2005). Thus, the memory resource issue can be effectively reduced by replacing the LIBSVM kernel routines with the gkm-kernel functions and larger datasets can be trained on the SVM. There are two limitations of LS-GKM where it does not allow gaps which occur at the end of a k -mer and it does not allow imperfect mismatches in which all nucleotides are treated equally (Ghandi, Lee, Mohammad-Noori, & Beer, 2014).

Mismatch Tree Algorithm or also known as MITRA (Eskin & Pevzner, 2002) is based on a mismatch tree in which the paths from the root to the leaves represent both the substrings and all valid mismatches of these substrings in the data. A rooted tree where each node can be either an internal node or a leaf node is known as a mismatch tree. There are four branches labelled with each element of $\{A, C, G, T\}$ for each of the internal node. Given a sequence S , and parameters l , d , and k , all of the (l, d) -patterns that occur at least k times in the sample which denoted as $(l, d) - k$ are retrieved. The maximum depth of the tree is represented as l . Prefixes are matching the path of the tree from the root to node by each node in the mismatch tree which coincides to the subspace of patterns. For example, the node that is reached by the path ATC corresponds to the space of all patterns that have a prefix ATC . $ATCTGC$ would be a possible pattern in this space while $AAATGC$ would not. The space of all patterns is corresponded by the root node of the mismatch tree. The fundamental approach used in MITRA is start with examining the root node. When examining a node, if the possibility of the subspace corresponding to the node containing a relevant pattern cannot be excluded, then, the node

children will be expanded and examined. A valid pattern can be obtained when the pattern corresponding to the path from the root to the leaf has reached a depth l . Unfortunately, when applied to spaced motifs, this algorithm may not be effective. MITRA finds the monad segments first then based on the locations of monad segments, locate a set of possible dyads. Dyad is a spaced motif with two segments. In the experiment of comparison between SPACE, MITRA and BioProspector on spaced motifs in real biological datasets, MITRA scored an average of 0.64 while SPACE and BioProspector scored an average of 0.77 and 0.69 respectively (Wijaya, Rajaraman, Yiu, & Sung, 2007). This has shown that when comes to spaced motifs, the performance of MITRA is deteriorated.

Another tool that implemented a word-based method is called Weeder. In Weeder, the exhaustive search for the exponential number of candidate consensus was employed with the initial indexing of the sequences with a suffix tree to perform exhaustive matching with no limits on the position of substitutions (Välimäki, Gerlach, Dixit, & Mäkinen, 2007). The algorithm of Weeder can be summarized as follows. Given as input a set of sequences and an error ratio $\epsilon < 1$, discovers all patterns that take place in at least q sequences of the set with at most ϵm mutations, where m is the length of the pattern. The only limitation is established on the location of mutations along the signal where a valid occurrence of a pattern can occur at most $\lceil \epsilon i \rceil$ mismatches in the first i nucleotides. In the performance assessment carried out by Tompa et al. (2005), the results of the comparison experiments showed that Weeder outperformed the other tools (AlignACE (Roth, Hughes, Estep, & Church, 1998), ANN-Spec (Workman & Stormo, 2000), Consensus (Hertz & Stormo, 1999), GLAM (Frith, Hansen, Spouge, & Weng, 2004), Improbizer (Ao, Gaudet, Kent, Muttumu, & Mango, 2004), MEME (Bailey & Elkan, 1995), MITRA, MotifSampler (Thijs et al., 2002), Oligo/Dyad-Analysis, QuickScore (Régnier & Denise, 2004), SeSiMCMC (Favorov, Gelfand, Gerasimova, Mironov,

& Makeev, 2004), YMF (Sinha & Tompa, 2003)) in most domains and by most measures. Some part of Weeder's success is the result of careful decisions made regarding when to predict no motif in a dataset. Only the strongest motifs were reported because Weeder was set to run in a "cautious-mode" (Das & Dai, 2007).

Oligo-analysis was developed by van Helden and the team in 1998 based on the word-based approach. They analyzed that the efficiency of the algorithm is proven for extracting motifs from most of the yeast (*Saccharomyces cerevisiae*) regulatory families where these motifs had been previously discovered by laboratory experimental analysis. In addition, putative regulatory sites were predicted within upstream regions of coregulated genes. Oligo-analysis is exhaustive and rigorous compared to heuristic methods. Other than that, only relatively simple patterns that include short motifs with highly conserved core can be detected. The procedure used in developing the algorithm includes the constitution of regulatory families and the calculation of expected oligonucleotide frequencies. Next, van Helden and the team enhanced their method to find spaced dyad motifs. The spacer length systematically ranges between 0 and 16 because the spacer can be varied for different motifs. There are two ways where the significance of spaced dyad motif can be measured either based on the combined score of the two conserved parts in the input data or based on the estimated complete dyad frequency from a background dataset. However, this algorithm does not allow variations within an oligonucleotide.

Next, Sinha and Tompa (2003) developed the algorithm YMF or also known as Yeast Motif Finder. The motif model was derived by the team from a study of known transcription factor binding sites in yeast. A set of upstream sequences act as inputs to the algorithm, the number of non-spacer characters in the motifs to be enumerated, and the transition matrix for an order m Markov chain assembled from the full complement of upstream sequences of yeast.

They carried out a validation experiment where YMF was used to recognize candidate binding sites in 23 well studied regulons of yeast. Regulon is a set of genes controlled by a common regulator. For 18 of these regulons, the known binding site consensus for the regulon's principal transcription factor has been successfully reported by YMF. YMF is also used for motif discovery in gene families especially the functional and mutant phenotype catalogues of *S. cerevisiae* MIPS database (Mewes et al., 2002) and discovered many putative transcription factor binding sites. The experimental setup that used to evaluate the performance of YMF, MEME (Bailey & Elkan, 1995), and AlignAce (Roth, Hughes, Estep, & Church, 1998) is using synthetic and yeast promoter sequences. From the experiments, YMF has yielded a high accuracy of known regulatory elements on more of the yeast regulons than the other two tools. Moreover, the team has suggested that trying a few different motif discovery tools other than YMF is beneficial because different tools performed better with different datasets.

A motif finding algorithm that looks for occurrences of regular expression-type patterns was developed by Brázma and the team (1998) using a word-based approach. The algorithm was applied for the discovery of patterns in the entire set of over 6000 sequences taken from upstream of the putative yeast genes and patterns in the upstream regions of co-regulated genes in yeast. Most had matches to known motifs in yeast among the highest rating patterns. This method is exhaustive in the sense that they heavily rely on counting and comparing patterns that occur more frequently by searching all possible regular expressions that are most overrepresented (van Helden, André, & Collado-Vides, 1998). The overrepresentation is rated based on the occurrences of patterns at least in some minimum number of sequences. The pattern rating is based on comparing the number of matching positive sequences and the number of matching negative sequences. The positive sequences are upstream regions or upstream regions from genes with a specific expression profile and negative sequences are random

genomic regions or regions upstream from the genes with different expression profiles. Binding site profiles that are constructed based on the information content or expectation maximization are generally better representations of binding sites than regular expression type patterns because they can give more accurate descriptions. In the meantime, it is more challenging to find profiles from datasets that include sequences that contain no specific binding site, because profiles can easily include this noise in the model.

In general, the word-based approach is rigorous and exhaustive in contrast with heuristic methods. The detection power is only limited to simple patterns which include highly conserved short motifs (Das & Dai, 2007). The word-based approach can be improved in terms of computational efficiency of the algorithm such as implementing a suffix tree into the representation of sequences (Sagot, 1998). Unfortunately, this will lead to more constraints on the searching for target motifs.

2.4.2 Probabilistic Approaches

Over the last twenty years, the probabilistic concept has constructed many useful and efficient algorithms. The probabilistic approach makes certain decisions randomly. It is proven that gene modules that are identified based on motif discovery using a probabilistic approach, are suitable for analyzing synthetic lethal genetic interaction datasets, thus, is promising in the integrative analysis of heterogeneous datasets (Sinha, Van Nimwegen, & Siggia, 2003). A position weight matrix used to represent the motif model which was involved in the probabilistic approach (Bucher, 1990). A pictogram where each position is represented by a stack of letters whose height is proportional to the information content of that position is known as a position weight matrix (Schneider & Stephens, 1990). The discovery of longer or more

general motifs is done by many of the designed algorithms which were developed from the probabilistic approach. Thus, these algorithms are less required for the discovery of transcription factor binding sites (Das & Dai, 2007). This is why they are more suitable to be used in prokaryotes for motif discovery, where the motifs' length are normally longer than eukaryotes. Besides, they employ some form of local search, such as Gibbs sampling, expectation maximization (EM) or greedy algorithms, therefore, globally optimum solutions are not guaranteed by these algorithms and may converge to a locally optimal solution. The two frequently used probabilistic methods called MEME and Gibbs sampling is described below.

Multiple expectation maximization for motif elicitation (MEME) is a software tool for motif finding in a set of DNA or protein sequences. The expectation maximization (EM) algorithm was employed to find motifs in a number of sequences by representing motifs as position weight matrices (PWMs). Lawrence and Reilly (1990) introduced EM algorithm and their intention is to apply this algorithm for the discovery of protein motifs. However, this algorithm can also be applied for DNA motif discovery. In order to identify motifs in unaligned biopolymer sequences, the MEME algorithm extended the EM algorithm. The purpose of MEME is to find new motifs in a set of biopolymer sequences where the presence of any motifs is not known. There are three new ideas included by MEME for motif discovery. First, for the EM algorithm, the probability of finding globally optimum motifs is increased using sub-sequences as a starting point where they usually appear in the biopolymer sequences. Second, each sequence does not necessarily contain exactly one occurrence of the shared motif. Third, a procedure for shared motifs to be probabilistically deleted is incorporated after they are discovered so that different motifs can be discovered in the same set of sequences, both when different motifs occur in different sequences and when a single sequence may contain different motifs. A set of correlated genes from genomic, transcriptomic, and proteomic analyses was

acquired by Hall and the team (2005) where they applied MEME to scan 1,000 bp of the 3' end of the stop codon, six of the analyzed sequences has discovered a 47 bp motif. After that, 20 other genes were identified to have the same motif using the motif found to search the entire genome. Puf protein was bound by this motif which means the transcription of the analyzed genes may control by Puf protein (Hall et al., 2005).

Another example of probabilistic approach is the Gibbs sampling algorithm where it is one of the simplest Markov chain Monte Carlo algorithms. The joint distribution of the parameters will converge to the joint probability of the parameters by Gibbs sampling algorithm in the given dataset. This method is said to be fast and sensitive. An optimized local alignment model for N -linear time can be found, thus, the problem faced by the EM algorithm can be avoided. For example, in order for weakly conserved patterns to reach statistical significance, a relatively large dataset such as 15 or more sequences was required (Lawrence et al., 1993). By using Gibbs Sampler, Petersen and the team (2000) tried to find motifs that are not fully conserved in 17 putative promoter regions acquired from microarray experiments (Petersen et al., 2000). Gibbs Sampler found motifs "TTGACT" and "GACTWWHC" repeatedly which had been identified by previous experiments and the search was performed in sequences between 6 to 16 bp.

In recent years, feature-based motif representation has gained attention, especially on enhancers to model protein-DNA specificities. Machine learning is acquired to learn the discriminative features which can be associated directly or indirectly with binding site and must purportedly engineer from labelled input DNA sequences. Direct features are those obtained directly from DNA sequences like sequence features or motif scores whereas indirect features are served as additional information such as sequence conservation among orthologous species (Berman et al., 2004), chromatin accessibility (Wang, Zhang, & Zhang, 2013), epigenetic data

(Fernández & Miranda-Saavedra, 2012), or GC content that can be used to discriminate binding or non-binding regions. The tool called DISCOVER proposed by Fu, Ray, and Xing (2009) used 9 types of features to predict CRM regions. *K*-mer feature (Pinello, Bosco, & Yuan, 2013) is used to represent epigenetic marks and signal processing technique has also been employed (van Duijvenboden, de Boer, Capon, Ruijter, & Christoffels, 2016). Other than that, Yáñez-Cuna et al. (2014) introduced the feature known to be associated with active enhancers called repeated dinucleotide feature.

2.5 Ensemble Approaches

Ensemble technique employs several motif finding tools to search for high potential candidate motifs in the same input dataset and aggregates results to conclude a final motif prediction. Based on the performance comparison of 13 *de novo* motif finders, in Tompa's benchmark and *Escherichia coli* datasets, Tompa et al. (2005) described that the performance of stand-alone motif finding algorithm was not consistent because there is no motif finding tool performs consistently well for all eukaryotic datasets and assembling multiple motif discovery tools enhanced the average sensitivity by 62%, however, the average precision is reduced by 15%. In addition to that, another team by Hu, Li, and Kihara (2005) who had produced a comprehensive set of performance measures and benchmarked five modern sequence-based motif finding algorithms using large datasets generated from *Escherichia coli* RegulonDb, reported that an improvement of 6-45% in sensitivity and specificity could possibly be obtained from the ensemble algorithms. Tran & Huang (2014) also suggested users to use different motif finding tools to get significant motifs, non-overlapping motifs, and overlapping similar motifs. Overlapping motifs are more reliable because they are reported by different tools that

implement different algorithms. The ensemble approach was introduced by Hu et al. (2005) to improve the accuracy of motif prediction for motif finding algorithms. This is because of earlier motif finding algorithms such as BioProspector that suffered from low prediction accuracy and it is proven that the performance of an ensemble approach was never worse than single motif finder (Hu, Yang, & Kihara, 2006). There are several ensemble algorithms which have been applied in prediction methods such as gene prediction (Aggarwal, Worthey, McDonagh, & Myler, 2003), protein tertiary structure prediction (Fischer, 2003), protein domain prediction (Saini & Fischer, 2005), and protein secondary structure prediction (Albrecht, Tosatto, Lengauer, & Valle, 2003). Thus, Hu's proposed EMD algorithm to serve the purpose by ensembling multiple predictions from multiple runs of one or more *de novo* motif discovery tools (Hu, Yang, & Kihara, 2006). It was the first time to apply ensemble approach to the motif discovery problem. There were five motif discovery tools used in EMD: (1) AlignACE, (2) BioProspector, (3) MDscan, (4) MEME, and (5) MotifSampler. Buhler and Tompa (2002) have developed Projection and it was integrated to improve the scalability of EMD algorithm. The performance of all the possible combinations of one to five component algorithms are assessed. There are two different types of datasets used in the algorithm. A benchmark dataset that is generated from E. coli RegulonDB and another dataset is input sequences of different lengths generated by adding margins of different sizes to each known site. It has shown 22.4% improvement in nucleotide prediction accuracy compared to the best stand-alone motif discovery algorithm. They also discovered that EMD always outperformed or shown equal performance level as its stand-alone tool either for short or long sequences. Shorter sequences which have been tested on EMD algorithm have shown improvement in terms of accuracy in contrast to longer sequences because the performance of the EMD algorithm over the single algorithms deteriorates for long input sequences.

Other than EMD algorithm, there are few other tools which employed ensemble approach to perform motif discovery. For example, TAMO combines results from three motif discovery algorithms which are MEME, AlignACE, and MDscan (Gordon, Nekludova, McCallum, & Fraenkel, 2005); GimmeMotifs employs 9 motif finding tools and the candidate motifs produced by the tools are merged using a new similarity metric (van Heeringen & Veenstra, 2011); MotifVoter employs 10 motif finders that are based on PWM and (l, d) model where l is a continuous string of length l and d is the number of mismatches (Wijaya, Yiu, Son, Kanagasabai, & Sung, 2008); SCOPE assembles 3 programs' prediction results (Chakravarty, Carlson, Khetani, & Gross, 2007). A brief description of the four tools is described below.

The interpretation of transcriptional regulation using DNA-sequence motifs by an object-oriented computational framework is called TAMO or also known as Tools for Analysis of MOtifs. TAMO introduces a motif object with interfaces to several popular tools in order to simplify the application of multiple motif discovery tools to genome-wide data. Other than that, TAMO also includes modules for the integration of motif analysis with diverse data sources which includes genomic sequences, microarrays and various databases. Finally, programs for sequence analysis, algorithms for scoring, comparing, and clustering motifs, as well as several other statistical tests also included by TAMO. Tens of thousands of motifs derived from hundreds of microarray experiments were analyzed by these tools (Gordon, Nekludova, McCallum, & Fraenkel, 2005).

Next, another example of ensemble approach is GimmeMotifs. It is a pipeline that combines *de novo* motif discovery tools to predict motifs from ChIP-seq data. Binding peaks resulting from ChIP-seq experiments were used by this method to predict motifs *de novo* and validate these motifs in an independent fraction of the dataset. The weighted information content (WIC) similarity score is used to compare similar redundant motifs and these similar

motifs will be clustered using an iterative procedure. This method is proven to perform well on human and mouse ChIP-seq datasets as shown by the benchmarks which consist of 18 transcription factor ChIP-seq datasets (van Heeringen & Veenstra, 2011).

MotifVoter is an example of ensemble method which uses the results from single motif discovery tools to predict the motifs and binding sites. MotifVoter comprised of two stages includes motif filtering and site extraction. In the first stage, the candidate motifs predicted by different motif finders were processed by MotifVoter which attempts to discard the spurious motifs. A cluster of motifs with high conformity based on a motif similarity measure can be obtained. In the second stage, MotifVoter identifies a set of potential binding sites based on the candidate motifs retained in the first stage. The use of motif similarity in selecting those highly similar putative binding sites is not new. The similarity measure is crucial in identifying how similar is one cluster compared to the other. Also, the similarity measure not only can tell how similar is the cluster of binding sites over to the other, it can shift location to measure the similarity between two binding sites and only those highest score position are selected for overall similarity computation. MotifVoter is capable of locating more than 95% of the binding sites found by its component motif discovery tools based on the evaluations done on 186 datasets. Besides, MotifVoter also outperforms stand-alone motif discovery tools and ensemble methods especially on Tompa's benchmark (Tompa et al., 2005), *Escherichia coli* (Salgado et al., 2004), and ChIP-Chip datasets in terms of sensitivity and precision (Wijaya, Yiu, Son, Kanagasabai, & Sung, 2008).

The last example of ensemble approach described in this section is SCOPE. SCOPE is a novel parameter-free approach for the identification of putative regulatory motifs in sets of coordinately regulated genes. The SCOPE algorithm incorporates the output of three component algorithms which includes BEAM (Carlson, Chakravarty, & Gross, 2006), PRISM

(Carlson, Chakravarty, Khetani, & Gross, 2006), and SPACER (Chakravarty A., Carlson, Khetani, DeZiel, & Gross, 2007), with each designed to identify a particular class of motifs. SCOPE identifies the most potential candidate motifs from its component algorithms using a ‘winner takes all’ learning rule. Through the tests on 78 published regulons from four different species, SCOPE has successfully identified motifs with a higher level of accuracy than a number of other web-based motif finders which run with their default parameters (Carlson, Chakravarty, DeZiel, & Gross, 2007).

2.6 Deep Learning Approach

The multiple layers of a deep neural network can attain multiple levels of information processing and abstraction within the cells, thus, deep learning is suitable to be used in genomics. The insight about the biological structures can be inferred at multiple levels of resolution and this is revealed by the explicit representations at each layer. Meaningful features or representations that can be used for protein-nucleic acid binding prediction, are either individual sequence motifs or *k*-mers at the lowest layers combinations of motifs at intermediate layers and complex motif ‘grammars’ at the highest layers. Traditional motif scanning algorithms would miss higher-level features that may underlie ultraconserved elements or enhancer clusters (Park & Kellis, 2015). This is why deep learning has come into consideration where more abstract features can be obtained throughout the learning process.

Deep learning is considered as one of machine learning techniques that specifically focus on the deep network to extract higher level features from lower level ones. There are a plenty of bioinformatics approaches rely on machine learning techniques, specifically employed deep learning techniques in achieving their purposes. To the best of our knowledge

at the time of this thesis, the recent works introduced are DeepBind (Alipanahi, Delong, Weirauch, & Frey, 2015), Basset (Kelley, Snoek, & Rinn, 2016), deep feature selection (DFS) model (Li, Chen, & Wasserman, 2016), PEDLA (Liu, Li, Ren, Bo, & Shu, 2016), EP-DNN (Kim et al., 2016), DanQ (Quang & Xie, 2016), DeepCpG (Angermueller, Lee, Reik, & Stegle, 2017), BiRen (Yang et al., 2017), etc.

Employing deep learning approach in bioinformatics is firstly introduced by Alipanahi, Delong, Weirauch, and Frey (2015) called DeepBind. DeepBind has outperformed other state-of-the-art methods when training on *in vitro* data and testing on *in vivo* data. This software tool is built based on deep convolutional neural networks (CNNs) which can discover new features regardless of its position in the sequence. Besides, predicting the activity of potential drug molecules, analyzing particle accelerator data, reconstructing brain circuits, and predicting the effects of mutations in non-coding DNA on gene expression and disease also can be achieved excellently by multiple processing layers within deep learning computational models (LeCun, Bengio, & Hinton, 2015).

The implementation of deep learning approach is followed by Basset which led by Kelley, Snoek, and Rinn (2016) from Harvard University. Basset was trained on a compendium of accessible genomic sites mapped in 164 cell types by DNase-seq. A single sequencing assay in their cell type of interest can be performed using Basset and cell's chromatin accessibility code can be annotated for every mutation in the genome. In this case, Basset is capable of annotating and interpreting the noncoding genome. Their works had been incredibly showed that deep learning neural network has begun to shed light on TFBSs prediction starting in the mid of 2015.

Another example of deep learning approach is a deep feature selection (DFS) model that was proposed to differentiate active promoters and enhancers from the rest of the genome by selecting input features in a deep structure, especially for multi-class data (Li, Chen, & Wasserman, 2016). The dependency of the features was considered by this model and cell-ubiquitous characteristics were taken into accounts, such as CpG-islands and evolutionary reservation Phastcons score, and cell-specific events. The cell-specific events include DNA-accessibility, histone modifications, and transcription factor binding sites were captured by the ENCODE consortium using ChIP-seq techniques (ENCODE Project Consortium, 2012). The DFS model was built by a sparse one-to-one linear layer between the input layer and the first hidden layer, elastic-net, nonlinear module such as sigmoid or tangent, and software classifier.

Next, a deep learning framework named PEDLA was developed to predict enhancers using heterogeneous types of data on a genome-wide scale. By inputting the normalized features and annotated labels, the training of the neural network is started. It is then followed by the estimation of the initial state probability vector and state transition probability matrix of hidden Markov model (HMM). The neural network needs to be pre-trained layer-by-layer in an unsupervised manner and the weight matrix was then fine-tuned by the initialization of the pre-training using mini-batch gradient descent in a supervised manner. For each of the 22 cell types/tissues in the training sets, the enhancer and non-enhancer classes were constructed independently using the optimal model of a hidden layer with 50 hidden units to train PEDLA. (Liu, Li, Ren, Bo, & Shu, 2016).

EP-DNN is a deep learning approach introduced to predict the presence and types of enhancers in the human genome. The expression levels of the histone modifications at the peaks of the functional sites and its adjacent regions were used by this model as features. Four different cell types: H1, IMR90, HepG2, and HeLa S3 were employed by EP-DNN. The

positive controls used are P300 binding sites as enhancers, and negative controls used are TSS and random non-DHS sites as non-enhancers. The enhancer prediction involved a fully connected DNN with 480 inputs, 1 output, and softplus activation functions for each neuron. Stochastic gradient descent was applied in mini batches of 100 samples where the training was held. A dropout rate of 0.5 was used to avoid overfitting. The optimal architecture of EP-CNN consists of 3 hidden layers, with 600 neurons in the first layer, 500 in the second, and 400 in the third. Only 5 epochs of training are sufficient to achieve a convergence on the mean squared error (Kim et al., 2016).

Moreover, there is a hybrid framework introduced which combined deep learning technique known as CNNs and bi-directional long short-term memory networks (BLSTMs). This framework is known as DanQ and was developed to scan sequences for motif sites through convolution filtering (Quang & Xie, 2016). DanQ comprised of one convolution layer, one max pooling layer, and a BLSTM layer. A recurrent layer was included after the max pooling layer so that motifs can match a regulatory grammar governed by physical constraints. These physical constraints control the *in vivo* spatial arrangements and frequencies of combinations of motifs. An example of a feature associated with tissue-specific functional elements is enhancers. Lastly, a dense layer of rectified linear units and a multi-task sigmoid output formed the last two layers of the DanQ.

Besides that, a newer enhancer prediction tool that implemented deep learning based hybrid architecture to predict enhancers based on sequence feature alone such as BiRen is introduced (Yang et al., 2017). The common enhancer feature patterns can be directly obtained from the DNA sequence alone using BiRen. This model has demonstrated superior accuracy, generalizability, and robustness in enhancer prediction as compared to the other state-of-the-art enhancer prediction tools that are based on sequence characteristics.

A recent deep learning approach that was developed for the prediction of methylation states in single cells and modelling of the sources of DNA methylation variability is known as DeepCpG. We can see that the use of deep learning has been actively implemented in the other areas besides enhancer prediction. This model is built from a DNA module, a CpG module, and a multitask Joint module. The DNA module is a CNN with convolution layers, pooling layers, and one fully connected hidden layer. The non-linear embedding layer that used to model dependencies between CpG sites within cells resides within the CpG module, which is followed by a bidirectional gated recurrent network (GRU) to model dependencies between cells (Angermueller, Lee, Reik, & Stegle, 2017). Other than that, there is other deep learning approach that has been used in the biological field such as using the deep neural network for the prediction of splicing patterns in individual tissues and differences in splicing patterns across tissues (Leung, Xiong, Lee, & Frey, 2014).

2.7 Summary

This chapter construed some biological background that is necessary to understand the remaining chapters as well as previous works that are related to our study in this thesis. To sum up, original three-stage approach suffers from the high false positive rate. Besides, the traditional three-stage approach often requires some post-clustering steps in order to obtain motifs that approximate the real motifs. Since the partial set of input is sufficient for finding the motifs that repeatedly resides within the genome, it is not compulsory for motif finding tools to implement an enumerative search for the motifs in the whole genome. This is because only the predictive rate can be improved even when a more than sufficient amount of input set was used. We can see that this approach can be improved with the use of deep learning technique. A lower

level of the motif features can be learned by the model to build higher and more abstract features that can approximate the real motifs. The use of deep learning approach together with ensemble approach can significantly enhance the performance of the classifier in overall and significantly reduced the number of false positives in motif identification. This can be seen in the independent works described in Section 2.5 and 2.6. In addition to that, we also found out that the inclusion of gaps as motif features has not been taken seriously in any of the deep learning models yet. The inclusion of gaps as motif features is said to be crucial in determining the location of TFBSs especially the recognition sites of enhancer-bound transcription factors (Fletez-Brant, Lee, McCallion, & Beer, 2013). The current models also emphasized on the independence of the motifs features rather than the dependency features of the motifs. The adjacent bases should be considered as well in the recognition of TFBSs. In the next chapter, we provide a full description of DeepFinder.

CHAPTER 3

DNA MOTIF PREDICTION USING FEATURE-BASED AND DEEP LEARNING APPROACHES

3.1 Introduction

Motif identification has always been a field of interest to the bioinformatics community. The challenges that faced by those researchers is to find a group of motifs that approximate the real motifs. Thus, the first technical contribution in finding the best cluster of motifs using deep learning techniques is presented in this chapter. We proposed an improved three-stage method which is different from the past three-stage method where the previous researchers encountered high false positives in their motif identification task.

In this chapter, the rationale of proposed three-stage method is to predict the location of TFBSs which also termed as motifs using feature learning method. This approach used an ensemble of four motif finding tools for initial prediction of seed motifs which obtained from a subset of complete input DNA sequences. Stacked Autoencoders (SAEs) were implemented to learn those features associated with the seed motifs for the prediction of the TFBSs in the remaining sequences later on. Our novel method is named as DeepFinder that enhanced the three-stage method using the assembled results from several motif finding tools (e.g. MEME (Bailey & Elkan, 1995), BioProspector (Liu, Brutlag, & Liu, 2001), MDscan (Liu, Brutlag, & Liu, 2002), and MotifSampler (Thijs et al., 2001)) together with deep learning and other computational methods for motif identification. Until today, these motif finding tools are popular and widely adopted methods among the biologists and researchers due to the robustness

of their methods used which include position weight matrix (PWM) and expectation maximization.

In order to compare our proposed framework with other methods, we chose MAST (Bailey & Gribskov, 1998) and FIMO (Grant, Bailey, & Noble, 2011) in this case. These tools are capable of returning motifs with the highest score and to be selected as the identified motifs. Even though these two tools have been developed for a long time but the developer site has actively updated the commits of their feature branch to the master branch software tool. Therefore, it is reliable to pursue these two tools in the MEME suite. Top three profile matrix representations from the MEME results are scanned using MAST and FIMO tool which are available in the MEME Suite. The MAST, FIMO, and DeepFinder results are compared and shown in Table 4.3 and 4.4. The experiments showed that DeepFinder has better predictive power compared to the results of MAST and FIMO.

The remaining chapter is organized as follows: a full detail of proposed three-stage method is presented in Section 3.2 which includes a further description of the motif prediction system. In Section 3.3, performance evaluation metrics and experimental results for MAST, FIMO, and DeepFinder are addressed and compared. Finally, the discussions are presented in Section 3.4 and the whole chapter is concluded in Section 3.5.

3.2 Description of Proposed Framework

An assumption made by the heuristic three-stage approach is that binding sites bound by transcription factor have limited variations. Thus, a partial set of DNA sequences should be adequate to recognize the key protein-DNA codes. Unfortunately, there is no theory in existence that supports the implementation of this method until a recent publication by Zia and Moses

(2012). They reported that in order to predict motifs in DNA sequences, a sufficient number of input sequences is more than enough and more will only marginally enhance the predictive rates. An earlier empirical finding is aligned with this discovery which stated that the prediction accuracy does not improve when inputting more sequences (Hu, Li, & Kihara, 2005). The authors suggested “*one can input only partial input sequences to a motif discovery algorithm to obtain a motif model and then use this model to find motifs in the remaining sequences. In this manner, a significant reduction in the running time can be achieved without sacrificing the prediction accuracy.*” Another related approach is the seed-driven method of deterministic motif discovery. In this method, motifs are first identified in the sequence parts with highest priors and then used as seeds in the entire set of sequences for motif discovery (Sandve & Drablos, 2006). This idea of using a subset for initial motif prediction is not new. It is shown by MDscan which was the first tool to tackle on large-scale ChIP-on-chip (Chromatin Immunoprecipitation Microarray) dataset by generating initial motif patterns from a subset of input and use selected motif patterns to detect associated sites in the remaining dataset for the purpose of final motif profiles identification. It is encouraging to implement this method together with deep learning techniques to observe their respective results on higher eukaryotes.

Furthermore, this finding has not been put into practice or capitalized in algorithm design. It is probably insufficient practical guideline and significant evaluation has been conducted to find out the impacts and robustness of the finding. For example, choosing a proper method to select a subset and how to perform the search in the remaining sequence. The recent technologies enable genome-wide identification of regions bound *in vivo* by a given TF such as ChIP, ChIP-on-chip or ChIP-seq. In contrary, these regions often range from few dozen base pairs to few hundred base pairs (Zambelli, Pesole, & Pavesi, 2012). The advancement of this technology allows more regions of TFBSs can be accurately identified but not the exact

locations of TFBSs. Thus, these locations hold important features to some extent for the recognition of enhancer binding regions.

This project examines and comes up with some improvements on the use of the three-stage method for DNA sequence motif discovery in DeepFinder. DeepFinder is based on the three-stage method: (1) randomly select subset of DNA sequences for initial motif finding; (2) apply *de novo* motif discovery tools to select most potential putative motifs; (3) use the putative motif profiles to scan the binding sites in the remaining sequences not use in the initial subset. DeepFinder doesn't need to perform motif search in the whole dataset, thus, avoid exhaustive enumeration. Through this method, it is believed that any subset of the input sequences of a reasonable size can be used to build prominent motifs. Figure 3.1 depicts the DeepFinder's framework.

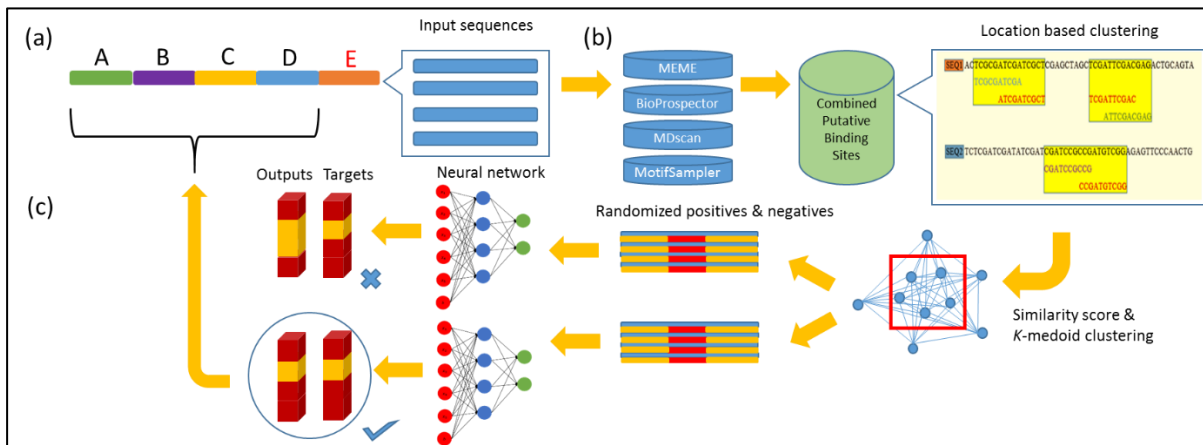


Figure 3.1: An overview of DeepFinder framework. (a) We randomly partitioned the dataset into five subsets. (b) Apply four *de novo* motif discovery tools on the partial input sequences E to predict putative motifs and respective binding sites. The top three motifs of four tools were clustered according to the location of the putative binding sites. Those binding sites that are overlapped by at least 1 bp or located next to the other are considered belonging to the same cluster. The pairwise similarity score is computed among the clusters, *k*-medoid clustering generates two clusters, with each cluster, added one hundred bases to both directions of binding sites and randomized with negative controls. The sequences are then converted into representation using the 76 designed features. The illustrated figure in details is presented in Section 4.2.2. (c) Stacked autoencoder, a deep learning technique is used to train the normalized cluster and only one cluster is chosen to perform the next task based on the size of the clusters generated. Those with bigger size cluster is constructed as a classifier to predict in the remaining DNA sequences.

In Stage 1 and 2, a subset is randomly selected from the whole dataset to be treated as input sequences for four *de novo* motif discovery tools, e.g. Multiple EM for Motif Elicitation (MEME), BioProspector, Motif Discovery scan (MDscan), and MotifSampler. The four *de novo* motif discovery tools are presented by a software called Toolbox of Motif Discovery (Tmod) which is developed by Harvard research team for Windows operating systems (Sun et al., 2010). For each *de novo* motif discovery tools results, we select top three motifs with their respected putative binding sites and combine them together as one before further processing in Stage 3. As in Stage 3, the assembled results are clustered according to the location of the putative binding sites. A putative binding site that is overlapped or located next to the other is considered belonging to the same cluster. After obtaining all the clusters, the pairwise similarity is computed for each and every possible pair of clusters formed and populate them to a distance matrix. Two clusters are generated by *k*-medoid clustering technique and the larger size cluster is trained as a classifier to predict the remaining motifs in the DNA sequences. This is to avoid missing binding sites from the smaller cluster and it is necessary to pursue larger cluster in order to avoid inadequate training data for the classifier model even though there is no specific guideline on the data size. Therefore, it is utterly important to keep the generated number of cluster small to avoid any missing significant binding site candidates. Moreover, the use of all candidate motifs is unnecessary. Stacked autoencoder is well known for its feature discovery especially in unlabelled data and its capacity in part-whole decomposition. Also, a single stacked autoencoder acquires greater expressive power compared to any deep learning neural network.

3.2.1 Motif Features

The feature is an important element that is crucial to differentiate or distinguish between different objects or behaviours. Motif feature is the necessary component of the learned model in order to distinguish between motifs and non-motifs. Osada, Zaslavsky, and Singh (2004) have reported a study of a larger set of *Escherichia coli* TFs together with their respected binding sites stated that binding site prediction can be improved by the use of dependencies among adjacent bases. It is important to acknowledge the work by Yáñez-Cuna et al. (2014) since the ten datasets used in this research is located within enhancer bound regions. The team has showed that a set of motifs enriched in all four enhancer classes which included some motifs of broadly expressed activators (e.g., AP-1 and STAT) and motifs that consists of repeated dinucleotides CA, GA, or CG (i.e., all possible dinucleotides except TA) can be used as features to determine whether it is an enhancer binding region or not. Dinucleotide repeat-like motifs are proven to be enriched in regions bound by different regulatory proteins in separate studies (Li et al., 2008; Wang et al., 2012). Therefore, the dependency features of TFBSs that include some k -mer features are utilized since they are often used in TFBS prediction. Each probability of occurrence tallied by the features is counted for every shifting size of the single base pair. There are total of seventy-six features used in DeepFinder which has the original PWM models that include A, C, G, and T occurrences, all possible 3-mer frequencies (64 features formed by A, C, G, T), CA, CG, GA, and GC-count (selected 2-mer frequencies), as well as their respective probabilities of occurrence. The feature value of a 2-mer k is given by $f(k) = c(k)/c(*)$ where $c(k)$ is the frequency count of k -mer k in a DNA sequence. The size of k -mer is limited to certain extent, for example, 4-mer size which produces 256 possible bases combinations is not used because this feature will yield a sparse representation of the DNA sequences where most of the representations are consist of zero value. Feature scaling

normalization method employed in this study is using this formula $X' = \frac{X - X_{min}}{X_{max} - X_{min}}$, where X_{max} is the highest frequency and X_{min} is the lowest frequency of the particular feature.

3.2.2 Motif Similarity Measures

The similarity measure is a method employed to examine how similar is the clusters in the putative binding sites. Those putative binding sites are of the same region or at least overlapped by one base pair is categorized as the same cluster. The similarity measure of the binding sites is defined by two cluster motifs x and y . Considered $A(x)$ be the set of binding sites defined by cluster motif x and $A(y)$ be the set of binding sites defined by cluster motifs y . Next, let $A(x) \cap A(y)$ be the set of regions occurred by at least one site in x and one site in y . This is what we called the intersection of bases between two cluster motifs. It is followed by $A(x) \cup A(y)$, is the set of regions occurred in any site of x and y in which we called the union of bases between two cluster motifs. The total number of nucleotides of all the regions in $A(x) \cap A(y)$ and $A(x) \cup A(y)$ is assigned as $|A(x) \cap A(y)|$ and $|A(x) \cup A(y)|$, respectively. The similarity of x and y , denoted $\text{sim}(x, y)$, is described as $|A(x) \cap A(y)| / |A(x) \cup A(y)|$. Note that $0 \leq \text{sim}(x, y) \leq 1$ and $\text{sim}(x, x) = 1$ (Wijaya, Yiu, Son, Kanagasabai, & Sung, 2008). After the pairwise similarities score is computed among predicted binding sites, the sum of the cluster's similarity score is calculated using this formula: $\sum \text{sim}(x_m, y_n) = |A(x_1) \cap A(y_1)| / |A(x_1) \cup A(y_1)| + \dots + |A(x_{m-1}) \cap A(y_{n-1})| / |A(x_{m-1}) \cup A(y_{n-1})| + |A(x_m) \cap A(y_n)| / |A(x_m) \cup A(y_n)|$ which concerns about the different length of binding sites. Thus, the end result of similarity score is computed as $\text{sim}(x, y) = 1 - \frac{\sum \text{sim}(x_m, y_n)}{\sum s_o}$, where s is the total count of window shifting in different length of the paired binding sites in both motifs. Examples of clustering and how similarity score is calculated are depicted in Figure 3.2 and 3.3. Finally, the final score is used

to populate the distant matrix to be used in k -medoid clustering algorithm. We described the clustering technique for the selection of similar binding sites that will be constructed as classifier later on in the next section.

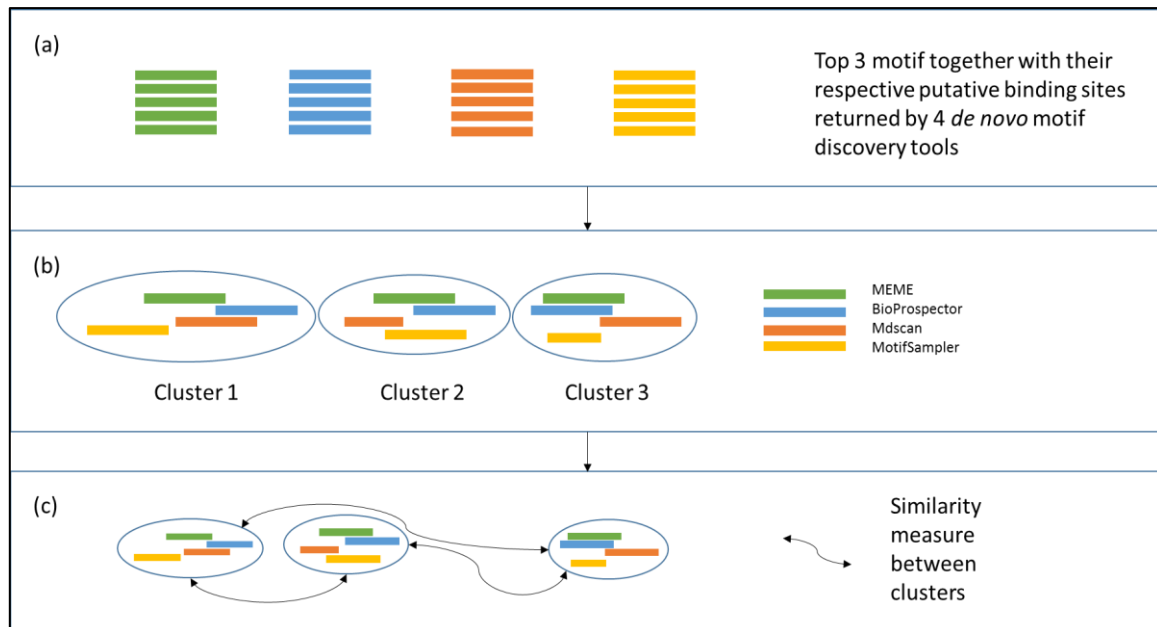


Figure 3.2: An overview of clustering technique and motif similarity measure. (a) The binding sites of top three motifs are accumulated after obtained from the four *de novo* motif discovery tools. (b) Those overlapped binding sites by at least 1 bp or next to one another based on their locations are grouped under the same cluster. (c) The similarity measure is computed among the clusters and populates their scores into distance matrix for the purpose of k -medoid clustering later on.

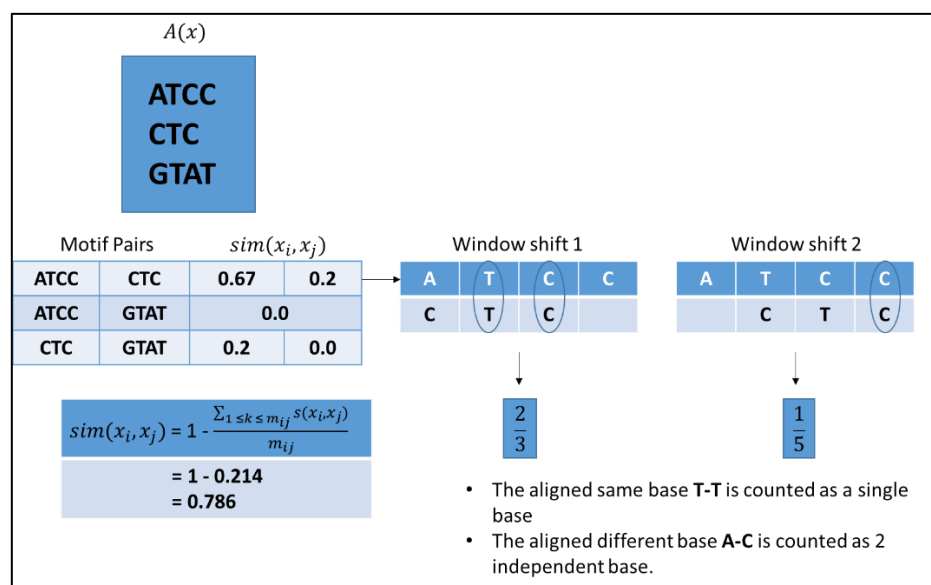


Figure 3.3: An example of how similarity score is calculated using three motif pairs.

3.2.3 Motif Clustering Technique

This section discussed the clustering technique used in the learned model to select similar putative binding sites and to gather them into the same cluster before training as a classifier. The two commonly used clustering techniques are *k*-means and *k*-medoids algorithms. In this study, we employed *k*-medoids algorithm because *k*-means is highly susceptible to the effects of outliers (Broin, Smith, & Golden, 2015). As for *k*-medoids algorithm, a group mean is computed and the cluster member with the smallest total pairwise distance to all other cluster members is adopted as the group representative. *K*-medoids algorithm is also more robust to noise and outlier effects, thus, a repeated multiple alignment calculations for each cluster can be avoided (Broin, Smith, & Golden, 2015). In a genome-wide study by Hannenhalli and Wang (2005), binding site prediction can be enhanced by clustering the binding sites in which the binding motif is represented by a mixture of PWMs instead of PWM alone. Therefore in our study, we set the *k*-medoid clustering outputting two clusters of binding sites using C Clustering Library and the corresponding Python C extension module Pycluster (de Hoon, Imoto, Nolan, & Miyano, 2004). It is vital to pursue more sequences in a cluster in order to avoid insufficient training data since there is no practical guideline on using how many sequences for the training process. Deep learning technique called stacked autoencoder is presented in the next section.

3.2.4 Deep Learning Classifier Model

Deep learning has astonishing performance in many areas such as business, technology, medicine, etc. The titan companies such as IBM, Apple, Google, Facebook, and Baidu are investing a huge amount of money in setting up deep learning lab to improve their current technologies such as image or pattern recognition, speech recognition, and text document

classification. In bioinformatics, the recent deep learning practitioner called Alipanahi and the team (2015) found out that deep learning surpassed other state-of-the-art methods when used to predict sequence specificities, even when training on *in vitro* data and testing on *in vivo* data. Furthermore, another team by Li et al. (2016) have applied deep learning in feature selection. Deep learning can model non-linearity and thus select a subset of features at the input level for multi-class data.

DeepFinder is built based on the larger size cluster obtained by *k*-medoids and trained by stacked autoencoder to predict TFBSs in the associated DNA sequences. It has shown astonishing results in motif prediction and the performance is presented in the Results section. Figure 3.4 illustrates the stacked autoencoder used to build DeepFinder.

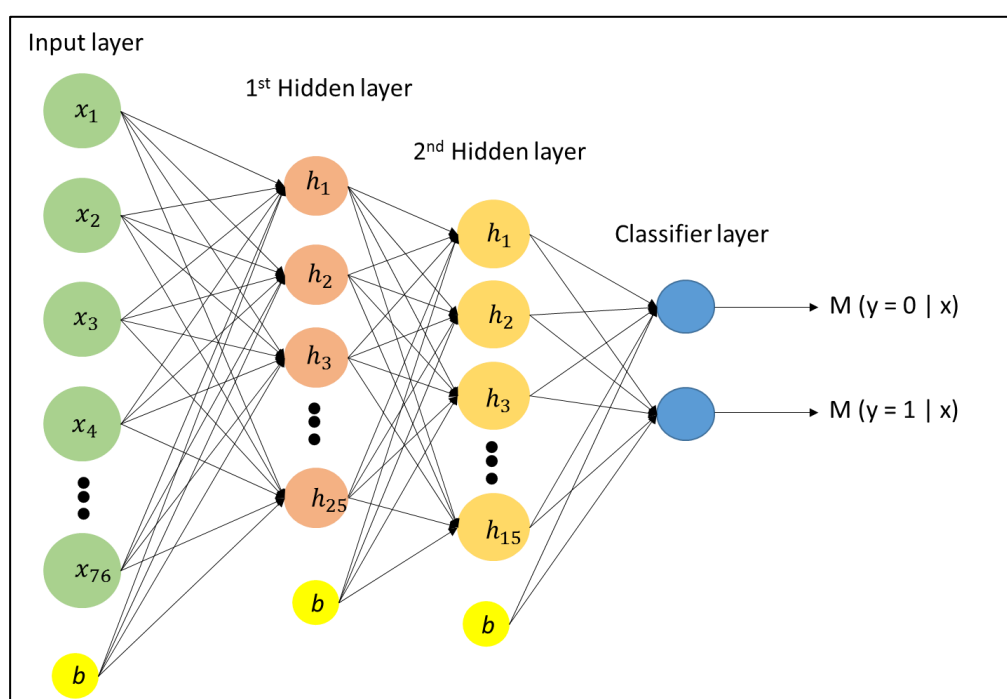


Figure 3.4: The neural network resides within DeepFinder.

It is first of its kind to employ stacked autoencoder in motif prediction where previous work was built based on convolutional neural networks called DeepBind (Alipanahi, Delong, Weirauch, & Frey, 2015). The training of DeepFinder requires an equal amount of positive and

negative controls and the negative controls used are complement intervals of the datasets retrieved from Galaxy Online (Giardine et al., 2005). The five-fold cross-validation experiment was run for five iterations each and it is proven to be sufficient for the classifier to learn the important features. Stacked autoencoder is commonly used in MNIST digit classification (Vincent, Larochelle, Bengio, & Manzagol, 2008), however, it did not gain popularity in motif prediction yet. The capabilities of stacked autoencoder are modelling complex systems with non-linear structures and learning a high-level representation of features. We employ DeepLearnToolbox (Palm, 2012) together with seventy-six features to ensure the model fully captured the probabilistic occurrences of the datasets. The neural network resides within DeepFinder as shown in the figure above where 76 engineered sequence features together with bias term act as input to the network. The same architecture was applied to all the datasets. Ten different DNA datasets of enhancer regions bound by the transcription factors are downloaded from UCSC Genome database (GRCh37/hg19 Feb. 2009 assembly). It was constructed by two layers of autoencoders with non-linear activation functions in which the outputs of each layer is connected to the inputs of the successive layer. The first layer of a stacked autoencoder tends to learn first-order features in the raw input and the second layer tends to learn second-order features corresponding to the patterns in the appearance of first-order features. The last layer is classifier that uses sigmoid activation function that maps the second-order features to the enhancer and non-enhancer labels. A fixed 5 epochs, learning rate value 1, the mini-batch size of 100, and dropout value 0.5 was found to be sufficient for feature learning. When the number of epochs is more than 5, the training becomes stagnant and no significant improvement in the learning can be obtained. The same number of epochs is employed for each layer. It is capable of modelling complex systems with non-linear structures and learning a high-level representation of features. The model is compelled to learn a compact representation of the

input through 25 hidden neurons in the first hidden layer and 12 hidden neurons in the second hidden layer. The formula used to obtain the required hidden neurons for the neural network, $N_h = \sqrt{(m+2)N} + 2\sqrt{N/(m+2)}$, introduced by Huang (2003) where m is the number of output neurons and N is the number of input neurons. In the neural network, the overall cost function (Ng, 2011) used here is

$$J_{sparse}(W, b) = J(W, b) + \beta \sum_{j=1}^{s_2} KL(\rho || \hat{\rho}_j)$$

$J(W, b)$ is the overall cost function which includes the weight decay term. The number of neurons in the hidden layer is represented by s_2 and the index j is summing over the hidden units in the network. $KL(\rho || \hat{\rho}_j)$ is the Kullback-Leibler (KL) divergence between a Bernoulli random variable with mean ρ and a Bernoulli random variable with mean $\hat{\rho}_j$. The weight of the sparsity penalty term is represented by β . According to Ng (2011), stacked autoencoder is capable of learning features that can form a good representation of the input or “part-whole decomposition” of the input. It also has greater expressive power compared to other neural network models. In this study, like any other neural networks, we implemented mini-batch stochastic gradient descent (SGD) and dropout regularization that resides within the learned model. These are the popular methods that are useful in acquiring promising results in computer vision, speech and face recognition.

3.3 Performance Evaluation Metrics

The performance evaluation metrics provided in this section are used to assess the performance of DeepFinder. The f -measure and false discovery rate (FDR) are both used to evaluate the classifier’s accuracy. The f -measure, f is given by $f = \frac{2pr}{p+r}$, where p and r are

precision and recall, respectively (Manning, Raghavan, & Schütze, 2008) and FDR is defined as $FDR = 1 - p$ (Benjamini & Hochberg, 1995). F -measure conveys the balance between the precision and recall. FDR identifies more true positive associations overall as compared to p -value measures of significance. Precision and recall are described as $p = \frac{TP}{TP+FP}$ and $r = \frac{TP}{TP+FN}$, where TP , FP , and FN are counts of true positives, false positives, and false negatives from the fivefold cross-validation experiment. Figure 3.5 illustrates TP , FP , FN , and TN on TFBS binding region and non-TFBS binding region prediction. The meaning of true positive (TP) is where the region (100bp symmetric margins on both sides of TFBS) is correctly identified as TFBS binding region; false positive (FP) is where the region is incorrectly identified as TFBS binding region; true negative (TN) is where the region is correctly identified as non-TFBS binding region, and false negative (FN) is where the region is incorrectly identified as non-TFBS binding region. Sensitivity or true positive rate (TPR) is defined as $TPR = \frac{TP}{TP+FN}$ and specificity (SPC) or true negative rate is defined as $SPC = \frac{TN}{TN+FP}$. These are widely used statistical measures in bioinformatics for the purpose of examining the performance of classifier. The classifier mainly differentiated motifs and non-motifs (2-class classifications), thus, we employ Matthews correlation coefficient (MCC) which invented by Matthews, B. W. (1975) to measure the quality of binary classifications and can be assessed using this formula: $MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$.

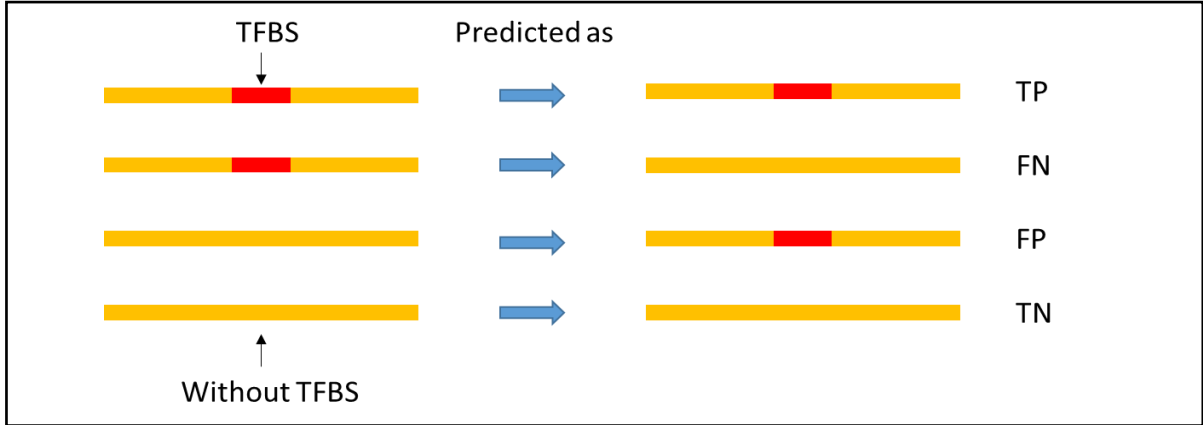


Figure 3.5: A visual description of *TP*, *FN*, *FP*, and *TN* on TFBS binding region and non-TFBS binding region prediction.

In the next section, ten different enhancer-bound transcription factor datasets are presented and followed by experiment setup which bestows the computing elements that involved in carrying out the experiments. Results and comparisons are conferred in Section 3.3.3.

3.3.1 Datasets

In this thesis, ten different DNA datasets of enhancer regions bound by the following transcription factors are downloaded from UCSC Genome database (GRCh37/hg19 Feb. 2009 assembly). These datasets are selected because the binding sites that are accurately predicted can be used to determine the enhancer regions which could possibly bound by the respected transcription factors. Table 3.1 shows the number of site locations annotated in the database for the selected TFs. However, we only randomly selected five thousand locations and each is extended to both directions by adding symmetric margins of 1000bp along the genome.

Table 3.1: TFBSs obtained from UCSC Genome Table Browser.

Enhancer bound transcription factor	Total conserved TFBS
CREB	6822
GATA1	9248
P53	18282
P300	13189
SRF	24889
STAT1	14030
NFE2	19635
MEF2	41426
ELK1	10781
HNF4	14306

From the above total conserved TFBS showed, five thousand binding sites were randomly selected as a subset and one thousand extra bases were added to the symmetric margins of the binding sites. Partial binding sites (that will be used as seed later) which are twenty percent of the total binding sites is treated as input to four *de novo* motif discovery tools. In the experiment, we employed four motif finding tools that are based on PWM model, namely MEME, BioProspector, MDscan, and MotifSampler.

3.3.2 Experiment Setup

The results presented in the next section are the performance averaged over the fivefold cross-validation of both training and testing data. Experiments are conducted on a computer with an Intel Core i5, 1.7 GHz CPU, and 16 GB memory card. All source codes are written in Matlab language and Python language with installed libraries: numpy, scipy, Pycluster, Biopython. The essential source code developed in the study is shown in Appendix 1 and 2.

3.3.3 Results and Comparisons

In this section, we evaluated DeepFinder's performance with respect to MAST and FIMO using different performance evaluation metrics. A thorough assessment of DeepFinder is conducted based on a standard fivefold cross-validation. DeepFinder was trained by stacked autoencoder in each fold using twenty percent of a dataset and use the model to scan the remaining eighty percent dataset. While for MAST and FIMO, MEME tool was employed in each fold to acquire the top 3 motif representations where each fold was generated from randomly selected subset and used the best representation to search for the other TFBSs not occurred in the initial subset. Representation with the highest number of true positives predicted is chosen. The results and comparisons between MAST, FIMO, and DeepFinder presented later in this section are based on average values obtained from fivefold cross-validation.

A description of DeepFinder using CREB data set is conferred as follows. Four motif finding tools were used on the five CREB subsets and their respective top 3 motifs sequence logo are illustrated in Table 3.2. The binding sites were assembled and clustered into two groups and each cluster is trained using the deep learning classifier.

Table 3.2: Sequence logo for top 3 motifs using MEME, BioProspector, MDscan, and MotifSampler.

Tool	Rank	Sequence Logo
MEME	1	
	2	
	3	
BioProspector	1	
	2	
	3	
MDscan	1	
	2	
	3	
MotifSampler	1	
	2	
	3	

The comparisons between MAST, FIMO, and DeepFinder in terms of precision, recall, f -measure, false discovery rate (FDR), and accuracy are reported in Table 3.3 and 3.4. Since there are no AUC values for this study, the significance of the results generated using f -measure is shown in Table 3.5 where the p -value score is 0.00 which means that there is a significant difference on the f -measure values computed by each of the tools. It is clearly shown that MAST has performed poorly compared to FIMO and DeepFinder in all evaluation metrics. This is because MAST only used a single motif finding tool's result and only sequences that have the best overall match to a collection of motifs were selected. While FIMO's performance as in precision and FDR is slightly better compared to DeepFinder for ELK1, GATA1, P300, and SRF datasets. This is because the sequence of ELK1, GATA1, P300, and SRF have better matches to motifs. DeepFinder, as expected, was outperformed the other models for recall, f -measure, and accuracy for all datasets. In addition to that, the specificity of DeepFinder is over 0.97 for most of the datasets. Nevertheless, MAST and FIMO have no true negatives since the scanning is only conducted on positive controls and hence no specificity rates were demonstrated. The true negatives value was assigned as zero throughout the computation of evaluation metrics such as specificity and MCC.

Table 3.3: Comparisons of precision, recall, and f -measure between MAST, FIMO and DeepFinder (DF).

Enhancer bound TF	Precision			Recall			f -measure		
	MAST	FIMO	DF	MAST	FIMO	DF	MAST	FIMO	DF
CREB	0.60	0.99	1.00	0.03	0.45	0.97	0.06	0.62	0.99
ELK1	0.69	0.99	0.91	0.04	0.54	0.91	0.08	0.70	0.90
GATA1	0.68	0.99	0.95	0.04	0.53	0.94	0.08	0.69	0.95
HNF4	0.68	1.00	1.00	0.05	0.51	0.99	0.09	0.67	1.00
MEF2	0.80	1.00	1.00	0.01	0.20	0.98	0.03	0.33	0.99
NFE2	0.73	1.00	1.00	0.03	0.36	0.99	0.06	0.52	1.00
P53	0.71	0.99	1.00	0.05	0.49	0.99	0.09	0.66	0.99
P300	0.65	0.99	0.98	0.06	0.68	0.96	0.11	0.80	0.97
SRF	0.75	1.00	0.98	0.03	0.37	0.99	0.06	0.51	0.98
STAT1	0.67	1.00	1.00	0.04	0.44	0.99	0.07	0.61	0.99

Table 3.4: Comparisons of false discovery rate (FDR), accuracy, and Matthews correlation coefficient (MCC) between MAST, FIMO, and DeepFinder (DF).

Enhancer bound TF	FDR			Accuracy			MCC		
	MAST	FIMO	DF	MAST	FIMO	DF	MAST	FIMO	DF
CREB	0.40	0.01	0.00	0.03	0.45	0.99	-0.62	-0.06	0.97
ELK1	0.31	0.01	0.09	0.04	0.54	0.89	-0.54	-0.05	0.81
GATA1	0.32	0.01	0.05	0.04	0.53	0.94	-0.55	-0.05	0.89
HNF4	0.32	0.00	0.00	0.05	0.51	1.00	-0.55	-0.04	0.99
MEF2	0.20	0.00	0.00	0.01	0.99	0.99	-0.44	-0.04	0.98
NFE2	0.27	0.00	0.00	0.03	0.36	1.00	-0.51	-0.03	0.99
P53	0.29	0.01	0.00	0.05	0.49	0.99	-0.53	-0.06	0.99
P300	0.35	0.01	0.02	0.06	0.67	0.97	-0.57	-0.05	0.94
SRF	0.25	0.00	0.02	0.03	0.37	0.98	-0.49	-0.03	0.96
STAT1	0.33	0.00	0.00	0.03	0.44	0.99	-0.56	-0.04	0.99

Table 3.5: One way ANOVA test of MAST, FIMO, and DeepFinder (DF).

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	4.126927	2	2.063463	335.3407	0.00	3.354131
Within Groups	0.16614	27	0.006153			
Total	4.293067	29				

Figure 3.6 and 3.7 showed the precision rate and recall rate for five subsets. The discrepancy of precision and recall rate were observed in SRF where its subset 2 and 5 have performed poorly in contrast to other subsets for that particular transcription factor. This is possibly due to the outliers that reside within the subset that caused the average result to be lower than the other subsets. DeepFinder's overall f -measure, precision, recall, and accuracy rates for the ten datasets were reported in Figure 3.8. Two datasets that have lower performance were observed in ELK1 and GATA1 because of the outlier effects that possessed by a single subset. On the contrary, these two datasets have shown performance consistency in MAST and FIMO. Last but not least, the three models comparisons are depicted in Figure 3.9 which concludes that DeepFinder outperformed MAST and FIMO in overall results. As for precision and FDR, FIMO is outperformed DeepFinder and MAST because the individual of ELK1, GATA1, P300, and SRF better matches to motifs. While, DeepFinder scored higher than MAST and FIMO in recall, f -measure, and accuracy. There is an extreme gap in between DeepFinder

and MAST in terms of recall, f -measure, and accuracy because MAST only used a single motif finding tool's result and only sequences that have the best overall match to a collection of motifs were selected. Thus, a single best sequence might be missed out in this way.

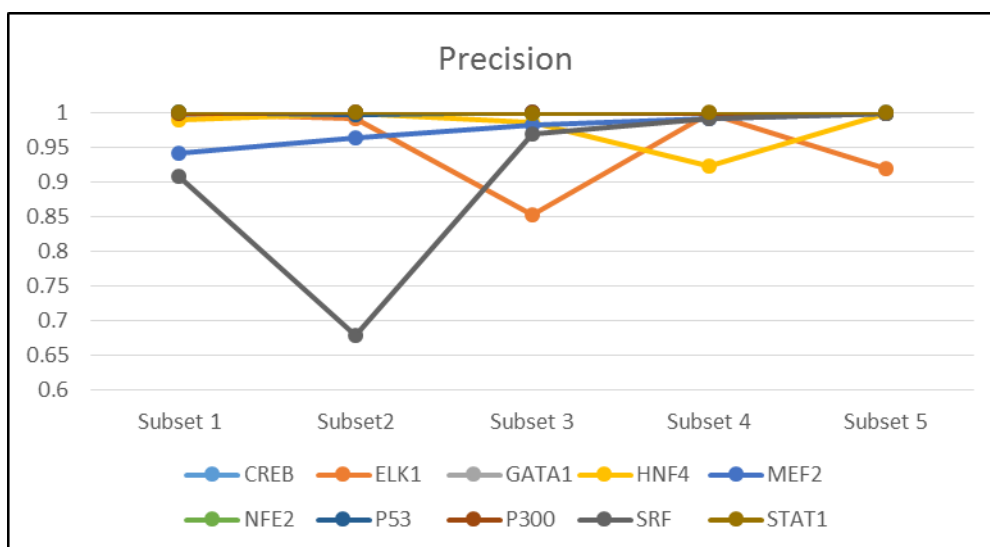


Figure 3.6: Precision values for each of the subsets.

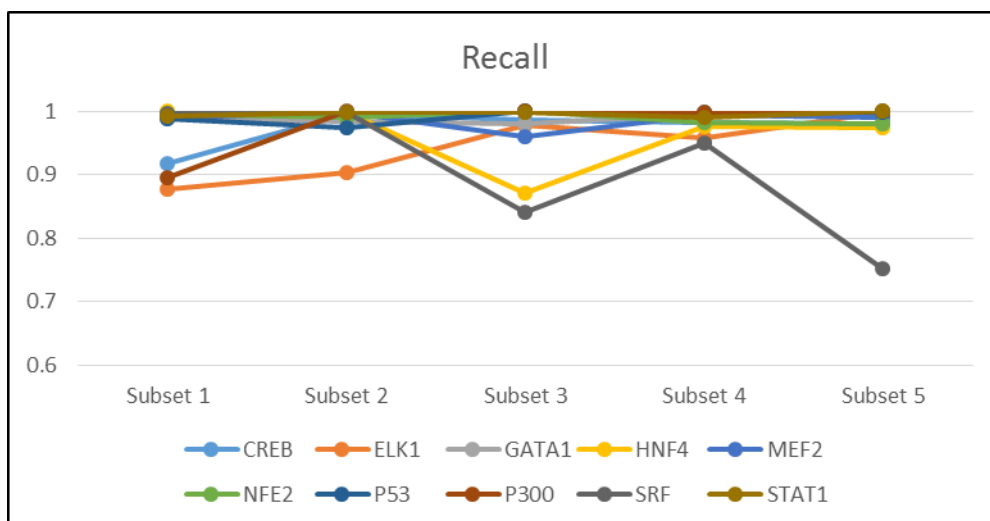


Figure 3.7: Recall values for each of the subsets.

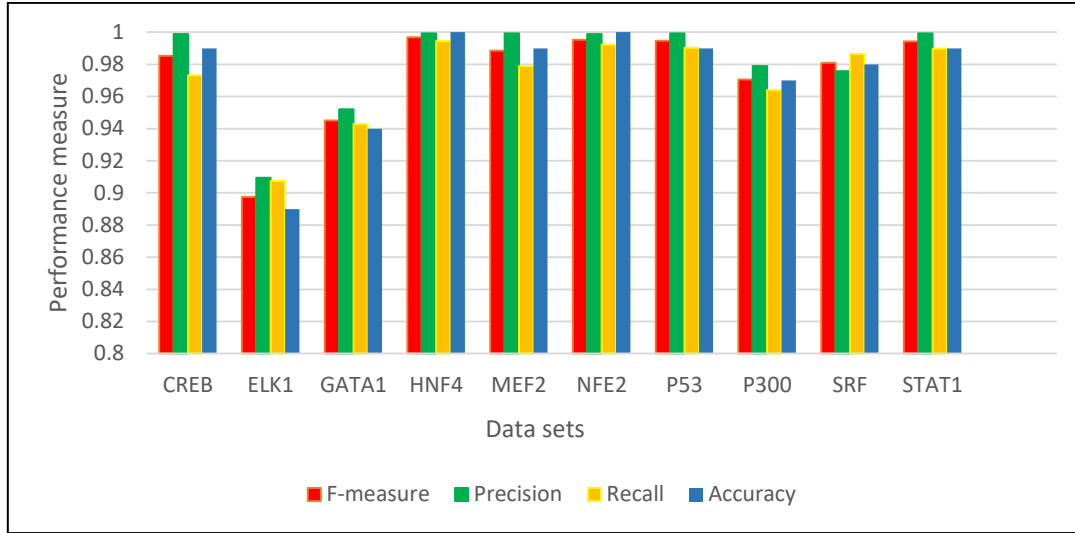


Figure 3.8: Overall f -measure, precision, recall, and accuracy for ten datasets.

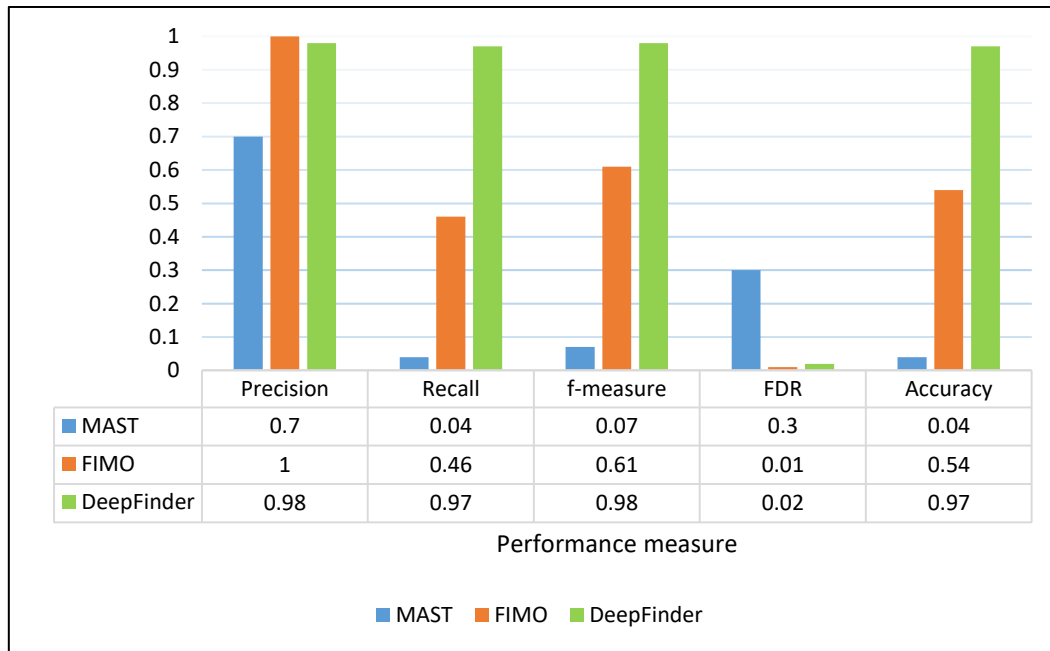


Figure 3.9: Average precision, recall, f -measure, false discovery rate (FDR), and accuracy for MAST, FIMO, and DeepFinder.

3.4 Discussion

The proposed deep learning three-stage method indeed shown promising results when performed on the ten datasets and it is also convincing that the implementation of deep learning technique namely stacked autoencoder, as well as integrated features based approach, is a major

success in improving the three-stage method. Overall performance as reported in the previous section, DeepFinder is obviously performed better than MAST and FIMO and it is not inclined to overfitting during the fivefold cross-validation experiments. Our approach also excels in false discovery rate (FDR) which expresses the proportion of the predictions that are false positives and f -measure which is the harmonic mean of precision and recall. This is benefited to the work where researchers can use sufficient input data rather than whole input set to predict the enhancer regions.

In this section, we discussed the possible reasons that caused poor performance of MAST and FIMO. It is probably driven by the single use of MEME result which ignores the strengths of other potential motif finding tools. Every single motif finding tools has its own strengths and weaknesses, therefore, it is encouraged to incorporate more tools which could take the benefits of other tools' strengths that a single tool lack of. Indeed, DeepFinder has better predictive power especially for those TFBS that binds within enhancer regions. The current best-performing algorithm has only sensitivity <13% and precision <35% (Tomba et al., 2005; Tran & Huang, 2014). Therefore, there is a need to enhance the motif prediction framework for the purpose of reduction in false positives.

Currently, there are no robust practical guidelines on three-stage method. Past studies by Wang and Lee (2009) reported that the number of false positive hits obtained from step (3) is excessively high. The inability of current motif model to represent the specificity of binding sites recognized by transcription factor contributed to this issue. This is why DeepFinder is introduced here and it has high predictive power in enhancer prediction. DeepFinder uses stacked autoencoder as the learning neural network to learn the features that are important to identify between the enhancer and non-enhancer regions. The deep learning technique resides within DeepFinder is capable of constructing higher-order features from lower level ones such

as dimer or trimer. These features will form higher order features that best represent motifs which will be used to search for potential enhancers in the remaining DNA sequences. Besides, DeepFinder also includes a dependent and independent model that represent the DNA sequences. It has been shown that independent and dependent features play a vital role in recognizing the motifs of the enhancer. Several efforts have been recommended to reduce the false positive rates, for example, using a position-dependent model, estimate a better threshold value to be considered a match hit during searching, and use of orthologous genes. Even though Position Frequency Model (PFM) has many weaknesses, it is still the most commonly used model compare to other more recently proposed models (Stormo, 2000; Boeva, 2016). Statistical independence among nucleotides within a motif was assumed by most algorithms in Position Weight Matrix (PWM) of *de novo* motif discovery tools (Eggeling et al., 2014). In this study, DeepFinder uses extended PWM model which includes different features that concerned about nucleotide dependencies. It can be seen that DeepFinder has successfully outperformed other classical models with the implementation of deep learning technique.

3.5 Conclusion

As a conclusion for this chapter, stacked autoencoder as classifier model in DeepFinder can predict the regulatory sequences accurately without any prior knowledge about transcription factor binding sites. DeepFinder only used general genomic sequence information rather than take into account biophysical or biochemical properties of the sequences. The inclusion of these properties could have the potential to enhance our approach. It is perhaps surprising that such a simple description of sequence features (*k*-mer frequencies) is capable of producing high accuracy in TFBSs classification.

CHAPTER 4

AN INTEGRATION OF GAPPED MOTIF FEATURE REPRESENTATION AND DEEP LEARNING APPROACH FOR ENHANCER PREDICTION

4.1 Introduction

In this chapter, an integration of gapped motif feature representation (GMFR) and convolutional neural networks (CNNs) is presented. We proposed a novel sequence encoding method called GMFR to fully utilize the strength of gapped motifs as an input feature for CNN learning. A gapped pattern that occurs repeatedly in a sequence or in a set of sequences is defined as a gapped motif (Michael, Nicolas, & Ukkonen, 2010). In this study, we employed CNN together with proposed GMFR to improve current enhancer prediction and compared our results with DeepBind (Alipanahi, Delong, Weirauch, & Frey, 2015) and LS-GKM (Lee, 2016). DeepBind employs deep convolutional neural networks to specify RNA- and DNA-binding proteins while LS-GKM employs advanced gapped k -mer based kernel functions together with support vector machine to predict regulatory elements. The strengths of CNN include location invariance and composition of higher order feature from lower level ones. The enhancer can be located either in upstream or downstream of the gene it regulates, unlike promoter where it can be found at the transcription start site of a gene. This is why CNN is chosen over the other deep learning techniques. Besides, a novel method called GMFR that represents DNA sequence is proposed in this study which allows the convolution of the sequence features by CNN while leaving gaps in between them. Based on our experimental results, the integration of both methods has shown consistency and achieved over 98% accuracy for ten different ChIP-seq datasets which were downloaded from DeepBind's literature. We believe that the advancement

in deep learning technique certainly improves the performance of enhancer prediction in the near future.

The remaining chapter is structured as follows: the description of the proposed framework GMFR-CNN is presented in Section 4.2 while in Section 4.2.1 a further description is given on the gapped motif feature representation. In Section 4.2.2, a classifier model called convolutional neural networks are conferred and experimental results for LS-GKM, DeepBind, and GMFR-CNN are addressed and compared in Section 4.3. We discuss the chapter in Section 4.4 and followed by conclusion in Section 4.5.

4.2 Description of Proposed Framework

Since mammalian enhancer sequences can be differentiated from random genomic loci using sequence features alone, our proposed framework GMFR-CNN is solely depended on sequence characteristics to maximize the discrimination between enhancers and non-enhancers. The possible substrings of length k are often referred to as k -mers and they are commonly used for the classification of regulatory elements. The nucleic acid A, C, G, and T are substituents of k -mer for DNA sequence. For the purpose of enhancer prediction, previous proposed k -mer frequency vector and its refined version called gapped k -mer vector were integrated into the SVM models (Fletez-Brant, Lee, McCallion, & Beer, 2013). Other than that, the experiments were done on predicting enhancers bound by P300 in 500 to 2, 500 sequences in mouse forebrain, midbrain, and limb also used k -mer sequence features. Unfortunately, when the binding sites involved are longer than 6 to 8 bp, this approach suffers from overfitting. When most motif finding algorithms consider continuous sequence motifs, it is crucial to take into account gapped sequence motifs. The inclusion of gaps in the motif allows the modelling of

short internal loops in RNA structures and enable for a small number of mutations in the conserved regions (Pervouchine, 2014). Therefore, enhancer motif prediction could be greatly improved by the implementation of gapped motif feature. This helps to discover motifs that are not discovered in the earlier continuous sequence motifs.

In recent years, the adoption of deep learning technique in bioinformatics field has started to gain popularity especially in the prediction of sequence specificities. This can be seen by the work of Alipanahi et al. (2015) and Kelley et al. (2016) where both methods have shown significant results in the overall sensitivity of prediction. Previously, there are some other machine learning methods implemented to predict enhancers. For example, DEEP framework (Kleftogiannis, Kalnis, & Bajic, 2015) and CSI-ANN (Firpi, Ucar, & Tan, 2010), both tools have been used extensively for enhancer prediction with an overall accuracy of 90.2% and 65.6% sensitivity.

4.2.1 Sequence Encoding

The design of DNA sequence representation is important for the training of the neural network in determining the features later on in the deeper layers. The weight represented by features means how conserved is that features in specific locations. In this study, our representation solely depends on sequence characteristics where it is sufficiently used to discriminate between enhancers and non-enhancers. This is because mammalian enhancer can be differentiated from non-functional regions by using the sequence features alone. This is clearly shown by the experiment done on EP300/CREBBP ChIP-seq datasets to differentiate between enhancers that operate in different tissues or at different developmental stages (Lee, Karchin, & Beer, 2011). First, we align a set of positive training sequences and compute an

extended position weight matrix (PWM). The representation built is served as a benchmark for negative control sequences later on. PWM is originally aligned for transcription factor binding sites only and now we extend it to capture longer sequences. The aligned sequences are defined as $S_n = \{s_1, s_2, s_3, \dots, s_n\}$, where S_n is formed by n number of aligned training sequences. PWM is commonly applied by researchers due to its simplicity and less tuning of parameters. The implementation of this technique provides an alternative to estimate the conservation of nucleotide position in the sequence. The formula used to form PWM for the positive training sequences is as follows:

$$P_{i,j} = \frac{1}{N} \sum_{i=1}^N I(S_{i,j} = k),$$

where i symbolizes four nucleotides (e.g. A, C, G, and T), and j resembles each of the nucleotides positions of length l , k is a set of symbols in the alphabets, and $I(\bullet)$ is an indicator function where it will have value 1 if $S_{i,j} = k$ and 0 otherwise (Guigo, 2003). Moreover, we employed a probabilistic method of nucleotide frequency for each of the sequences which derived as follow:

$$F_i = \frac{n_i}{\sum n_{i,i+1,\dots,i-1}},$$

where n is the count of nucleotides in one sequence. The probabilistic weight value is obtained at every nucleotide positions which describe as $W_{i,j} = \frac{P_{i,j}}{F_i}$ and then normalized using feature scaling method (Juszczak, Tax, & Duin, 2002). An example of how the probabilistic weight value is calculated from the extended PWM to probabilistic method is illustrated in Figure 4.1. The sequence length provided by DeepBind datasets is 101bp and we trimmed the size of the sequence length to 100bp for the purpose of compiling the representation into 10 x 10 matrix.

We arrange the sequence of normalized nucleotides as such to allow the kernel filter to convolve and leaves some gaps during convolution. Figure 4.2 depicts the gapped motif feature representation convolved by kernel size of 3. The kernel size 3 is chosen because the recognition site is separated by 0-21 bp in the genome of many organisms, thus, kernel size 3×3 convolves 10×10 matrix representation leaves a reasonable number of gaps in between them.

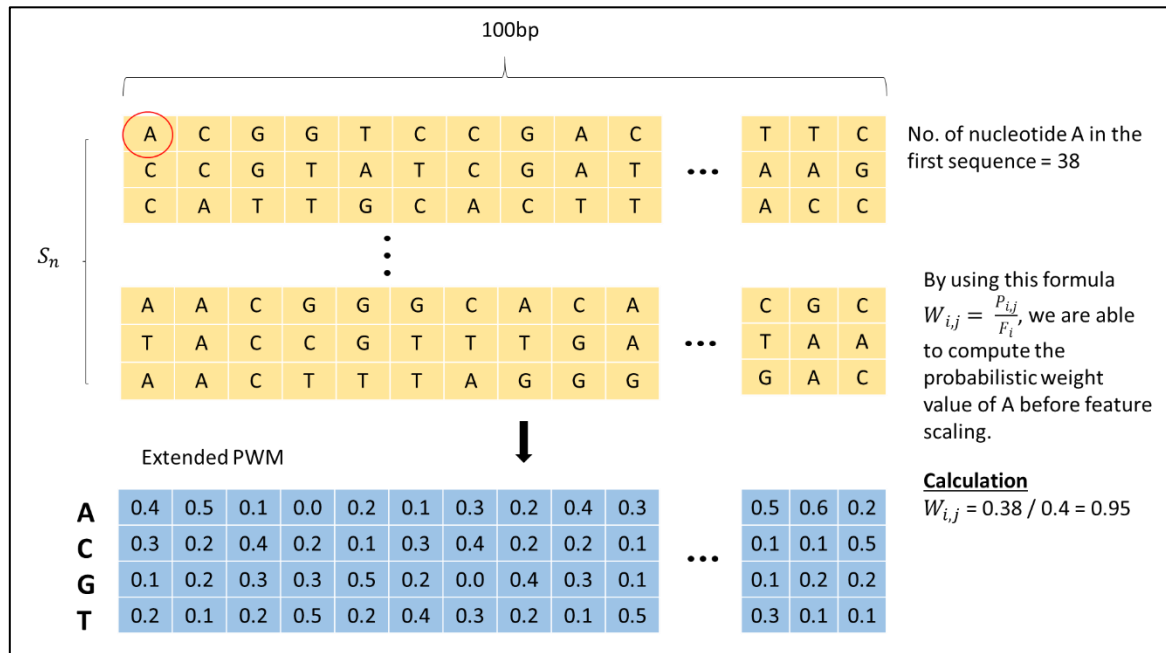


Figure 4.1: The computation of probabilistic weight values.

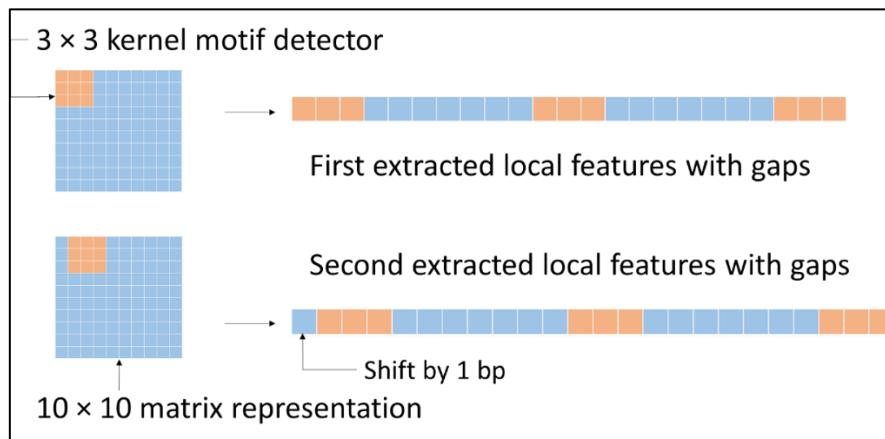


Figure 4.2: GMFR convolved by 3×3 kernel size motif detector.

4.2.2 CNN Architecture

The CNN architecture resides within GMFR-CNN is described as follows. The input to the neural network is where GMFR acts as a two-dimensional DNA “image” that is ready to be convolved by CNN which comprised of six layers including an input layer, two convolution layers, two subsampling layers, and an output layer. In the first layer, the GMFR which is in 10×10 matrices is ready to be convolved in the next layer by convolution and thus produce $8 \times 8 \times 6$ matrices where 6 is known as the number of output maps or neurons. This is generated by the kernel size of 3 to obtain different possible features and the resolution of the features are reduced to the scale of 2 in the following subsampling layer. In the third layer, a matrix representation of size $6 \times 6 \times 6$ is generated. Next, 12 feature maps were used in the second convolution layer and the rest of the parameters remained. Thus, a $4 \times 4 \times 12$ matrices were produced by this layer where the number of output maps is 12. After that, a subsampling layer took place in the next layer to compel the matrices to become $2 \times 2 \times 12$ matrices. Then, the resulting output maps are unwound into final feature vector containing 48 values ($2 \times 2 \times 12 = 48$). The sigmoid nonlinearity function was implemented in the last layer of the neural network as a simple classifier to perform classification task where value 1 is enhancer and value 0 is non-enhancer. In order to avoid vanishing gradient problem which occurred in earlier deep learning studies, rectifier unit (ReLU) was applied in GMFR-CNN. This unit is defined as $f(x) = \max(0, x)$.

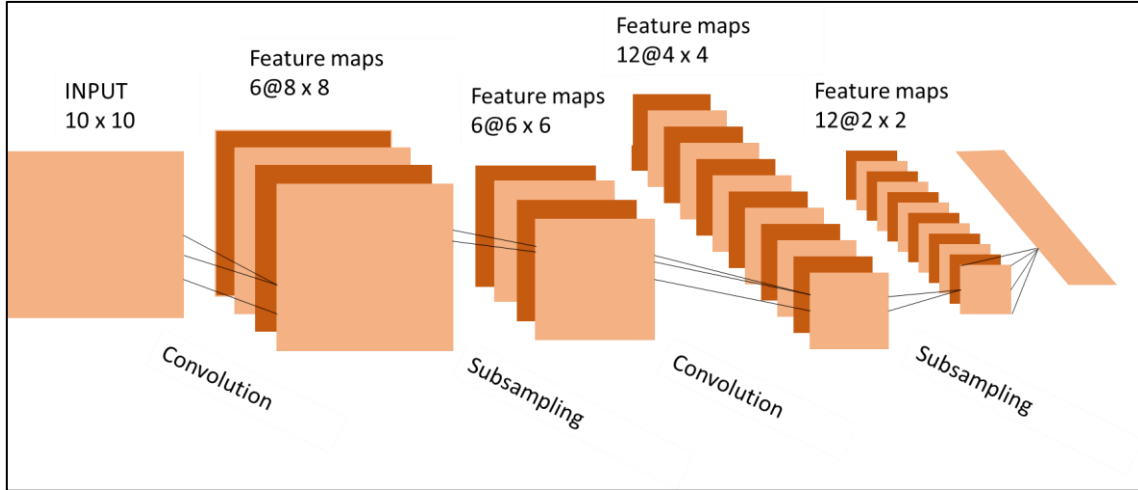


Figure 4.3: An architecture of GMFR-CNN.

During the development of the neural network for GMFR-CNN, different parameters were tested to obtain the most optimal area under a receiver operating characteristic (ROC) or also known as AUC where we found that learning rate of 0.8, 200 iterations, batch-size of 20, and 6, 12 of output maps in first and second convolution layer are most suitable. AUC is commonly used as a performance measure for machine learning algorithms (Bradley, 1997; Jiao & Du, 2016) and the detail of AUC is described in the next section. For configuring GMFR-CNN using ELK1 as an example, Table 4.1 shows the comparison in terms of AUC between different parameters. For the 250 iterations and 8, 16 output maps, it can be clearly seen that the AUC values obtained are 0.15 - 0.16% higher than AUC value of 0.9919 by using the default parameters in GMFR-CNN (0.8 in learning rate, 200 iterations and 6, 12 in output maps). However, if this parameter applied to CTCF, the cost of time being used will be more than 9 hours because the current default parameters used for CTCF is nearly 9 hours. This applies to small batch-size as well.

Table 4.1: Comparison in terms of AUC between different parameters for configuring GMFR-CNN using ELK1 as an example.

	Learning rate (0.1, 0.5, 1)	No. of iterations (100, 150, 250)	No. of output maps (2, 4; 4, 8; 8, 16)	Batch-size (10, 40, 80)
AUC	0.9749	0.9866	0.9904	0.9947
	0.9891	0.9895	0.9918	0.9875
	0.9828	0.9935	0.9934	0.9848

4.3 Performance Evaluation Metrics

We have assessed the performance of learned models using precision, recall, f -measure, accuracy, and AUC. Both precision and recall are defined as $p = \frac{TP}{TP+FP}$ and $r = \frac{TP}{TP+FN}$ respectively, where TP , FP , and FN are a count of true positives, false positives, and false negatives. The precision and recall are working in different direction. For example, high precision doesn't resembles high recall and it can results in low recall value. Due to the different direction of both performance metrics, it is encouraged to implement f -measure as the weighted harmonic mean for both metrics. F -measure can be described as $f = \frac{2pr}{p+r}$, where p and r are precision and recall (Manning, Raghavan, & Schütze, 2008; Powers, 2011). Higher f -measure value represents high precision and recall as well. As for accuracy, it demonstrates the proportion of true results, either true positive or true negative in a total positives and negatives control (Swets, 1988). It measures the degree of veracity of the five-fold cross validation experiments. Lastly, the AUC value is computed based on this formula, $\hat{A} = \frac{S_0 - n_0(n_0+1)/2}{n_0n_1}$, where n_0 and n_1 are the numbers of positive and negative testing set respectively, and $S_0 = \sum r_i$, where r_i is the rank of i_{th} positive testing set in the ranked list (Ling, Huang, & Zhang, 2003).

4.3.1 Datasets

Ten ChIP-seq datasets were downloaded from ENCODE (Landt et al., 2012). The number of sequences is in between 5,084 to 60,504 sequences and 26,052 in average. The list of transcription factor enhancer bound regions and their DeepBind model identity is stated in Table 4.2. The number of sequences contained in each dataset is described as follows.

Table 4.2: Ten ChIP-seq datasets, their DeepBind model ID, and the number of sequences.

TF	Model ID for DeepBind	No. of Seq
CEBPB	D00317.009	60504
TFAP2A	D00681.004	18470
YY1	D00710.007	17828
CTCF	D00753.001	45103
EP300	D00755.005	27413
TBP	D00817.001	14393
HNF4A	D00446.009	20305
CEBPD	D00318.004	10933
ELK1	D00360.007	5084
FOXA2	D00762.001	40489

4.3.2 Experiment Setup

In this study, we compared our model GMFR-CNN with LS-GKM and DeepBind models in terms of their predictive power using the ten ChIP-seq datasets. LS-GKM implements support vector machines technique and DeepBind employs convolutional networks, both are two well-known algorithms for classification tasks. We employed five-fold cross-validation experiment to assess GMFR-CNN’s performance. The 500 positives are randomly selected as the testing set without overlapping with one another and an equal amount of negatives are generated by randomly shuffled dinucleotides of the 500 positives. We referred to DeepBind’s literature on preparing the negative controls. The remaining of the datasets served as the training set. These steps are repeated for five times for fivefold cross-validation later on. In each fold, the learned model is used to predict on the testing set. All the experiments including

comparative tools were carried out using a computer with Intel Core i5, 1.7GHz CPU, and 16 GB memory, and the algorithm was employed in DeepLearn Toolbox Matlab and pre-processed using Python. Appendix 3, 4, and 5 showed the source code that played a crucial part of GMFR-CNN.

DeepBind was downloaded from this link <http://tools.genes.toronto.edu/deepbind/> and it is a command-line executable that can be used to score DNA or RNA sequences according to any RBP or TF model listed in the web repository. For each input sequence, the DeepBind executable scores each subsequence of a pre-determined length and returns only the maximum or the average over these scores. Larger scores indicated stronger binding. The scores themselves are on an arbitrary scale and vary from model to model due to variation in the quality of training data for different proteins. There are two things are needed to generate predictions with DeepBind: (1) a list of model IDs and (2) a list of DNA or RNA sequences. The model ID is stated in the file ‘example.ids’ and the list of sequences that are ready for prediction is saved into the file ‘example.seq’.

LS-GKM was downloaded from this link <https://github.com/Dongwon-Lee/lsgkm/> and it is also a command-line executable that is used for predicting regulatory DNA elements. Unlike DeepBind, LS-GKM allows users to train their own models and two executables are provided for training and prediction namely ‘gkmtrain’ and ‘gkmpredict’. ‘gkmtrain’ takes in positive controls and negative controls for training and generate a prefix of output which will then be used for scoring of any DNA sequences. Then, ‘gkmpredict’ takes in the prefix of output and testing set to generate output scores for each of the testing sequences. Table 5.3 shows the parameters used in GMFR-CNN, DeepBind, and LS-GKM. We use default parameters as mentioned by the authors that the parameters set are meant to achieve optimality in the

discovery of regulatory elements. Batch-size of 20 is sufficiently used to train the model in this study. If larger batch-size is used, overfitting could have happened to the model.

Table 4.3: Parameters used by GMFR-CNN, DeepBind, and LS-GKM.

Tools	Parameters
GMFR-CNN	Motif length = 23; learning rate = 0.8; number of iterations = 200; output maps = 6, 12; batch-size = 20
DeepBind	Motif length = 24; learning rate ranges between 0.0005 and 0.05; dropout expectation = 0.5, batch-size = 64, minimum iterations = 4000 and maximum 20, 000.
LS-GKM	Motif length = 11; batch-size = 5,000

4.3.3 Results and Comparisons

In brief, we have reported the performance of GMFR-CNN, LS-GKM, and DeepBind learner models. GMFR-CNN has accomplished higher predictive power with respect to LS-GKM and DeepBind for the ten datasets used. Overall, our model has obtained 11% and 14% higher in f -measure values as compared to LS-GKM and DeepBind. The average f -measure values of LS-GKM, DeepBind, and GMFR-CNN are 87% (± 1.0), 84% (± 1.0), and 98% (± 0.4). Besides, GMFR-CNN also exceeds both learner models by 13% and 12% in terms of their average accuracy. While the average accuracy rates of LS-GKM, DeepBind, and GMFR-CNN are 85% (± 1.0), 86% (± 1.0), and 98% (± 0.4). Last but not least, GMFR-CNN has obtained an average AUC value of 0.9979 while LS-GKM and DeepBind both have obtained 0.9683 and 0.9385. From the experiments, we realized that using genomic sequence feature alone is capable of identifying enhancers from non-enhancer regions with high f -measure and accuracy. Our model also has been proven that it is not affected by overfitting during fivefold cross-validation experiments.

Overall, as shown in Table 5.4, 5.5, 5.6, and 5.7 which report the mean $\pm$ SD of precision, recall, f -measure, and accuracy, our model, GMFR-CNN has shown significant improvement over the performance metrics used. As we can observe from the three models, except for recall metric, LS-GKM has performed the worst in all evaluation metrics. It is to our concern that GMFR-CNN has performed consistently and remains to be 0.96 and above for all the evaluation metrics shown. This also convinces that GMFR-CNN has a key advantage over a variety of TFs used for training and testing purposes.

Table 4.4 shows the mean $\pm$ SD of precision rates for LS-GKM, DeepBind, and GMFR-CNN together with their average mean $\pm$ SD of precision rates for all the ten datasets used. It can be observed that LS-GKM has scored the lowest precision rates as compared to both DeepBind and GMFR-CNN with average 80%. Other than that, TBP dataset has scored the lowest precision rates for both LS-GKM and DeepBind while FOXA2 has scored the lowest precision rate for GMFR-CNN. DeepBind and GMFR-CNN have consistently scored over 90% for the precision evaluation metric except for TBP which scored only 89%.

Table 4.4: Average precision rates of LS-GKM, DeepBind, and GMFR-CNN.

TF	LS-GKM	DeepBind	GMFR-CNN
CEBPB	0.84 $\pm$ 0.01	0.94 $\pm$ 0.008	0.99 $\pm$ 0.002
CEBPD	0.77 $\pm$ 0.01	0.94 $\pm$ 0.008	0.99 $\pm$ 0.002
CTCF	0.81 $\pm$ 0.01	0.95 $\pm$ 0.007	0.99 $\pm$ 0.003
ELK1	0.84 $\pm$ 0.01	0.91 $\pm$ 0.009	0.98 $\pm$ 0.005
EP300	0.79 $\pm$ 0.01	0.92 $\pm$ 0.009	0.99 $\pm$ 0.002
FOXA2	0.84 $\pm$ 0.01	0.94 $\pm$ 0.008	0.96 $\pm$ 0.006
HNF4A	0.81 $\pm$ 0.01	0.93 $\pm$ 0.008	0.98 $\pm$ 0.004
TBP	0.7 $\pm$ 0.01	0.89 $\pm$ 0.01	0.97 $\pm$ 0.005
TFAP2A	0.76 $\pm$ 0.01	0.96 $\pm$ 0.006	0.98 $\pm$ 0.005
YY1	0.87 $\pm$ 0.01	0.93 $\pm$ 0.008	0.99 $\pm$ 0.003
Average	0.8 $\pm$ 0.01	0.93 $\pm$ 0.008	0.98 $\pm$ 0.004

Next, Table 4.5 shows the mean $\pm$ SD of recall rates for LS-GKM, DeepBind, and GMFR-CNN together with their average mean $\pm$ SD of recall rates for all the ten datasets. In this table, it can be clearly seen that DeepBind has scored the lowest average recall rate as compared to LS-GKM and GMFR-CNN which is only 81%. It could be affected by TBP dataset where the recall rate is only 27%. On the other hand, LS-GKM has shown better performance for both FOXA2 and HNF4A in recall evaluation metric and acquired similar performance for CTCF, HNF4A, and TFAP2A. Meanwhile, DeepBind performed inconsistently for EP300 and TBP in recall metric. The length of both TFs could possibly contribute to the poor performance (Ahn et al., 2012).

Table 4.5: Average recall rates of LS-GKM, DeepBind, and GMFR-CNN.

TF	LS-GKM	DeepBind	GMFR-CNN
CEBPB	0.94 $\pm$ 0.007	0.97 $\pm$ 0.005	0.99 $\pm$ 0.005
CEBPD	0.9 $\pm$ 0.009	0.97 $\pm$ 0.005	0.99 $\pm$ 0.003
CTCF	0.99 $\pm$ 0.004	0.92 $\pm$ 0.009	0.99 $\pm$ 0.003
ELK1	0.94 $\pm$ 0.007	0.9 $\pm$ 0.009	0.97 $\pm$ 0.005
EP300	0.96 $\pm$ 0.006	0.49 $\pm$ 0.02	0.99 $\pm$ 0.002
FOXA2	0.99 $\pm$ 0.002	0.93 $\pm$ 0.008	0.97 $\pm$ 0.006
HNF4A	0.98 $\pm$ 0.004	0.9 $\pm$ 0.01	0.98 $\pm$ 0.004
TBP	0.71 $\pm$ 0.01	0.27 $\pm$ 0.01	0.97 $\pm$ 0.006
TFAP2A	0.97 $\pm$ 0.005	0.79 $\pm$ 0.01	0.97 $\pm$ 0.005
YY1	0.96 $\pm$ 0.006	0.96 $\pm$ 0.006	0.99 $\pm$ 0.003
Average	0.93 $\pm$ 0.006	0.81 $\pm$ 0.009	0.98 $\pm$ 0.004

In Table 4.6, the highest average mean $\pm$ SD of f -measure rate is scored by GMFR-CNN then followed by LS-GKM and DeepBind. This table has clearly shown that TBP has scored the lowest f -measure value for both LS-GKM and DeepBind with 70% and 41%. While for GMFR-CNN, FOXA2 has scored the lowest f -measure value among the datasets which is 96%. In this table, it can be seen that GMFR-CNN has scored 11% and 14% higher in f -measure value as compared to LS-GKM and DeepBind.

Table 4.6: Average f -measure rates of LS-GKM, DeepBind, and GMFR-CNN.

TF	LS-GKM	DeepBind	GMFR-CNN
CEBPB	0.91 ± 0.009	0.95 ± 0.006	0.99 ± 0.002
CEBPD	0.83 ± 0.01	0.95 ± 0.006	0.99 ± 0.002
CTCF	0.89 ± 0.01	0.93 ± 0.008	0.99 ± 0.003
ELK1	0.89 ± 0.01	0.9 ± 0.009	0.97 ± 0.005
EP300	0.87 ± 0.01	0.64 ± 0.02	0.99 ± 0.002
FOXA2	0.91 ± 0.009	0.93 ± 0.008	0.96 ± 0.006
HNF4A	0.89 ± 0.01	0.91 ± 0.009	0.98 ± 0.004
TBP	0.7 ± 0.01	0.41 ± 0.02	0.97 ± 0.006
TFAP2A	0.85 ± 0.01	0.86 ± 0.01	0.97 ± 0.005
YY1	0.91 ± 0.009	0.94 ± 0.007	0.99 ± 0.003
Average	0.87 ± 0.01	0.84 ± 0.01	0.98 ± 0.004

Moreover, the average mean $\pm$ SD of accuracy for each of the datasets performed by LS-GKM, DeepBind, and GMFR-CNN is depicted in Table 4.7. GMFR-CNN has scored the highest average accuracy value and it is followed by DeepBind and LS-GKM. The average accuracy scored by LS-GKM and DeepBind is rather near. The lowest accuracy obtained for both LS-GKM and DeepBind is TBP which is 70% and 62% and FOXA2 has obtained 96% in accuracy for GMFR-CNN.

Table 4.7: Average accuracy of LS-GKM, DeepBind, and GMFR-CNN.

TF	LS-GKM	DeepBind	GMFR-CNN
CEBPB	0.9 ± 0.01	0.95 ± 0.007	0.99 ± 0.002
CEBPD	0.81 ± 0.01	0.82 ± 0.01	0.99 ± 0.003
CTCF	0.88 ± 0.01	0.94 ± 0.008	0.99 ± 0.003
ELK1	0.88 ± 0.01	0.9 ± 0.009	0.97 ± 0.005
EP300	0.85 ± 0.01	0.72 ± 0.01	0.99 ± 0.002
FOXA2	0.9 ± 0.009	0.93 ± 0.008	0.96 ± 0.006
HNF4A	0.88 ± 0.01	0.92 ± 0.009	0.98 ± 0.004
TBP	0.7 ± 0.01	0.62 ± 0.02	0.97 ± 0.006
TFAP2A	0.83 ± 0.01	0.88 ± 0.01	0.97 ± 0.005
YY1	0.91 ± 0.009	0.94 ± 0.007	0.99 ± 0.003
Average	0.85 ± 0.01	0.86 ± 0.01	0.98 ± 0.004

According to the AUC table below, we can observe that TBP has scored the lowest AUC for LS-GKM and DeepBind while ELK1 has scored the lowest AUC for GMFR-CNN. The highest AUC scored by CEBPB in both LS-GKM and DeepBind while CEBPD and CTCF in GMFR-CNN. From Table 4.8, we can see that the average AUC of GMFR-CNN is 5.9% higher than DeepBind and 3% higher than LS-GKM. Also, the results have shown a significant difference on the AUC values generated by the 3 tools which depicted in Table 4.9 where the p -value is 0.04.

Table 4.8: Average AUC for LS-GKM, DeepBind, and GMFR-CNN.

TF	LS-GKM	DeepBind	GMFR-CNN
CEBPB	0.9969	0.9862	0.9998
CEBPD	0.9162	0.9081	0.9999
CTCF	0.9928	0.9831	0.9999
ELK1	0.958	0.9537	0.9919
EP300	0.9884	0.9037	0.9996
FOXA2	0.9954	0.9791	0.9969
HNF4A	0.9959	0.9759	0.9983
TBP	0.8678	0.745	0.9973
TFAP2A	0.987	0.9658	0.9975
YY1	0.9847	0.9842	0.9981
Average	0.9683	0.9385	0.9979

Table 4.9: One way ANOVA test of LS-GKM, DeepBind, and GMFR-CNN.

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.017666	2	0.008833	3.566745	0.042215	3.354131
Within Groups	0.066864	27	0.002476			
Total	0.084529	29				

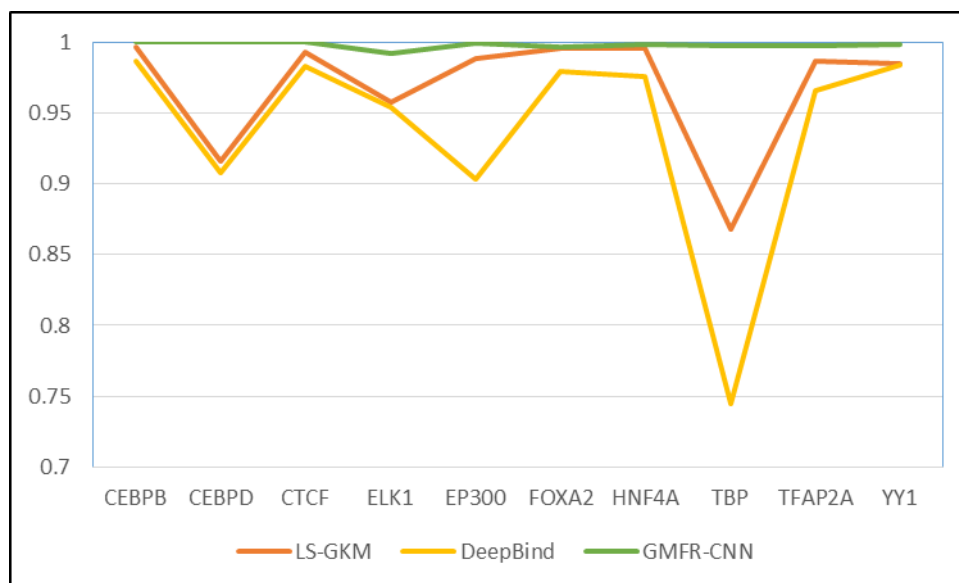


Figure 4.4: Summary graph for the ten datasets using LS-GKM, DeepBind, and GMFR-CNN.

The results for binary decision problems in machine learning are commonly represented by ROC curves. The classifier labels used in our study are either 1 or 0. The value 1 defines enhancer and the value 0 defines non-enhancer. A confusion matrix or contingency table is used to represent the decision made by the classifier. The confusion matrix has four categories: TP, FP, TN, and FN. Later on, the confusion matrix can be used to construct a point in ROC space. One plots the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) on the y-axis in the ROC space. The FPR used in our study measures the fraction of non-enhancer that are misclassified as an enhancer and the TPR measures the fraction of enhancer that is correctly labelled (Davis & Goadrich, 2006). Figure 4.5 until Figure 4.14 depicts the comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for the ten ChIP-seq datasets. The red line is used to represent GMFR-CNN, blue line for LS-GKM, and green line for DeepBind.

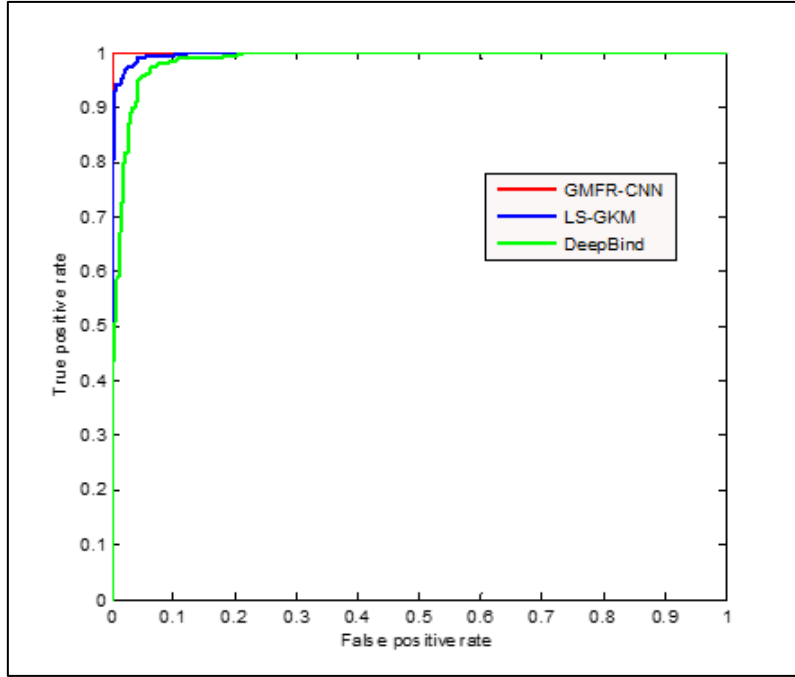


Figure 4.5: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CEBPB.

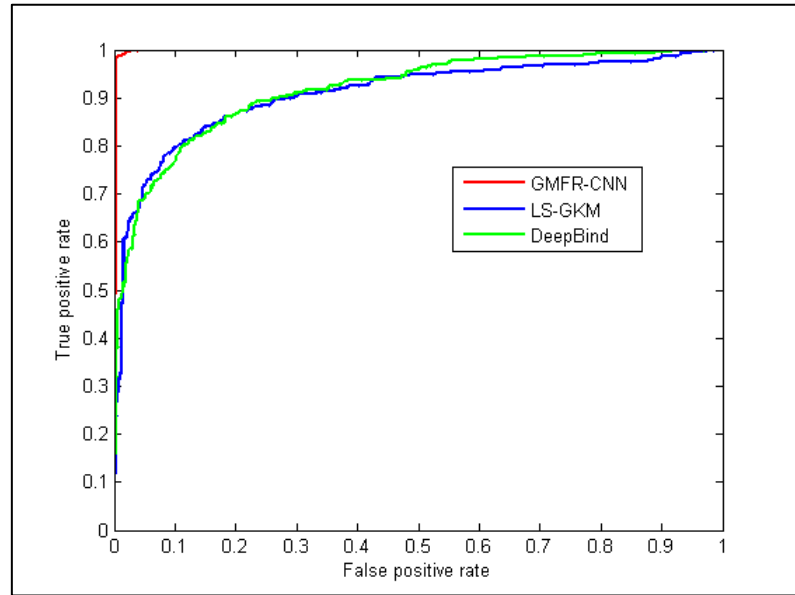


Figure 4.6: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CEBPD.

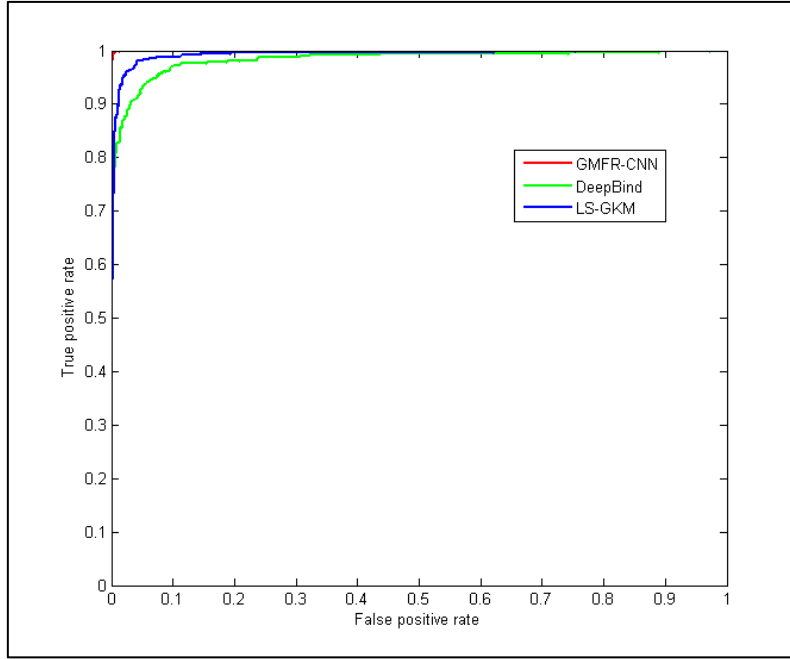


Figure 4.7: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for CTCF.

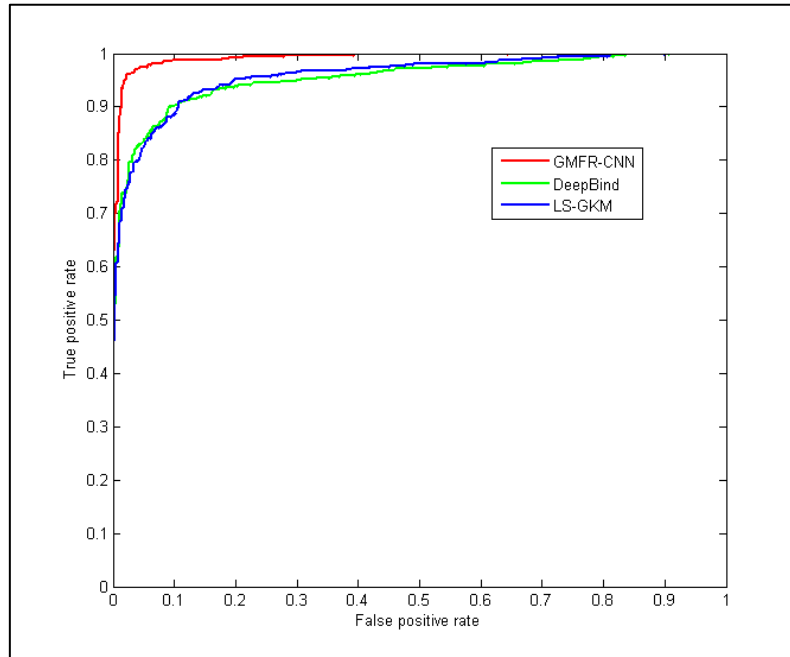


Figure 4.8: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for ELK1.

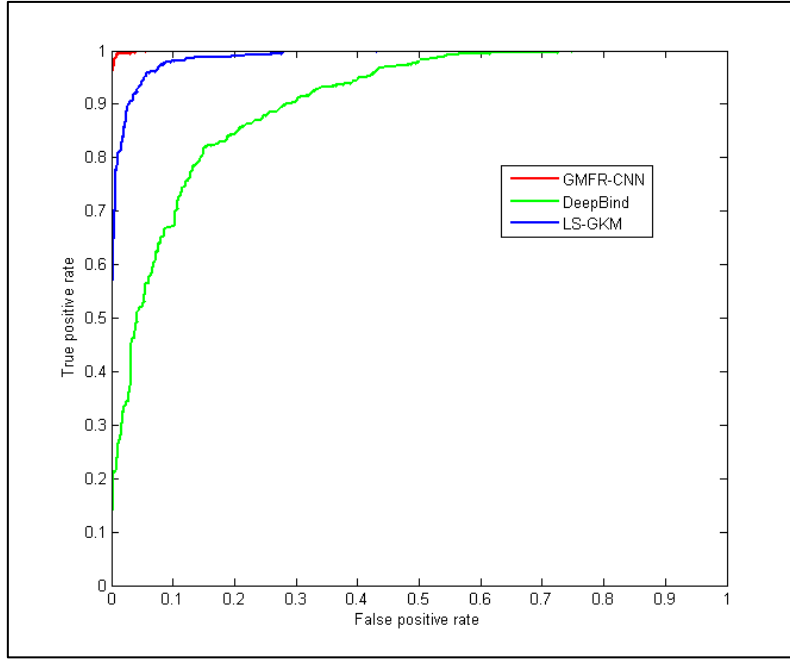


Figure 4.9: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for EP300.

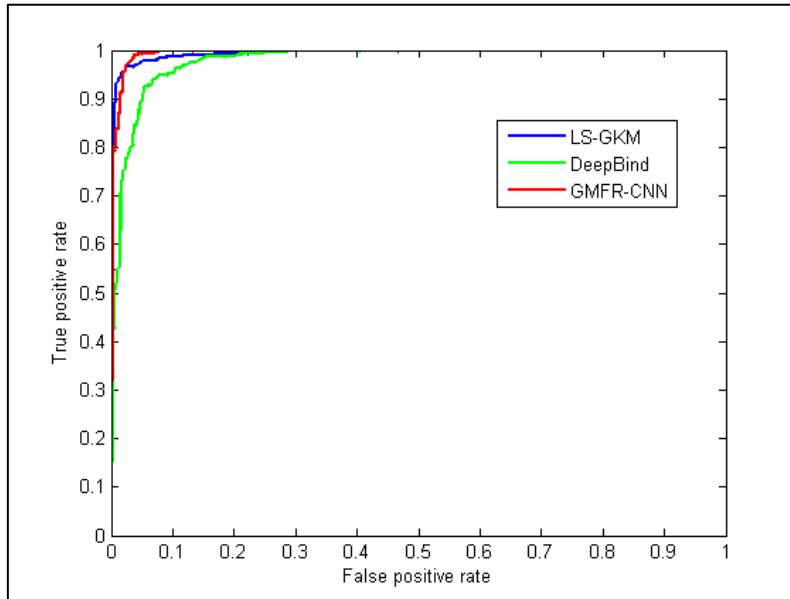


Figure 4.10: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for FOXA2.

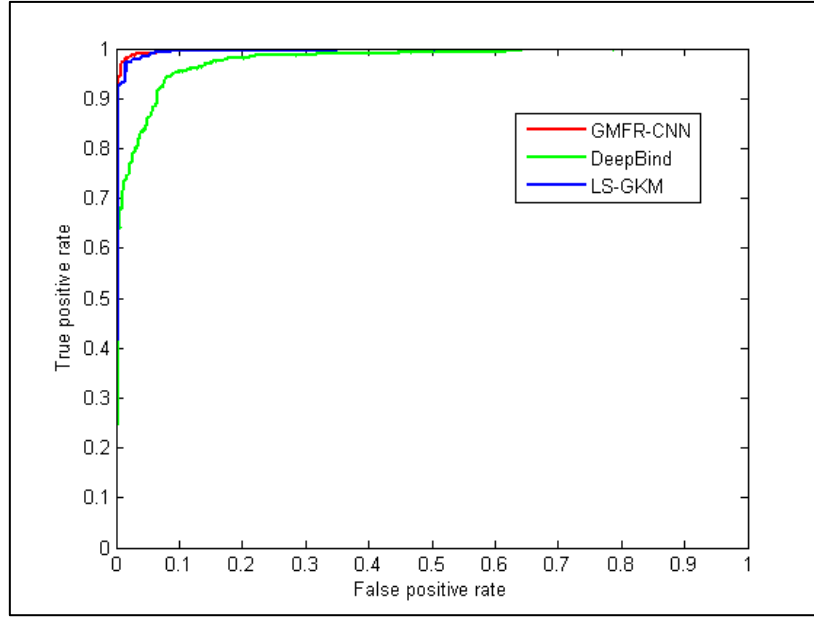


Figure 4.11: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for HNF4A.

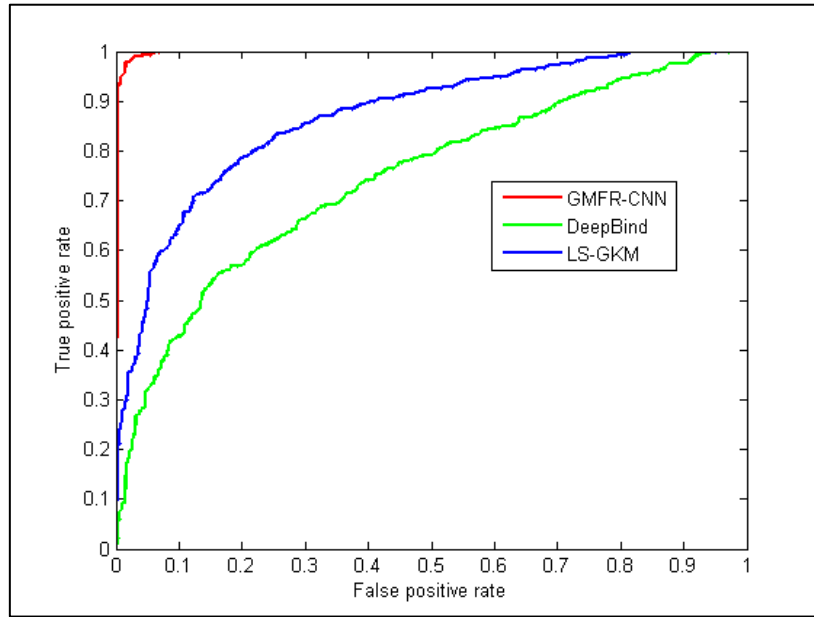


Figure 4.12: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for TBP.

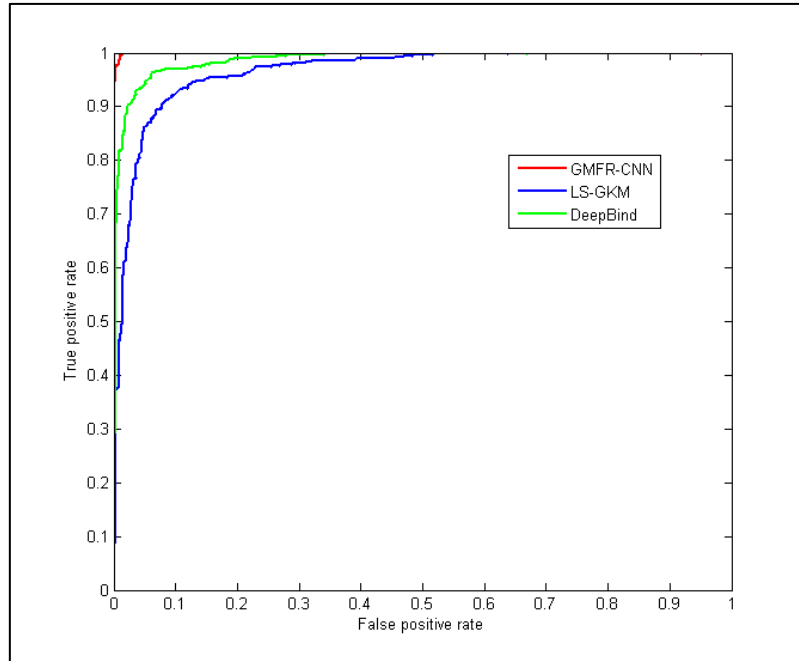


Figure 4.13: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for TFAP2A.

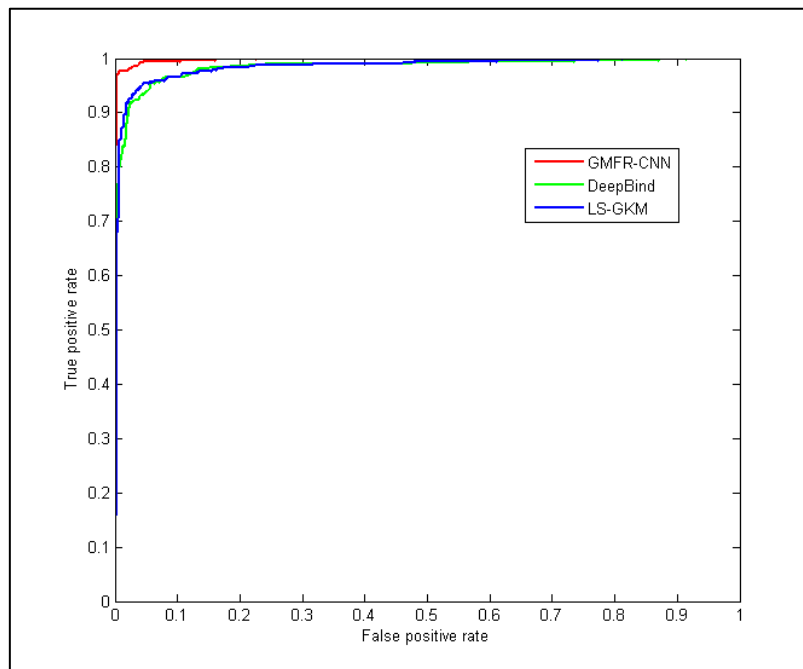


Figure 4.14: Comparison of GMFR-CNN, LS-GKM, and DeepBind using receiver operating characteristic (ROC) for YY1.

4.4 Discussion

Based on our findings from this study, the implementation of gapped motif approach in deep learning technique has proven to be beneficial to the task of enhancer prediction. Detailed comparisons of our proposed method with DeepBind and LS-GKM showed that our GMFR-CNN outperformed these two existing methods in terms of classification accuracy on the ten ChIP-seq datasets from DeepBind’s literature. We also introduced a novel method of representing the DNA sequence as an “image” where the gaps of the motif can be included during the convolution of CNN. In this way, the location of the motif has become less important where the motif feature is already attained by the neural network in deeper layers. The inclusion of gaps in the motif prediction task is crucial for longer transcription factor binding sites such as CTCF and thus cannot be completely represented by the short k -mers. This is because longer k -mers generate extremely sparse feature vectors where most of the k -mers do not appear in a training sequence and thus receive zero counts or appear only once. This will cause a severe overfitting problem even at a moderate size of k . GMFR-CNN is capable of overcoming this problem for long k -mers and hence significantly improves the classification accuracy, especially for long TFBSs.

Despite all the strengths possessed by GMFR-CNN, there are some limitations of this method which are worth noticing include (1) the inclusion of gaps is only applied internally, and (2) insertions and deletions within motifs are not allowed. Therefore, it is possible that those motifs with gaps which appear at the end on either one or both side of the motifs might be ignored. In addition to that, some sequence motifs that exhibit insertion and deletion might be ignored as well. A neural network that is able to identify inserted and deleted gapped motifs could improve the classification accuracy.

4.5 Conclusion

In this study, we have successfully built a model called GMFR-CNN which is an integration of a gapped motif feature representation (GMFR) and deep convolutional neural networks (CNNs) that acknowledge the significance of gapped features in enhancer prediction. This study also shows that the use of genomic sequence property is adequate for the purpose of our experiments. Fivefold cross-validation technique was implemented to assess our model using ten ChIP-seq datasets from DeepBind’s literature. Our model has shown excellent results which outperformed competitor tools such as LS-GKM and DeepBind for the task of enhancer identification. It is worth noticing that GMFR-CNN has accomplished near optimal f -measure and accuracy. We strongly believe that our model can be enhanced by garnering insights from academia and industry since deep learning has been an active research area that is heavily invested in academia and industry.

CHAPTER 5

GENES AND ENHANCER REGIONS ANNOTATION PROTOCOL FOR PROBOSCIS MONKEY GENOME

5.1 Introduction

In this study, we use GMFR-CNN to predict enhancer locations in the proboscis monkey (*Nasalis larvatus*) genome where the genome has been completely sequenced. A gene annotation pipeline called MAKER was used to perform the initial annotation of the genes. By using six eukaryotic model species, this pipeline has been successfully predicted 3,084 genes from chromosome 18 of the genome. A comparison between five tools include DeepBind (Alipanahi, Delong, Weirauch, & Frey, 2015), LS-GKM (Lee, 2016), GMFR-CNN (Wong, Lee, & Omar, 2016), CSI-ANN (Firpi, Ucar, & Tan, 2010), and iEnhancer-2L (Liu, Fang, Long, Lan, & Chou, 2016) on the enhancer prediction in the intergenic regions is described in discussions section later in this chapter. Enhancers are potentially resided within intergenic regions due to the epigenetic features possessed by these regions. GMFR-CNN has shown the highest coverage percentage as compared to the other four tools which is 79.04%. This value is generated by the coverage formula where the sum of a fraction of bases in each tool that had occurred in the other four tools.

The remaining chapter is structured as follows: the background is presented in Section 5.2 and followed by methods in Section 5.3 which include annotation using MAKER and intergenic regions for enhancer prediction. Next, analysis of coverage is shown in Section 5.4 and discussion is presented in Section 5.5. Lastly, we conclude this chapter in Section 5.6.

5.2 Background

Proboscis monkey or scientifically named as *nasalis larvatus* is endemic to the Borneo island (Meijaard & Nijman, 2000). This species has been viewed as the most primitive colobine and a long isolated genus of the group (Giusto & Margulis, 1981; Peng, Pan, & Jablonski, 1993) due to the exceptional diploid number, $2n = 48$ (Chiarelli, 1966; Soma, Benirschke, & Robinson, 1974; Stanyon, Camperio-Ciani, Sineo, & Morescalchi, 1992) which has played an important role in phylogenies. Substitutional polarity can be reliably determined by a primate that is close enough to humans and chimpanzees, however, for the purpose of defining the regions of functional sequence conservation, some evolutionary distance is also beneficial (Varki & Altheide, 2005). Old World monkeys and humans share a common ancestor. Therefore, the identification of the many evolutionarily, physiologically, and biomedically pivotal differences that distributed throughout these genomes is challenging. The identification involved is required to take into account the emerging knowledge about the corresponding “phenomes” and the respective environmental influences. Potential targets of opportunity can be simultaneously identified by the analysis of genome-wide which could help in the elimination of the most genomic sequence differences from consideration. Besides, genomic analysis can be used as fundamentals by candidate gene approaches on genes that may contribute to known differences in phenotypes or disease incidence or severity, or on mutations in the human population which impact unique aspects of the human condition (Varki & Altheide, 2005). The genome sequencing, assembly, and annotation of proboscis monkey genome enable the exploration of primate evolution and genetic contributions to human physiology and disease.

A brief description of the genome sequencing and assembly of proboscis monkey genome is described as follows while the annotation of proboscis monkey genome will be described in the next section. Genome sequencing is a procedure held to obtain a genome draft

that would represent the complete nucleotide base sequence or the order of nucleotides for all chromosomes in the species of interest (Ekblom & Wolf, 2014). Genome assembly is the process where the short sequence fragments or also known as reads are reconstructed to form the complete sequence of the original sequence (Ekblom & Wolf, 2014). This process is crucial because current genome sequencing technology cannot read genomes as a whole.

The proboscis monkey genome used in this study was sequenced by Malaysian Genomics Resource Centre Berhad (MGRC) in 2012. First, a shotgun sequencing strategy was used to perform genome sequencing and the genomic DNA is sheared into small random fragments (Batzoglou et al., 2002). These fragments are sequenced independently to a given length depends on the technology used. Then, the resulting sequence reads are joined back together into a longer continuous sequence or also known as contigs by a procedure called *de novo* assembly. During the assembly process, a high sequencing coverage is required because it indicates that the sequence reads at each position in the genome is sufficiently overlapped. As for longer fragments, those with several hundred base pairs, are sequenced from both ends to provide additional information on correct read placement in the assembly. This process is called paired-end sequencing. Next, contigs are joined back together to form longer stretches of the sequence which also known as scaffolds after the initial assembly. In order to accomplish this task, the libraries from long DNA fragments which span several kilobases (kb) of sequence in the genome are prepared and their endpoints are sequenced. These libraries are called paired-end, mate-pair, or jump libraries depending on the technology and the specifics of the library preparation. The several independent fragments are joined in a scaffold if their endpoint sequences come to lie on two different contigs. The information on the physical distance between the two contigs is provided by the expected fragment length of the library and the generated gap is filled with the uninformative base-pair character 'N'. In order to fill in the

missing base-pair information, subsequent gap closing methods can be implemented, ideally using long reads that read across repetitive sequences (Ekblom & Wolf, 2014). For more details on the field guide to whole-genome sequencing, assembly, and annotation, readers can refer to the work by Ekblom and Wolf (2014). Figure 5.1 described the genome sequencing and assembly pipeline of the proboscis monkey genome together with the results obtained by MGRC.

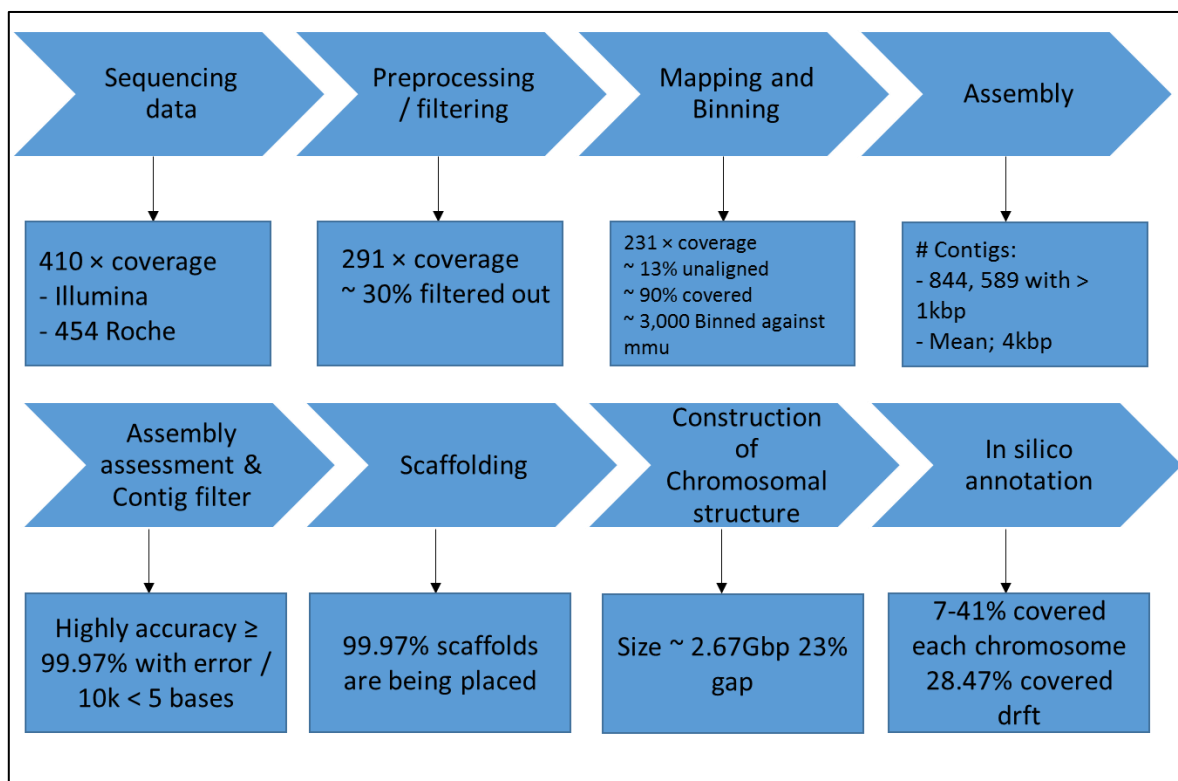


Figure 5.1: Genome sequencing and assembly pipeline of the proboscis monkey genome.

5.3 Methods

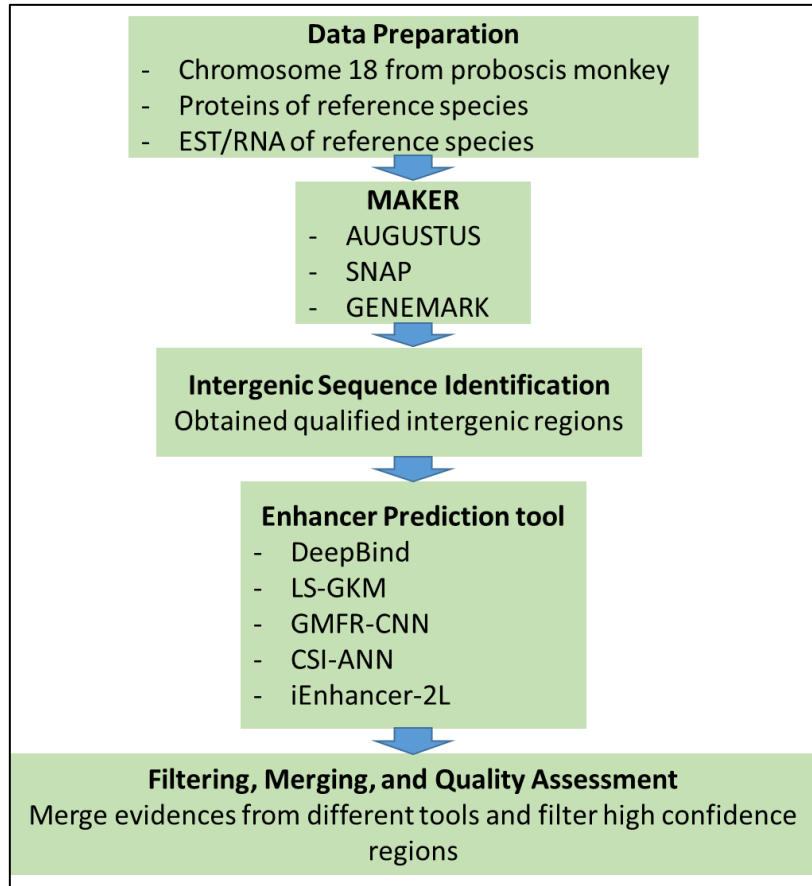


Figure 5.2: Enhancer annotation pipeline.

In brief, the enhancer annotation for Chromosome 18 in proboscis monkey consists of several steps as shown in Figure 5.2. We obtained chromosome 18 of proboscis monkey from NCBI GenBank under accession number GCA_000772465.1. The annotation pipeline used is MAKER to perform the genome annotation. The annotated protein and transcript sequences as reference data are collected from six different species which are *gorilla gorilla* (western gorilla) (Sclay et al., 2012), *macaca mulatta* (rhesus monkey) (Zimin et al., 2014), *mus musculus* (house mouse) (Church et al., 2011), *pan troglodytes* (chimpanzee) (Sequencing & Consortium., 2005), *homo sapiens* (human) (Lander et al., 2001), and *pongo abelii* (Sumatran orangutan) (Locke et al., 2011) (<ftp://ftp.ncbi.nlm.nih.gov/genomes/>). Other than that, three gene

prediction programs: SNAP, Augustus, and GeneMark were used. There are 3084 genes have been identified in Chromosome 18 of the proboscis monkey. The repetitive elements are masked from the Chromosome 18 during the annotation process, then followed by the gene prediction and the evidence protein and EST sequences from six species were aligned using BLAST. The *ab initio* gene models predicted by SNAP, Augustus, and GeneMark together with the aligned evidence through MAKER pipeline are further refined.

Next, the intergenic regions, which possibly contain enhancers are identified by the following extraction process. GFF-Ex (Gupta, 2011) was used to process the gff file generated from MAKER for the purpose of the intergenic region extraction. GFF-Ex is capable of extracting sequences based on numerous region boundaries such as exons and introns regions, gene regions, upstream regions and also the intergenic regions. GFF-Ex has successfully extracted 1,783 intergenic regions, 19,782 exons, and 15,769 introns from chromosome 18 of the proboscis monkey. Refinement of the regions was done by eliminating intergenic regions with length not more than 500bp and only take into account those regions that located 500bp away from transcription start site (TSS) and transcription end site (TES).

There are five computational tools including DeepBind, LS-GKM, GMFR-CNN, CSI-ANN, and iEnhancer-2L were used to predict enhancers in the intergenic regions. DeepBind, LS-GKM, and GMFR-CNN have been previously described in Chapter 4. CSI-ANN is a framework that comprised of a data transformation, a feature extraction, and time-delay neural network that is able to identify functional DNA elements by using chromatin signals (Firpi, Ucar, & Tan, 2010). Meanwhile, iEnhancer-2L is a support vector machine (SVM) predictor based on the feature in the pseudo k-tuple nucleotide composition (Liu, Fang, Long, Lan, & Chou, 2016). The transcription factor binding sequences of CTCF and sequences associated with co-factor EP300 were chosen for DeepBind, LS-GKM, and GMFR-CNN. Additional two

tools CSI-ANN and iEnhancer-2L which utilize histone datasets are introduced because not all enhancers are associated with CTCF and EP300. For CSI-ANN, H3K4me1 histone marks were used for enhancer prediction while for iEnhancer-2L, multiple histone marks including H3K4me1, H3K4me3, and H3K27ac were used for this purpose too.

In total, there are 1,783 intergenic sequences were successfully extracted from chromosome 18. These sequences were obtained via the previous annotation process and will be treated as input to DeepBind, LS-GKM, GMFR-CNN, CSI-ANN, and iEnhancer-2L for enhancer prediction. Besides, there are two challenges on predicting enhancers using different prediction tools because each tool may require different input features and the parameters for each tool may vary from one another.

There are two sets of predicted enhancers generated by DeepBind, LS-GKM, and GMFR-CNN due to two transcription factors used. Therefore, the two sets of predicted enhancers from each tool are combined into a single file and this file will be merged using BEDTools *merge* utility (Quinlan & Hall, 2010). This utility enables features that are overlapped by at least 1bp or next to the other base pair to be merged together. An example shown by BEDTools *merge* is depicted in Figure 5.3 (a). These merged regions obtained by all five tools are regions where the enhancers for chromosome 18 are likely to be found. In addition to that, BEDTools *intersect* utility (Quinlan & Hall, 2010) was used to obtain the overlapped features in the two, three, four, or five tools. Any common features between two or more set of genomic features can be discovered by BEDTools *intersect*. Figure 5.3 (b) illustrates the example of intersection using BEDTools *intersect*. Finally, the overlapping features by the combination of five tools can be merged together using BEDTools *merge*.

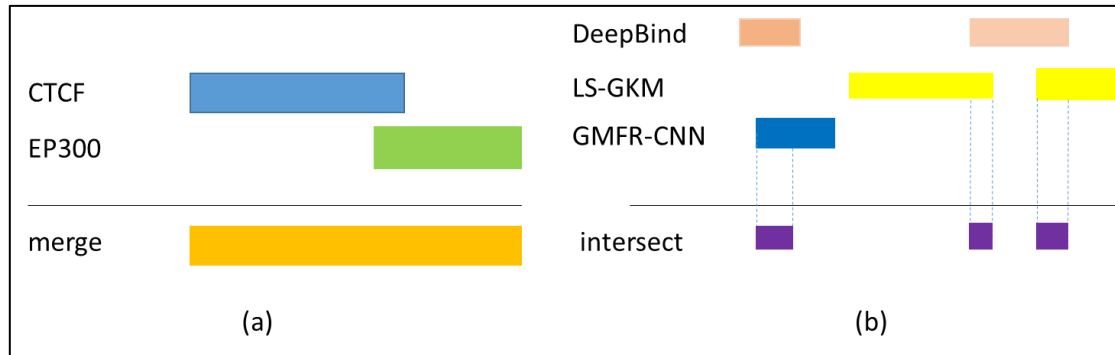


Figure 5.3: BEDTools *merge* and *intersect*. (a) Merging two sequences that overlapped by at least 1 bp or next to the other. (b) The intersection of DeepBind to LS-GKM and GMFR-CNN.

5.3.1 Annotation using MAKER

Elucidating the characteristics of the genome are known as annotation and through this method to further identify the genes reside in the genome. There are a few famous annotation databases such as Ensembl and RefSeq. The annotated data of proboscis monkey is not available in any of the databases so far. Thus, MAKER is considered in this case. Computational genome annotation of the genome acquires a reasonable amount of time to produce a high quality of annotation results, for example, MAKER is widely used to perform such task. In general, MAKER is a gene annotation pipeline that combines the results of multiple gene finding tools, along with EST and protein sequences, to yield a single consistent set of annotations that considers both predictions and evidence. MAKER consists of four steps: (a) repeat masking, (b) transcript and protein alignments, (c) initial *ab initio* gene prediction, and (d) refine *ab initio* gene prediction.

RepeatMasker is run on input sequences in the first step of MAKER to recognize and mask repeated regions of the genome. A significant amount of the genome could be made up of these repetitive sequences. The presence of repetitive elements could lead to poor quality of genome annotation. Next, RNA and protein alignment help to indicate whether a region is

associated with a gene by identifying how active the region is being transcribed or homologous to a known protein. Three *ab initio* predictors (SNAP, Augustus, GeneMark) are used to optimize Hidden Markov Model (HMM) by modifying its HMM using the information obtained from MAKER to produce preliminary gene models. After that, this task can be refined by running *ab initio* programs using hints yielded by predictor and MAKER merges them with evidence to produce a complete set of annotation.

5.3.2 Enhancer Prediction

The enhancer regions before prediction are obtained from intergenic regions with certain conditions as described in the following section. Chromosome 18 is chosen among the rest of the chromosomes for the purpose of this experiment because it is the most computationally feasible as in the size of the genomic data is manageable by the available sources. While the rest of the chromosomes will be put into consideration when the outcome has achieved positivity by wet-lab researchers. This specific region has to be determined based on previous literature to obtain high quality of distal enhancers. These enhancer regions will be further verified by selected enhancer prediction tools and then merge these regions together to yield enhancer segments.

5.3.3 Intergenic Regions for Enhancer Identification

A single chromosome normally consists of genes, intergenic regions, and unknown regions that are usually represented by alphabet N (e.g. NNNN). In order to predict enhancers, intergenic regions are determined by obtaining the complement intervals of predicted gene regions. Intergenic regions are usually located at more than 10 kilo base pairs away from the

closest transcription start site (TSS) or transcription end site while the upstream regions are within 10 kilo base pairs upstream of TSSs (Chen, Morris, & Mitchell, 2012). The intergenic enhancers are at least 3 kilo base from TSS of a gene and are often outside gene bodies (Zhu et al., 2013). In addition, the enhancer regions in intergenic regions also have been decided where only enhancer of its closest gene that is 100 kilo base from the TSS is considered. Moreover, enhancers are summarized as the peaks of the H3K4me1-enriched regions which are usually situated at least 3 kilo base away from TSS (Zhu et al., 2013). However, some of the works only considered enhancers that were between 2 to 100 kilo base from a TSS which associated to their respected genes (González-Vallinas, Pagès, Singh, & Eyra, 2015). Other than that, in order to distinguish between putative enhancers and promoters, only DHSs that are located in intergenic regions which are downstream of transcription terminal site and more than 1.5 kilo base upstream of TSS are considered as putative enhancers. In short, putative enhancers are those DHSs with at least 1.5 kilo base away from TSSs (Zhu, Zhang, Zhang, Liu, & Jiang, 2015).

The regions summarized as shown in Figure 5.4, 5.5, and 5.6 are based on different kinds of literature which focus only the enhancer prediction regions in between 3 kilo base pairs and 100 kilo base pairs from the TSSs. This study only focuses on upstream enhancers.

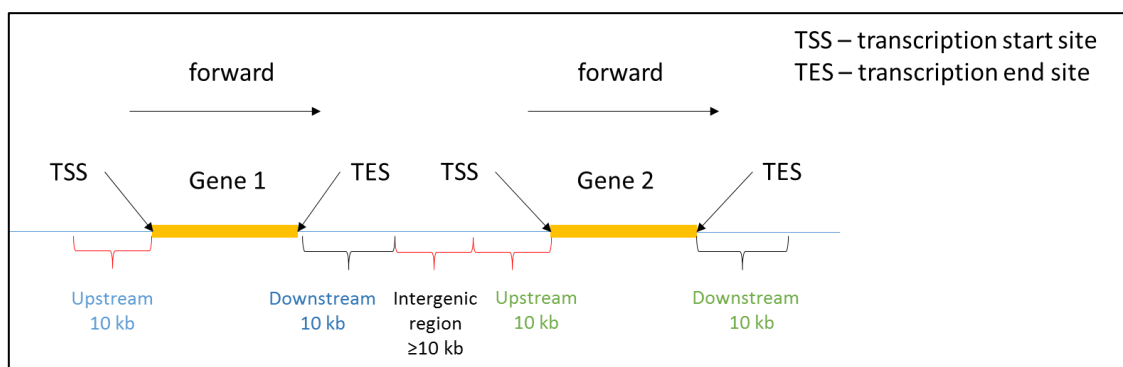


Figure 5.4: Both genes are in forward strand.

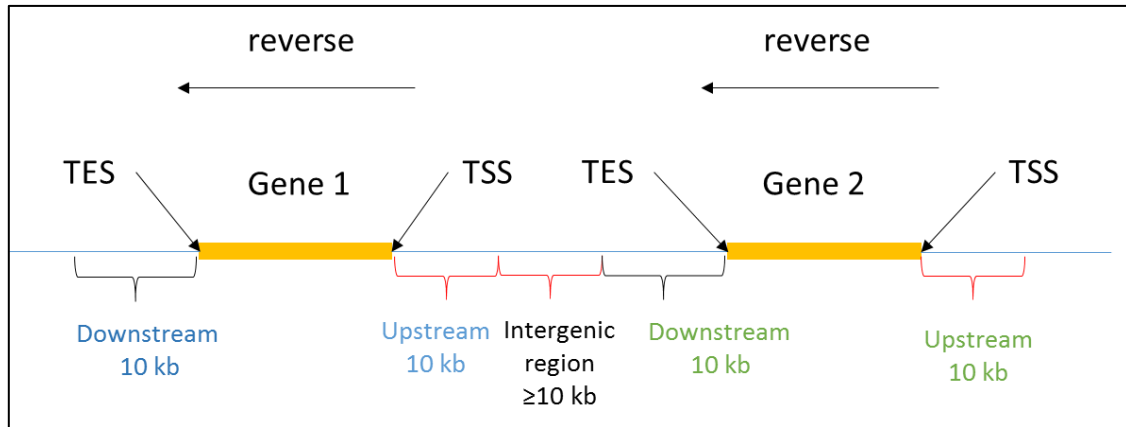


Figure 5.5: Both genes are in reverse strand.

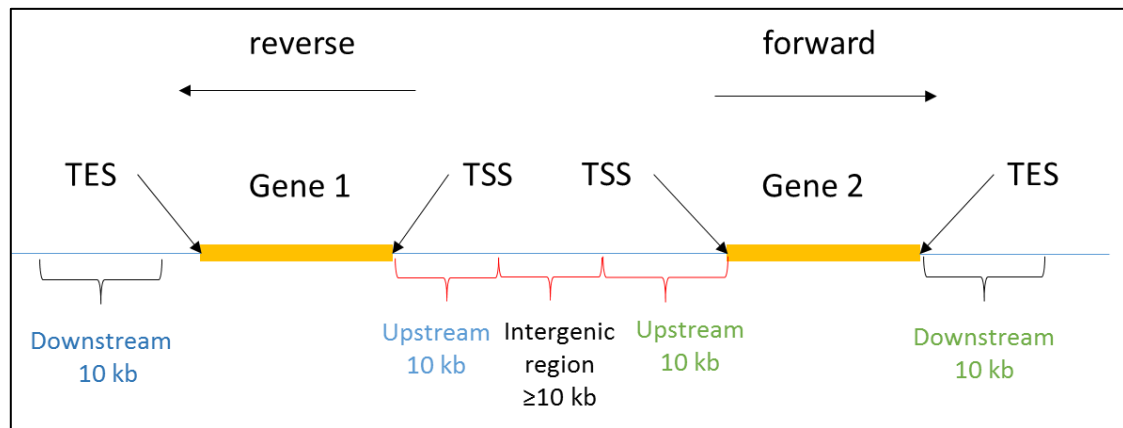


Figure 5.6: Both genes are in opposite directions where the gene on the left is in reverse strand and the gene on the right is in forward strand.

After determining the intergenic regions, five enhancer prediction tools were used to predict the enhancers. The five enhancer prediction tools are iEnhancer-2L, CSI-ANN, LS-GKM, DeepBind, and GMFR-CNN. The enhancers predicted from the five tools are selected with at least overlapped by 3 tools.

5.4 Results

Some tools may have fewer parameters that need to be tuned, but some may have numerous of it. The parameters used for each tool are summarized in Table 5.1. The number of predicted enhancers produced by each of the prediction tools is shown in Table 5.2. In Table 5.3, the number of overlapped and merged region as performed by two, three, four, and five tools are conferred.

Table 5.1: Parameters used by different prediction tools.

Tools	Parameters
DeepBind	We used trained model provided by DeepBind; Consists of input layer, convolutional layer, rectification layer, pooling layer, neural network prediction layer, and an output layer; Learning rate = 0.0005, 0.05; Learning momentum = 0.95, 0.99; Number of iterations = 4,000-20,000
LS-GKM	We trained the model using CTCF and EP300 datasets; $l = 14$ (CTCF), 9 (EP300); $k = 6$ (CTCF), 6 (EP300)
GMFR-CNN	Consists of 6 layers (input layer, 2 convolutional layers, 2 subsampling layers, and an output layer); We trained the model using CTCF and EP300 datasets; Learning rate = 0.8; Number of iterations = 200
CSI-ANN	Time delay neural network (TDNN) classifier with a delay of 9, 2 hidden layer nodes and an output layer; We trained the model using Chromosome 18 intergenic regions; w is trained using particle swarm optimization (PSO)
iEnhancer-2L	2-layer predictor with 2,968 trained samples obtained from final benchmark datasets; $k = 6$ (1 st and 2 nd layers); $w = 0.1$ (1 st layer), 0.4 (2 nd layer); $\lambda = 9$ (1 st and 2 nd layers)

Table 5.2: The number of predicted enhancers obtained from five prediction tools.

Tool	Number of Predicted Enhancers
DeepBind	31,805
LS-GKM	65,349
GMFR-CNN	73,133
CSI-ANN	22,954
iEnhancer-2L	14,700

Table 5.3: The number of overlapped and merged regions by a different number of enhancer prediction tools for Chromosome 18 of the proboscis monkey.

No. of enhancer prediction tools	No. of overlapped and merged regions
2	165544
3	500145
4	496632
5	3861

The predicted overlapped and merged regions by 3 tools have shown the highest number of enhancer regions while the second highest number of enhancer predicted regions can be seen in using 4 tools to produce overlapped and merged enhancer regions. This is followed by 2 enhancer prediction tools and lastly 5 enhancer prediction tool which has the lowest number of regions produced. This is possibly due to the low coverage of certain tool that has caused the number of overlapped regions to drop.

5.4.1 Analysis of Coverage

The analysis of coverage is carried out to compare between prediction tools on how wide a single prediction tool can cover the prediction regions as compared to the other four enhancer prediction tools. The software tool used to calculate the coverage value is called BEDTools (Quinlan & Hall, 2010) with the function named coverage. Coverage function is chosen because it can compute both the depth and breadth of coverage of features in file Y on the features in file X for example. The number of features that overlap an interval in file X is counted and the fraction of bases in the interval in X that was overlapped by one or more features is computed as well. In order to carry out the task, two BED (.bed) files are needed. A BED file is a tab-delimited text file that defines a feature track and for the computation of

coverage, BED file with only chromosome name, chromosome starting position, and chromosome ending position are needed which include the coordinates of a single comparison tool and the other four tools' merged coordinates. The other four tools' coordinates are merged because we want to inspect the coverage of single tool when compared to the other four tools. Figure 5.7 illustrates how the coverage percentage of a single tool DeepBind as compared to the other four tools is being calculated. Table 5.4 shows the coverage percentage accomplished by each of the prediction tools. Summation of coverage is the total value of coverage by each sequence returned by BEDTools *coverage*. The percentage of coverage is calculated as such where the total number of sequences is divided by the summation of coverage times 100.

Chromosome	Start Coor	End Coor	No. of overlapping occurrences	No. of overlapped bases	Total bases	Coverage of features value
chr18	10699637	10699853	1	19	216	0.087963
chr18	10702790	10702942	1	19	152	0.125
chr18	10703190	10703289	1	19	99	0.191919
chr18	10705428	10705527	0	0	99	0
chr18	10705628	10705807	1	148	179	0.826816
chr18	10707793	10707892	1	19	99	0.191919
$\text{Percentage of coverage} = \frac{\text{Sum of coverage values}}{\text{No. of sequences}} \times 100$ $= \frac{1.423617}{6} \times 100$ $= 23.73\%$						Sum = 1.423617

Figure 5.7: An example of the calculation of coverage percentage.

Table 5.4: A comparison of coverage between CSI-ANN, DeepBind, LS-GKM, GMFR-CNN, and iEnhancer-2L.

Tool	Summation of coverage	Total number of enhancer features	Percentage of Coverage (%)
CSI-ANN	3582.89	22954	15.61
DeepBind	12053.15	31805	37.90
LS-GKM	15055.69	65349	23.04
GMFR-CNN	57803.93	73133	79.04
iEnhancer-2L	4218.95	14700	28.70

From Table 5.4, it can be seen that GMFR-CNN has shown wider coverage as compared to CSI-ANN, DeepBind, LS-GKM, and iEnhancer-2L. This implies some of the enhancers that are predicted by other four tools also can be found in the prediction of GMFR-CNN. This has also proved that with genomic sequence features alone, GMFR-CNN is capable of predicting enhancer regions which cover 79.04% prediction by other four enhancer prediction tools. It is then followed by DeepBind which has covered 37.9% regions of the other four enhancer prediction tools. While iEnhancer-2L and LS-GKM both have obtained 28.7% and 23.04% in coverage. Lastly, CSI-ANN has only 15.61% in coverage as compared to the rest of the prediction tools.

5.5 Discussion

Based on the analysis of coverage presented in the previous section, it can be observed that GMFR-CNN indeed has predicted more enhancer features as compared to DeepBind, LS-GKM, CSI-ANN, and iEnhancer-2L. This is possibly due to the method implemented by GMFR-CNN where the gapped motif features are taken into account as part of enhancer prediction framework. Besides, the convolution and subsampling techniques employed in GMFR-CNN contributes to the location invariance of the enhancer features. From the result, DeepBind has scored the second highest coverage percentage in this study where it also implemented CNN in its framework, however, DeepBind did not utilize the benefit of gapped motif features. It is clearly shown that by using gapped motif features, the coverage percentage has been increased by half in GMFR-CNN. The third highest coverage percentage scored by iEnhancer-2L which consists of a two-layer predictor. This tool has benchmark data containing 742 strong enhancers, 742 weak enhancers, and 1,484 non-enhancers. In the first predictor layer,

a DNA sequence will be identified as enhancer or non-enhancer and if the result is yes, then the second predictor layer will be decided on its strength. Only 78 enhancers are belonged to chromosome 18 out of the total of benchmark data which is 2968 and only 10 are strong enhancers. Therefore, it is possible why iEnhancer-2L did not score high in coverage as compared to the other tools. However, this can be improved when the number of strong enhancer in benchmark data has been increased. As for LS-GKM, which employed gapped motif features and support vector machine (SVM) has scored the second-lowest coverage percentage in this study. The SVM used is only comprised of two shallow layers and motif features that formed enhancer could hardly attain. Unlike GMFR-CNN and DeepBind, both tools have at least one hidden layer to construct higher order feature from lower level ones. Lastly, CSI-ANN has scored the lowest coverage percentage among all the tools involved. The Fisher discriminant analysis which a feature reduction technique used in CSI-ANN is sensitive to outlier features. Therefore, those features will affect the ecological efficacy of the analysis. The time-delay neural network used in CSI-ANN is consists of one input layer, one hidden layer, and one output layer. The drawback of this neural network is rather long, computationally intensive, learning phase. From the analysis of coverage, the number of enhancer features that predicted by CSI-ANN and found in other four tools is rather small.

In this study, the predicted enhancers are not able to be verified as true positives by the wet-lab experimental procedure at the moment and the predicted enhancers produced by the five different tools are likely to be the real one. Therefore, there is some information such as the number of true positives, false positives, true negatives, and false negatives couldn't be provided for further evaluation practices. Other than that, there are other epigenetic features such as DNase I hypersensitivity, TAL1, and GATA1 that have not been included in this study can also be considered in the near future to improve the enhancer prediction for other

chromosomes of the proboscis monkey. The annotation process carried out in this study is costly and time-consuming. A machine with higher computational power is highly recommended for the tasks being done in this study.

5.6 Conclusion

In conclusion, GMFR-CNN has achieved wide coverage enhancer prediction region as compared to CSI-ANN, iEnhancer-2L, DeepBind, and LS-GKM. While DeepBind has shown nearly half the coverage enhancer prediction region predicted by GMFR-CNN. Both GMFR-CNN and DeepBind have accomplished higher coverage in enhancer prediction regions and the common practice that can be observed in both tools are deep learning technique. This also proves that deep learning technique can be used for new genomic data such as the proboscis monkey genome and performs prediction that is yet to be verified in wet-lab later on. Thus, in recent years, computational methods for predicting the enhancer regions have gained favour in the bioinformatics and biological sciences communities where the cost spent on wet-lab alone to perform gene annotation and enhancer verification can be greatly reduced.

CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 Contributions of Thesis

The contributions of the works from this thesis specifically on DeepFinder and GMFR-CNN are described below:

a) DeepFinder

- The improved three-stage approach in DeepFinder which uses a partial set of DNA sequences is adequately used as input to four motif discovery tools (MEME, BioProspector, MDscan, and MotifSampler) for enhancer prediction. The putative motif profiles obtained can be used to scan the remaining sequences not use in the partial set for potential binding sites. Thus, prominent motifs can be built from any subset of the input sequences of a reasonable size.
- In DeepFinder, a total of seventy-six features are employed which include A, C, G, and T occurrences, all possible 3-mer frequencies, selected 2-mer frequencies of CA, CG, GA, and GC as well as their respective probabilities of occurrence. These features play a crucial role in distinguishing between motifs and non-motifs. It is proven that the dependency features of TFBSs are included by these seventy-six features because no matter how many layers of the neural network used and if the representation of the input is not meaningful, the learning of the neural network will not yield a positive outcome.
- A method called similarity measure was employed together with *k*-medoid clustering technique to obtain the highest quality cluster which will be treated as input to deep learning model later on. The similarity measure examines how similar are the clusters

in the putative binding sites. These putative binding sites are returned by the four *de novo* motif discovery tools. The scores obtained are populated to a distant matrix where *k*-medoid clustering technique will select the highly similar putative binding sites and group them into the same cluster. The implementation of *k*-medoid clustering technique in DeepFinder allows high-quality input set to be selected. This method is robust to noise and outlier effects.

- The implementation of deep learning method called stacked autoencoder (SAE), as well as integrated feature-based approach have contributed to the major success in the improvement of the three-stage method. Previous studies by Wang and Lee (2009) have shown that three-stage method is potential in producing an excessively high number of false positives. The high number of false positive also caused by the inability of current motif model to represent the specificity of the binding sites. DeepFinder has indeed improved this condition by reducing the number of false positives significantly and consistently performed well for the ten datasets used.

b) GMFR-CNN

- A novel sequence encoding method called gapped motif feature representation (GMFR) was developed so that the strength of gapped motifs as an input feature for the learning of neural network can be fully utilized. This advantage allows the detection of binding sites that are longer than 6 to 8 bp, thus, overfitting can be avoided. Besides, motifs that are not identified in the continuous sequence motif approach can be discovered by GMFR during the learning of the neural network.
- A carefully designed of the convolutional neural network (CNN) which takes GMFR as a two-dimensional DNA “image” was developed in this study. The neural network

consists of an input layer, two convolution layers, two subsampling layers, and an output layer and consistently performed for the ten ChIP-seq datasets used with average 98% in accuracy.

- The use of GMFR-CNN in predicting enhancers for proboscis monkey genome also has shown high coverage percentage as compared to the other enhancer prediction tools such as DeepBind, LS-GKM, CSI-ANN, and iEnhancer-2L. It also means that GMFR-CNN is capable of predicting enhancers in the proboscis monkey genome where some of these enhancers are predicted by other prediction tools as well.

6.2 Summary

- A partial dataset is sufficiently used to build seed motifs that are able to scan for potential enhancers in the remaining dataset. These partial set of data will be treated as input to *de novo* motif discovery tools and the outcome from these motif discovery tools is a list of putative binding sites. According to the assumption made by heuristic three-stage method, binding sites that bound by transcription factor have limited variations. Therefore, the key protein-DNA codes should be sufficiently recognized by a partial set of DNA sequences. This theory is supported by Zia and Moses (2012) where a sufficient number of input sequences is sufficient and more will only marginally increase the predictive rates. In DeepFinder, we use partial data size of 20% from the whole ChIP-seq dataset as input to four *de novo* motif discovery tools. Another earlier empirical finding by Hu, Li, and Kihara (2005) stated that inputting more sequences does not improve the prediction accuracy. It is proven in our study that partial dataset can be sufficiently used to search for potential enhancers in the remaining dataset not used in the initial dataset.

- The DNA sequences that are represented by the designed 76 features are meaningful enough to capture the features as input in the neural network. This representation is crucial for feature discovery by the neural network later on. We employed dinucleotide repeat-like motifs as part of the 76 features because these motifs are proven to be enriched in regions bound by different regulatory protein (Li et al., 2008; Wang et al., 2012). Besides, the use of dependencies among adjacent bases that include some k -mer features is utilized as part of the 76 features where all possible 3-mer features are included. However, 4-mer combinations are not considered in our study because this feature will yield sparse representation. Never forget the inclusion of all 4 bases occurrences since it is originated from the most widely used PWM representation, even though it possessed independent features. We combined independent and dependent features for the representation of DNA sequences to avoid any important features that might be ignored if any of the features used are not taken into account.
- We find it effective to implement similarity measure and k -medoid clustering algorithm for grouping binding sites that are highly similar. During k -medoid clustering, it is important to keep the number of generated clusters small because when more clusters are being generated, those significant binding sites for enhancer prediction might be distributed to other clusters. Therefore, in this study, we have set the parameter in k -medoid clustering to generate at most two clusters. Other than that, it is utterly important to select a cluster with greater size because if the cluster size is small it may not contain enough information for building the representation using the designed 76 features.
- Readers also can consider of implementing stacked autoencoder (SAE) in feature discovery tasks. Based on our findings, SAE is not only good for feature discovery but also in recognizing features that are polluted by noise (Masci, Meier, Cireşan, &

Schmidhuber, 2011). This is beneficial to the task of enhancer prediction especially when the sequenced genome is filled with noise. Thus, noise filtering task would be required to discard the noise in order to produce a high resolution of the genome data.

- Enhancers are located either in upstream or downstream, therefore, motifs learned by the neural network model should be able to recognize these features regardless of the location of the enhancers. This is why convolutional neural network (CNN) is implemented together with our novel sequence encoding method called GMFR to search for the enhancers in the genome. Our proposed encoding method is able to simulate gapped motifs which are said to play a pivotal role in enhancer motif discovery. The inclusion of gaps is beneficial because this can avoid overfitting if the binding sites involved are longer than 6 to 8bp and also it allows the modelling of short internal loops which enable for a small number of mutations in the conserved region (Pervouchine, 2014). The use of GMFR-CNN for the prediction of enhancers in proboscis monkey genome has significantly outperformed other tools, for example, DeepBind and LS-GKM which employs either CNN or gapped motif features. An integration of both methods has shown astonishing performance in terms of f -measure and accuracy.

6.3 Limitations

- In the work of DeepFinder, the number of *de novo* motif discovery tools that can be sufficiently used to obtain most putative binding sites is not known. There are no single or multiple best motif discovery tools for the purpose of motif prediction because every single tool has their advantages and disadvantages. However, using all the motif discovery tools for DeepFinder is not feasible.

- Currently, there are no robust practical guidelines to obtain the sufficient amount of partial input for the three-stage method. This is a difficult task for researchers when they carry out the experiment, e.g., the percentage of data selected for motif discovery tools. The implication of selecting less than the sufficient amount of partial input will lead to the inadequate amount of features that actually can be used to predict an enhancer. From the experiments carried out in this studies, 20% out of the whole dataset seems to be sufficiently used as input to the *de novo* motif discovery tools for motif prediction. Five-fold cross-validation can be used to assess the quality of the randomly chosen partial input and thus, minimize the chances of getting insufficient features later on.
- Biophysical or biochemical properties of sequences are not taken into account in our work in DeepFinder even though these properties are said to play a critical role in determining the specificities of the sequences, especially for cis-acting elements (Shlyueva, Stampfel, & Stark, 2014). Besides, the conformational and physicochemical DNA features play an important role in binding site recognition (Ponomarenko et al., 1999). However, using genomic sequence features alone are sufficient to produce significant result throughout different datasets used. Thus, it is interesting to know the effects that would bring by the implication of biophysical or biochemical properties in predicting enhancers. This could possibly help in reducing the number of false positives.
- The gapped motif feature representation used in GMFR-CNN is only limited to gapped *k*-mer sequences. We use gapped motif features instead of continuous motif features because the approach will likely to face overfitting when the *k*-mer size involved is longer than 6bp (Fletez-Brant, Lee, McCallion, & Beer, 2013). Therefore, an integration of continuous and gapped motif features could possibly be used to further improve the enhancer prediction.

6.4 Future Works

In conclusion, the prediction of DNA motifs specifically on enhancers is an increasingly important topic in the field of bioinformatics and the improvements in this area have taken place in the past several years. The currently available motif discovery tools play a crucial role in helping experimental scientists to accelerate their findings on the functional characterization of enhancers. However, the overall performances of different motif prediction methods are all-time low and the tools could be more effective in practical use.

The following aspects can be addressed to improve the performance of DNA motif prediction. First, due to the large size of most DNA sequences, inputting whole genome sequences for motif prediction is challenging and time-consuming (Hu, Li, & Kihara, 2005). Second, the binding site information often depends on both the protein and DNA structure because of the protein and DNA complex. The current motif prediction tools do not consider conformational changes during the process of DNA-binding to the protein even though they apply machine learning methods that consider different characteristics of protein and DNA. These are few methods that can be employed for DNA-binding proteins which include molecular dynamics and ligand docking methods (Gherzi & Sanchez, 2009; Kauffman, Rangwala, & Karypis, 2008). Third, TFs are the most important DNA-binding proteins and its models are built by several studies to train classifiers using both DNA and protein information (Chu et al., 2009; Schröder et al., 2010; Cai et al., 2008; Qian, Lu, Liu, Cai, & Li, 2007). According to the different features possessed by each TF family, the specific classifier for each TF family could be developed directly which could eventually produce higher prediction accuracy (Qian, Cai, & Li, 2006). Fourth, many features are employed in the task of motif prediction. The combination of different feature information is beneficial for correct identification of DNA-binding residues, however, the high-dimensional space can lead to an

overfitting problem (Zhu, Yang, Song, Chou, & Shen, 2010). In order to improve the effectiveness of current motif prediction, we strongly believe that effective feature selection and novel feature mining should be further developed. Through these efforts, we foresee that the overall performance for the prediction of transcription factor binding sites could be improved significantly.

REFERENCES

- Aggarwal, G., Worthey, E. A., McDonagh, P. D., & Myler, P. J. (2003). Importing statistical measures into Artemis enhances gene identification in the Leishmania genome project. *BMC Bioinformatics*, 4(1), 23.
- Ahn, S., Huang, C., Ozkumur, E., Zhang, X., Chinnala, J., Yalcin, A., Bandyopadhyay, S., Russek, S.J., Ünlü, M.S., DeLisi, C., & Irani, R.J. (2012). TATA binding proteins can recognize nontraditional DNA sequences. *Biophysical Journal*, 103(7), 1510-1517.
- Albrecht, M., Tosatto, S. C., Lengauer, T., & Valle, G. (2003). Simple consensus procedures are effective and sufficient in secondary structure prediction. *Protein Engineering*, 16(7), 459-462.
- Alipanahi, B., Delong, A., Weirauch, M. T., & Frey, B. J. (2015). Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. *Nature Biotechnology*, 33(8), 831-838.
- Amaral, T., Silva, L. M., Alexandre, L., Kandaswamy, C., Santos, J. M., & de Sá, J. M. (2013). Using different cost functions to train stacked auto-encoders. *Artificial Intelligence*, 12, 114-120.
- Angermueller, C., Lee, H. J., Reik, W., & Stegle, O. (2017). DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning. *Genome Biology*, 18(1), 67.
- Ao, W., Gaudet, J., Kent, W. J., Muttumu, S., & Mango, S. E. (2004). Environmentally induced foregut remodeling by PHA-4/FoxA and DAF-12/NHR. *Science*, 305(5691), 1743-1746.
- Bailey, T. L. (2008). Discovering sequence motifs. *Comparative Genomics*, 1, 271-292.
- Bailey, T. L., & Elkan, C. (1995). Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, 21(1), 51-80.
- Bailey, T. L., & Gribskov, M. (1998). Combining evidence using p-values: application to sequence homology searches. *Bioinformatics*, 14(1), 48-54.

- Bailey, T. L., & Noble, W. S. (2003). Searching for statistically significant regulatory modules. *Bioinformatics*, 19(suppl 2), ii16-ii25.
- Bailey, T., Krajewski, P., Ladunga, I., Lefebvre, C., Li, Q., Liu, T., & Zhang, J. (2013). Practical guidelines for the comprehensive analysis of ChIP-seq data. *PLoS Computational Biology*, 9(11), e1003326.
- Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods*, 43(1), 3-31.
- Batzoglou, S., Jaffe, D., Stanley, K., Butler, J., Gnerre, S., Mauceli, E., Berger, B., Mesirov, J.P., & Lander, E.S. (2002). ARACHNE: a whole-genome shotgun assembler. *Genome Research*, 12(1), 177-189.
- Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 1-127.
- Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798-1828.
- Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, 19, 153.
- Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, 57 (1), 289-300.
- Berman, B. P., Pfeiffer, B. D., Laverty, T. R., Salzberg, S. L., Rubin, G. M., Eisen, M. B., & Celniker, S. E. (2004). Computational identification of developmental enhancers: conservation and function of transcription factor binding-site clusters in *Drosophila melanogaster* and *Drosophila pseudoobscura*. *Genome Biology*, 5(9), 61.

- Boeva, V. (2016). Analysis of genomic sequence motifs for deciphering transcription factor binding and transcriptional regulation in eukaryotic cells. *Frontiers in Genetics*, 7, 24.
- Boureau, Y. L., & Cun, Y. L. (2008). Sparse feature learning for deep belief networks. *Advances in Neural Information Processing Systems*, 8, 1185-1192.
- Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7), 1145-1159.
- Brazma, A., Jonassen, I., Eidhammer, I., & Gilbert, D. (1998). Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2), 279-305.
- Broin, P. Ó., Smith, T. J., & Golden, A. A. (2015). Alignment-free clustering of transcription factor binding motifs using a genetic-k-medoids approach. *BMC Bioinformatics*, 16(1), 22.
- Bryne, J., Valen, E., Tang, M., Marstrand, T., Winther, O., da Piedade, I., Krogh, A., Lenhard, B., & Sandelin, A. (2008). JASPAR, the open access database of transcription factor-binding profiles: new content and tools in the 2008 update. *Nucleic Acids Research*, 36(suppl 1), D102-D106.
- Bryson, A. E., & Ho, Y. C. (1975). *Applied optimal control: optimization, estimation, and control*. CRC Press. United States of America.
- Bucher, P. (1990). Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212(4), 563-578.
- Bulyk, M. L. (2006). Protein binding microarrays for the characterization of DNA–protein interactions. In *Analytics of Protein–DNA Interactions* (1st ed.), (pp. 65-85). Springer Berlin Heidelberg.
- Cai, Y., He, J., Li, X., Lu, L., Yang, X., Feng, K., Lu, W., & Kong, X. (2008). A novel computational approach to predict transcription factor DNA binding preference. *Journal of Proteome Research*, 8(2), 999-1003.

- Carlson, J. M., Chakravarty, A., & Gross, R. H. (2006). BEAM: a beam search algorithm for the identification of cis-regulatory elements in groups of genes. *Journal of Computational Biology*, 13(3), 686-701.
- Carlson, J. M., Chakravarty, A., DeZiel, C. E., & Gross, R. H. (2007). SCOPE: a web server for practical de novo motif discovery. *Nucleic Acids Research*, 35(suppl 2), W259-W264.
- Carlson, J. M., Chakravarty, A., Khetani, R. S., & Gross, R. H. (2006). Bounded search for de novo identification of degenerate cis-regulatory elements. *BMC Bioinformatics*, 7(1), 254.
- Chakravarty, A., Carlson, J. M., Khetani, R. S., DeZiel, C. E., & Gross, R. H. (2007). SPACER: identification of cis-regulatory elements with non-contiguous critical residues. *Bioinformatics*, 23(8), 1029-1031.
- Chakravarty, A., Carlson, J., Khetani, R., & Gross, R. (2007). A novel ensemble learning method for de novo computational identification of DNA binding sites. *BMC Bioinformatics*, 8(1), 249.
- Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3), 27.
- Chen, C. Y., Morris, Q., & Mitchell, J. A. (2012). Enhancer identification in mouse embryonic stem cells using integrative modeling of chromatin and genomic features. *BMC Genomics*, 13, 1-19.
- Chiarelli, B. (1966). The chromosome complement of *Nasalis larvatus* (Wurm 1781). *Cellular and Molecular Life Sciences*, 22(12), 797.
- Chu, W. Y., Huang, Y. F., Huang, C. C., Cheng, Y. S., Huang, C. K., & Oyang, Y. J. (2009). ProteDNA: a sequence-based predictor of sequence-specific DNA-binding residues in transcription factors. *Nucleic Acids Research*, 37(2), 396-401.

- Church, D., Schneider, V., Graves, T., Auger, K., Cunningham, F., Bouk, N., Chen, H.C., Agarwala, R., McLaren, W.M., Ritchie, G.R., & Albracht, D. (2011). Modernizing reference genome assemblies. *PLoS Biology*, 9(7), e1001091.
- Cireşan, D. C., Giusti, A., Gambardella, L. M., & Schmidhuber, J. (2013). Mitosis detection in breast cancer histology images with deep neural networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013* (pp. 411-418). Springer Berlin Heidelberg.
- Cohen, J. (2004). Bioinformatics – an introduction for computer scientists. *ACM Computing Surveys (CSUR)*, 36(2), 122-158.
- Das, M. K., & Dai, H. K. (2007). A survey of DNA motif finding algorithms. *BMC Bioinformatics*, 8, S21.
- Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240). ACM.
- de Hoon, M. J., Imoto, S., Nolan, J., & Miyano, S. (2004). Open source clustering software. *Bioinformatics*, 20(9), 1453-1454.
- Defrance, M., Sand, O., & Van Helden, J. (2008). Using RSAT oligo-analysis and dyad-analysis tools to discover regulatory signals in nucleic sequences. *Nature Protocols*, 3(10), 1589-1603.
- Deng, L., & Yu, D. (2014). Deep learning: methods and applications. Foundations and Trends® in Signal Processing. *Foundations and Trends in Signal Processing*, 7(3-4), 197-387.
- D'haeseleer, P. (2006). What are DNA sequence motifs? *Nature Biotechnology*, 24(4), 423-425.
- Eggeling, R., Gohr, A., Keilwagen, J., Mohr, M., Posch, S., Smith, A. D., & Grosse, I. (2014). On the Value of Intra-Motif Dependencies of Human Insulator Protein CTCF. *PLoS One*, 9(1), e85629.

- Ekblom, R., & Wolf, J. B. (2014). A field guide to whole-genome sequencing, assembly and annotation. *Evolutionary Applications*, 7(9), 1026-1042.
- El-Sayed, M. A., Estaitia, Y. A., & Khafagy, M. A. (2013). Automated Edge Detection Using Convolutional Neural Network. *International Journal of Advanced Computer Science and Applications*, 4(10), 11-17.
- Eskin, E., & Pevzner, P. A. (2002). Finding composite regulatory patterns in DNA sequences. *Bioinformatics*, 18(suppl 1), S354-S363.
- Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639), 115-118.
- Ezziane, Z. (2006). Applications of artificial intelligence in bioinformatics: A review. *Expert Systems with Applications*, 30(1), 2-10.
- Fan, R. E., Chen, P. H., & Lin, C. J. (2005). Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6, 1889-1918.
- Favorov, A. V., Gelfand, M. S., Gerasimova, A. V., Mironov, A. A., & Makeev, V. J. (2004). A Gibbs sampler for identification of symmetrically structured, spaced DNA motifs with improved estimation of the signal length and its validation on the ArcA binding sites. *Bioinformatics*, 21(10), 2240-2245.
- Fernández, M., & Miranda-Saavedra, D. (2012). Genome-wide enhancer prediction from epigenetic signatures using genetic algorithm-optimized support vector machines. *Nucleic Acids Research*, 40(10), e77.
- Firpi, H. A., Ucar, D., & Tan, K. (2010). Discover regulatory DNA elements using chromatin signatures and artificial neural network. *Bioinformatics*, 26(13), 1579-1586.

- Fischer, D. (2003). 3D-SHOTGUN: a novel, cooperative, fold-recognition meta-predictor. *Proteins: Structure, Function, and Bioinformatics*, 51(3), 434-441.
- Fletez-Brant, C., Lee, D., McCallion, A. S., & Beer, M. A. (2013). kmer-SVM: a web server for identifying predictive regulatory sequence features in genomic data sets. *Nucleic Acids Research*, 41(W1), W544-W556.
- Frith, M. C., Hansen, U., Spouge, J. L., & Weng, Z. (2004). Finding functional sequence elements by multiple local alignment. *Nucleic Acids Research*, 32(1), 189-200.
- Fu, W., Ray, P., & Xing, E. P. (2009). DISCOVER: a feature-based discriminative method for motif search in complex genomes. *Bioinformatics*, 25(12), i321-i329.
- Gawehn, E., Hiss, J. A., & Schneider, G. (2015). Deep Learning in Drug Discovery. *Molecular Informatics*, 35(1), 3-14.
- Gherzi, D., & Sanchez, R. (2009). Improving accuracy and efficiency of blind protein-ligand docking by focusing on predicted binding sites. *Proteins: Structure, Function, and Bioinformatics*, 74(2), 417-424.
- Ghandi, M., Lee, D., Mohammad-Noori, M., & Beer, M. A. (2014). Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS Computational Biology*, 10(7), e1003711.
- Giardine, B., Riemer, C., Hardison, R. C., Burhans, R., Elnitski, L., Shah, P., & Nekrutenko, A. (2005). Galaxy: a platform for interactive large-scale genome analysis. *Genome Research*, 15(10), 1451-1455.
- Gibney, E. (2016). Google AI algorithm masters ancient game of Go. *Nature*, 529, 445-446.
- Giusto, J. P., & Margulis, L. (1981). Karyotypic fission theory and the evolution of old world monkeys and apes. *BioSystems*, 13(4), 267-302.

- Goldberg, Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57, 345-420.
- González-Vallinas, J., Pagès, A., Singh, B., & Eyraas, E. (2015). A semi-supervised approach uncovers thousands of intragenic enhancers differentially activated in human cells. *BMC Genomics*, 16, 1-14.
- Gordon, D. B., Nekludova, L., McCallum, S., & Fraenkel, E. (2005). TAMO: a flexible, object-oriented framework for analyzing transcriptional regulation using DNA-sequence motifs. *Bioinformatics*, 21(14), 3164-3165.
- Gotea, V., Visel, A., Westland, J. M., Nobrega, M. A., Pennacchio, L. A., & Ovcharenko, I. (2010). Homotypic clusters of transcription factor binding sites are a key component of human promoters and enhancers. *Genome Research*, 20(5), 565-577.
- Grant, C. E., Bailey, T. L., & Noble, W. S. (2011). FIMO: scanning for occurrences of a given motif. *Bioinformatics*, 27(7), 1017-1018.
- Griffiths, A. J., Miller, J. H., Suzuki, D. T., Lewontin, R. C., & Gelbart, W. M. (2000). *An Introduction to Genetic Analysis*, WH Freeman and Company, New York.
- Guigo, R. (2003). *An Introduction to Position Specific Scoring Matrices*. Retrieved from <http://bioinformatica.upf.edu/T13/MakeProfile.html>.
- Gülçehre, Ç., & Bengio, Y. (2016). Knowledge matters: Importance of prior information for optimization. *Journal of Machine Learning Research*, 17(8), 1-32.
- Guo, H., & Huo, H. (2017). A new algorithm for identifying cis-regulatory-modules based on Hidden Markov Model. *BioMed Research International*, 2017, 1-12.
- Gupta, D. (2011). GFF-Ex: A genome feature extraction package. *Journal of Natural Science, Biology and Medicine*, 2(3), 90.

- Hannenhalli, S., & Wang, L. S. (2005). Enhanced position weight matrices using mixture models. *Bioinformatics*, 21(1), 204-212.
- Hall, N., Karras, M., Raine, J., Carlton, J., Kooij, T., Berriman, M., Florens, L., Janssen, C.S., Pain, A., Christophides, G.K., & James, K. (2005). A comprehensive survey of the Plasmodium life cycle by genomic, transcriptomic, and proteomic analyses. *Science*, 307(5706), 82-86.
- Haykin, S. S. (2009). *Neural networks and learning machines* (Volume 3). Upper Saddle River, NJ, USA: Pearson.
- He, B., Chen, C., Teng, L., & Tan, K. (2014). Global view of enhancer-promoter interactome in human cells. In *Proceedings of the National Academy of Sciences*, 111(21), E2191-E2199.
- Hecht-Nielsen, R. (1987). Counterpropagation networks. *Applied Optics*, 26(23), 4979-4984.
- Heger, A., Lappe, M., & Holm, L. (2004). Accurate detection of very sparse sequence motifs. *Journal of Computational Biology*, 11(5), 843-857.
- Heintzman, N. D., & Ren, B. (2007). The gateway to transcription: identifying, characterizing and understanding promoters in the eukaryotic genome. *Cellular and Molecular Life Sciences*, 64(4), 386-400.
- Helmstaedter, M., Briggman, K. L., Turaga, S. C., Jain, V., Seung, H. S., & Denk, W. (2013). Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500(7461), 168-174.
- Hertz, G. Z., & Stormo, G. D. (1996). *Escherichia coli* promoter sequences: analysis and prediction. *Methods in Enzymology*, 273, 30-42.
- Hertz, G. Z., & Stormo, G. D. (1999). Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, 15(7), 563-577.

- Hertz, G. Z., Hartzell, G. W., & Stormo, G. D. (1990). Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Computer Applications in the Biosciences*, 6(2), 81-92.
- Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504-507.
- Hu, Z., Hu, B., & Collins, J. F. (2007). Prediction of synergistic transcription factors by function conservation. *Genome Biology*, 8(12), 257.
- Hu, J., Li, B., & Kihara, D. (2005). Limitations and potentials of current motif discovery algorithms. *Nucleic Acids Research*, 33(15), 4899-4913.
- Hu, J., Yang, Y. D., & Kihara, D. (2006). EMD: an ensemble algorithm for discovering regulatory motifs in DNA sequences. *BMC Bioinformatics*, 7(1), 342.
- Huang, G. B. (2003). Learning capability and storage capacity of two-hidden-layer feedforward networks. *IEEE Transactions on Neural Networks*, 14(2), 274-281.
- Huang, C. W., Lee, W. S., & Hsieh, S. Y. (2011). An improved heuristic algorithm for finding motif signals in DNA sequences. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 8(4), 959-975.
- Jayaram, N., Usvyat, D., & Martin, A. C. (2016). Evaluating tools for transcription factor binding site prediction. *BMC Bioinformatics*, 17, 1-12.
- Jia, C., Carson, M. B., & Yu, J. (2013). A fast weak motif-finding algorithm based on community detection in graphs. *BMC Bioinformatics*, 14(1), 227.
- Jiao, Y., & Du, P. (2016). Performance measures in evaluating machine learning based bioinformatics predictors for classifications. *Quantitative Biology*, 4(4), 320-330.

- Jin, F., Li, Y., Ren, B., & Natarajan, R. (2011). Enhancers: multi-dimensional signal integrators. *Transcription*, 2(5), 226-230.
- Johansson, Ö., Alkema, W., Wasserman, W. W., & Lagergren, J. (2003). Identification of functional clusters of transcription factor binding motifs in genome sequences: the MSCAN algorithm. *Bioinformatics*, 19(suppl 1), i169-i176.
- Johnson, A. D. (2010). An extended IUPAC nomenclature code for polymorphic nucleic acids. *Bioinformatics*, 26(10), 1386-1389.
- Jones, N. (2014). The learning machines. *Nature*, 146-148.
- Jothi, R., Cuddapah, S., Barski, A., Cui, K., & Zhao, K. (2008). Genome-wide identification of in vivo protein-DNA binding sites from ChIP-Seq data. *Nucleic Acids Research*, 36(16), 5221-5231.
- Juszczak, P., Tax, D., & Duin, R. P. (2002). Feature scaling in support vector data description. In *Proceedings of Advanced School for Computing and Image (ASCI)* (pp. 95-102). Citeseer.
- Kauffman, C., Rangwala, H., & Karypis, G. (2008). Improving homology models for protein-ligand binding sites. In *Proceedings of the 2008 LSS Computer System Bioinformatics Conference* (pp. 211-222). World Scientific Publishing Company.
- Kelley, D. R., Snoek, J., & Rinn, J. (2016). Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Research*, 26(7), 990-999.
- Kim, S. G., Theera-Ampornpunt, N., Fang, C. H., Harwani, M., Grama, A., & Chaterji, S. (2016). Opening up the blackbox: an interpretable deep neural network-based classifier for cell-type specific enhancer predictions. *BMC Systems Biology*, 10(2), 54.
- Kleftogiannis, D., Kalnis, P., & Bajic, V. B. (2015). DEEP: a general computational framework for predicting enhancers. *Nucleic Acids Research*, 43(1), e6.

- Kleftogiannis, D., Kalnis, P., & Bajic, V. B. (2015). Progress and challenges in bioinformatics approaches for enhancer identification. *Briefings in Bioinformatics*, 17(6), 967-979.
- Klepper, K., & Drabløs, F. (2013). MotifLab: a tools and data integration workbench for motif discovery and regulatory sequence analysis. *BMC Bioinformatics*, 14(1), 9.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1, 1097-1105.
- Kulakovskiy, I., Boeva, V., Favorov, A., & Makeev, V. (2010). Deep and wide digging for binding motifs in ChIP-Seq data. *Bioinformatics*, 26(20), 2622-2623.
- Lander, E., Linton, L., Birren, B., Nusbaum, C., Zody, M., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., & Funke, R. (2001). Initial sequencing and analysis of the human genome. *Nature*, 409(6822), 860-921.
- Landt, S., Marinov, G., Kundaje, A., Kheradpour, P., Pauli, F., Batzoglou, S., Bernstein, B.E., Bickel, P., Brown, J.B., Cayting, P., & Chen, Y. (2012). ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia. *Genome Research*, 22(9), 1813-1831.
- Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., & Wootton, J. C. (1993). Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262, 208-214.
- Lawrence, C. E., & Reilly, A. A. (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins: Structure, Function, and Bioinformatics*, 7(1), 41-51.
- Le, Q. V. (2013). Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP)* (pp. 8595-8598). IEEE.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.

- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE* (pp. 2278-2324). IEEE.
- Lee, D. (2016). LS-GKM: a new gkm-SVM for large-scale datasets. *Bioinformatics*, 32(14), 2196-2198.
- Lee, D., Karchin, R., & Beer, M. A. (2011). Discriminative prediction of mammalian enhancers from DNA sequence. *Genome Research*, 21(12), 2167-2180.
- Leung, G., & Eisen, M. B. (2009). Identifying cis-regulatory sequences by word profile similarity. *PloS One*, 4(9), e6901.
- Leung, K. S., Wong, K. C., Chan, T. M., Wong, M. H., Lee, K. H., Lau, C. K., & Tsui, S. K. (2010). Discovering protein-DNA binding sequence patterns using association rule mining. *Nucleic Acids Research*, 38(19), 6324-6337.
- Leung, M. K., Xiong, H. Y., Lee, L. J., & Frey, B. J. (2014). Deep learning of the tissue-regulated splicing code. *Bioinformatics*, 30(12), 121-129.
- Li, X. Y., MacArthur, S., Bourgon, R., Nix, D., Pollard, D. A., Iyer, V. N., Hechmer, A., Simirenko, L., Stapleton, M., Hendriks, C. L. L., & Chu, H.C. (2008). Transcription factors bind thousands of active and inactive regions in the *Drosophila* blastoderm. *PloS Biology*, 6(2), e27.
- Li, Y., Chen, C. Y., & Wasserman, W. W. (2016). Deep feature selection: theory and application to identify enhancers and promoters. *Journal of Computational Biology*, 23(5), 322-336.
- Ling, C. X., Huang, J., & Zhang, H. (2003). AUC: a better measure than accuracy in comparing learning algorithms. In *Conference of the Canadian Society for Computational Studies of Intelligence* (pp. 329-341). Springer Berlin Heidelberg.
- Liu, B., Fang, L., Long, R., Lan, X., & Chou, K. C. (2016). iEnhancer-2L: a two-layer predictor for identifying enhancers and their strength by pseudo k-tuple nucleotide composition. *Bioinformatics*, 32(3), 362-369.

- Liu, F., Li, H., Ren, C., Bo, X., & Shu, W. (2016). PEDLA: predicting enhancers with a deep learning-based algorithmic framework. *Scientific Reports*, 6, 28517.
- Liu, H., Han, F., Zhou, H., Yan, X., & Kosik, K. S. (2016). Fast motif discovery in short sequences. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference* (pp. 1158-1169). IEEE.
- Liu, X. S., Brutlag, D. L., & Liu, J. S. (2002). An algorithm for finding protein–DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nature Biotechnology*, 20(8), 835-839.
- Liu, X., Brutlag, D. L., & Liu, J. S. (2001). BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Biocomputing*, 127-138.
- Locke, D., Hillier, L., Warren, W., Worley, K., Nazareth, L., Muzny, D.M., Yang, S.P., Wang, Z., Chinwalla, A.T., Minx, P., & Mitreva, M. (2011). Comparative and demographic analysis of orang-utan genomes. *Nature*, 469(7331), 529-533.
- Lovén, J., Hoke, H. A., Lin, C. Y., Lau, A., Orlando, D. A., Vakoc, C. R., Bradner, J. E., Lee, T. I., & Young, R. A. (2013). Selective inhibition of tumor oncogenes by disruption of super-enhancers. *Cell*, 153(2), 320-334.
- Lu, X., Tsao, Y., Matsuda, S., & Hori, C. (2013). Speech enhancement based on deep denoising autoencoder. *Interspeech*, 436-440.
- Lu, Y., Qu, W., Shan, G., & Zhang, C. (2015). DELTA: a distal enhancer locating tool based on AdaBoost algorithm and shape features of chromatin modifications. *PloS One*, 10(6), e0130622.
- Lukoševičius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3), 127-149.
- Lusk, R. W., & Eisen, M. B. (2010). Evolutionary mirages: selection on binding site composition creates the illusion of conserved grammars in *Drosophila* enhancers. *PLoS Genetics*, 6(1), e1000829.

- Ma, J., Sheridan, R. P., Liaw, A., Dahl, G. E., & Svetnik, V. (2015). Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling*, 55(2), 263-274.
- Machanick, P., & Bailey, T. (2011). MEME-ChIP: motif analysis of large DNA datasets. *Bioinformatics*, 27(12), 1696-1697.
- Maier, H. R., & Dandy, G. C. (1996). The use of artificial neural networks for the prediction of water quality parameters. *Water Resources Research*, 32(4), 1013-1022.
- Makalo, A. (2016). A comparative analysis of motif discovery algorithms. *Computational Biology and Bioinformatics*, 4(1), 1-9.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.
- Mansour, M. R., Abraham, B. J., Anders, L., Berezovskaya, A., Gutierrez, A., Durbin, A. D., Etchin, J., Lawton, L., Sallan, S. E., Silverman, L. B., & Loh, M. L. (2014). An oncogenic super-enhancer formed through somatic mutation of a noncoding intergenic element. *Science*, 346(6215), 1373-1377.
- Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *Artificial Neural Networks and Machine Learning–ICANN 2011* (pp. 52-59). Springer Berlin Heidelberg.
- Maston, G. A., Evans, S. K., & Green, M. R. (2006). Transcriptional regulatory elements in the human genome. *Annual Review of Genomics and Human Genetics*, 7, 29-59.
- Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2), 442-451.

- Matys, V., Fricke, E., Geffers, R., Gößling, E., Haubrock, M., Hehl, R., Hornischer, K., Karas, D., Kel, A.E., Kel-Margoulis, O.V., & Kloos, D.U. (2003). TRANSFAC®: transcriptional regulation, from patterns to profiles. *Nucleic Acids Research*, 31(1), 374-378.
- Meijaard, E., & Nijman, V. (2000). Distribution and conservation of the proboscis monkey (*Nasalis larvatus*) in Kalimantan, Indonesia. *Biological Conservation*, 92(1), 15-24.
- Mellor, J. (2005). The dynamics of chromatin remodeling at promoters. *Molecular Cell*, 19(2), 147-157.
- Mewes, H., Frishman, D., Güldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Münsterkötter, M., Rudd, S., & Weil, B. (2002). MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 30(1), 31-34.
- Miah, M. B. A., Yousuf, M. A., Mia, M. S., & Miya, M. P. (2015). Handwritten courtesy amount and signature recognition on bank cheque using neural network. *International Journal of Computer Applications*, 118(5), 21-26.
- Michael, M., Nicolas, F., & Ukkonen, E. (2010). On the complexity of finding gapped motifs. *Journal of Discrete Algorithms*, 8(2), 131-142.
- Min, S., Lee, B., & Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5), 851-869.
- Munteanu, A., Ohler, U., & Gordân, R. (2014). COUGER—co-factors associated with uniquely-bound genomic regions. *Nucleic Acids Research*, 42(W1), W461-W467.
- Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1.
- Narang, A. S., & Mahato, R. I. (2010). *Targeted delivery of small and macromolecular drugs*. United States of America: CRC Press.

- Osada, R., Zaslavsky, E., & Singh, M. (2004). Comparative analysis of methods for representing and searching for transcription factor binding sites. *Bioinformatics*, 20(18), 3516-3525.
- Palm, R. B. (2012). Prediction as a candidate for learning deep hierarchical models of data. *Technical University of Denmark*, 5.
- Park, Y., & Kellis, M. (2015). Deep learning for regulatory genomics. *Nature Biotechnology*, 33(8), 825-826.
- Pavesi, G., Mereghetti, P., Mauri, G., & Pesole, G. (2004). Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic Acids Research*, 32(suppl 2), W199-W203.
- Peng, Y. Z., Pan, R. L., & Jablonski, N. G. (1993). Classification and evolution of Asian colobines. *Folia Primatologica*, 60(1-2), 106-117.
- Pervouchine, D. D. (2014). IRBIS: a systematic search for conserved complementarity. *RNA*, 20(10), 1519-1531.
- Petersen, M., Brodersen, P., Naested, H., Andreasson, E., Lindhart, U., Johansen, B., Nielsen, H.B., Lacy, M., Austin, M.J., Parker, J.E., & Sharma, S.B. (2000). Arabidopsis MAP kinase 4 negatively regulates systemic acquired resistance. *Cell*, 103(7), 1111-1120.
- Pinello, L., Bosco, G., & Yuan, G. (2013). Applications of alignment-free methods in epigenomics. *Briefings in Bioinformatics*, 15(3), 419-430.
- Pisanti, N., Crochemore, M., Grossi, R., & Sagot, M. (2005). Bases of Motifs for Generating Repeated Patterns with Wild Cards. *IEEE/ACM Transaction on Computational Biology and Bioinformatics*, 2(1), 40-50.

- Ponomarenko, J. V., Ponomarenko, M. P., Frolov, A. S., Vorobyev, D. G., Overton, G. C., & Kolchanov, N. A. (1999). Conformational and physicochemical DNA features specific for transcription factor binding sites. *Bioinformatics*, 15(7), 654-668.
- Powers, D. M. W. (2011). Evaluation: from precision, recall, and f -measure to ROC, informedness, markedness, and correlation. *Journal of Machine Learning Technologies*, 2(1), 37-63.
- Qi, Y., Wang, Y., Zheng, X., & Wu, Z. (2014). Robust feature learning by stacked autoencoder with maximum correntropy criterion. In *Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6716-6720). IEEE.
- Qian, Z., Cai, Y. D., & Li, Y. (2006). A novel computational method to predict transcription factor DNA binding preference. *Biochemical and Biophysical Research Communications*, 348(3), 1034-1037.
- Qian, Z., Lu, L., Liu, X., Cai, Y. D., & Li, Y. (2007). An approach to predict transcription factor DNA binding site specificity based upon gene and transcription factor functional categorization. *Bioinformatics*, 23(18), 2449-2454.
- Quang, D., & Xie, X. (2016). DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *Nucleic Acids Research*, 44(11), e107.
- Quinlan, A. R., & Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6), 841-842.
- Rajagopal, N., Xie, W., Li, Y., Wagner, U., Wang, W., Stamatoyannopoulos, J., Ernst, J., Kellis, M., & Ren, B. (2013). RFECs: a random-forest based algorithm for enhancer identification from chromatin state. *PloS Computational Biology*, 9(3), e1002968.
- Régnier, M., & Denise, A. (2004). Rare events and conditional events on random strings. *Discrete Mathematics & Theoretical Computer Science*, 6(2), 191-214.

- Rombauts, S., Déhais, P., Van Montagu, M., & Rouzé, P. (1999). PlantCARE, a plant cis-acting regulatory element database. *Nucleic Acids Research*, 27(1), 295-296.
- Roth, F. P., Hughes, J. D., Estep, P. W., & Church, G. M. (1998). Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotechnology*, 16(10), 939-945.
- Sagot, M. (1998). Spelling approximate repeated or common motifs using a suffix tree. *LATIN'98: Theoretical Informatics*, 374-390.
- Saini, H. K., & Fischer, D. (2005). Meta-DP: domain prediction meta-server. *Bioinformatics*, 21(12), 2917-2920.
- Salakhutdinov, R., & Hinton, G. E. (2009). Deep Boltzmann Machines. *Artificial Intelligence and Statistics*, 1, 448-455.
- Salgado, H., Gama-Castro, S., Martínez-Antonio, A., Díaz-Peredo, E., Sánchez-Solano, F., Peralta-Gil, M., Garcia-Alonso, D., Jiménez-Jacinto, V., Santos-Zavaleta, A., Bonavides-Martínez, C., & Collado-Vides, J. (2004). RegulonDB (version 4.0): transcriptional regulation, operon organization and growth conditions in Escherichia coli K-12. *Nucleic Acids Research*, 32(suppl 1), D303-D306.
- Sandve, G. K., & Drablos, F. (2006). A survey of motif discovery methods in an integrated framework. *Biology Direct*, 1(11), 11.
- Scally, A., Dutheil, J., Hillier, L., Jordan, G., Goodhead, I., Herrero, J., Hobolth, A., Lappalainen, T., Mailund, T., Marques-Bonet, T., & McCarthy, S. (2012). Insights into hominid evolution from the gorilla genome sequence. *Nature*, 483(7388), 169-175.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85-117.

- Schneider, T. D., & Stephens, R. M. (1990). Sequence logos: a new way to display consensus sequences. *Nucleic Acids Research*, 18(20), 6097-6100.
- Schröder, A., Eichner, J., Supper, J., Eichner, J., Wanke, D., Hennekes, C., & Zell, A. (2010). Predicting DNA-binding specificities of eukaryotic transcription factors. *PloS One*, 5(11), e13876.
- Sequencing, T. C., & Consortium., A. (2005). Initial sequence of the chimpanzee genome and comparison with the human genome. *Nature*, 437(7055), 69-87.
- Shlyueva, D., Stampfel, G., & Stark, A. (2014). Transcriptional enhancers: from properties to genome-wide predictions. *Nature Reviews Genetics*, 15(4), 272-286.
- Sinha, S., & Tompa, M. (2003). YMF: a program for discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic Acids Research*, 31(13), 3586-3588.
- Sinha, S., Van Nimwegen, E., & Siggia, E. D. (2003). A probabilistic method to detect regulatory modules. *Bioinformatics*, 19(suppl 1), i292-i301.
- Soma, H., Benirschke, K., & Robinson, P. (1974). The chromosomes of the proboscis monkey (*Nasalis larvatus*). *Chromosome Information Service*, 17, 24.
- Stanyon, R., Camperio-Ciani, A., Sineo, L., & Morescalchi, M. A. (1992). The G-banded chromosomes of the proboscis monkey (*Nasalis larvatus*) compared with the macaque (*Macaca mulatta*). *Antropologia Contemporanea*, 15, 101-104.
- Swaminathan M. B. M., Roy, K., Sarkar, S., Madasu, V. R., Rao, S. V., & Bala, R. (2015). A Deep Learning Approach for Categorizing Risk Impact in Software Domain. In *Proceedings of the 2015 Seventh International Conference on Computational Intelligence, Modelling and Simulation* (pp. 61-65). IEEE.
- Stormo, G. D. (2000). DNA binding sites: representation and discovery. *Bioinformatics*, 16(1), 16-23.

- Su, J., Teichmann, S. A., & Down, T. A. (2010). Assessing computational methods of cis-regulatory module prediction. *PLoS Computational Biology*, 6(12), e1001020.
- Sun, C., Huo, H., Yu, Q., Guo, H., & Sun, Z. (2015). An affinity propagation-based DNA motif discovery algorithm. *BioMedical Research International*, 2015, 1-9.
- Sun, H., Yuan, Y., Wu, Y., Liu, H., Liu, J. S., & Xie, H. (2010). Tmod: toolbox of motif discovery. *Bioinformatics*, 26(3), 405-407.
- Sun, J. (2012). Learning algorithm and hidden node selection scheme for local coupled feedforward neural network classifier. *Neurocomputing*, 79, 158-163.
- Swets, J. A. (1988). Measuring the accuracy of diagnostic systems. *Science*, 240(4857), 1285.
- Tan, A. C., & Gilbert, D. (2003). An empirical comparison of supervised machine learning techniques in bioinformatics. In *Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics* (pp. 219-222). Computer Australian Society, Inc.
- Thijs, G., Lescot, M., Marchal, K., Rombauts, S., De Moor, B., Rouze, P., & Moreau, Y. (2001). A higher-order background model improves the detection of promoter regulatory elements by Gibbs sampling. *Bioinformatics*, 17(12), 1113-1122.
- Thijs, G., Marchal, K., Lescot, M., Rombauts, S., De Moor, B., Rouzé, P., & Moreau, Y. (2002). A Gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *Journal of Computational Biology*, 9(2), 447-464.
- Thomas-Chollier, M., Darbo, E., Herrmann, C., Defrance, M., Thieffry, D., & Van Helden, J. (2012). A complete workflow for the analysis of full-size ChIP-seq (and similar) data sets using peak-motifs. *Nature Protocols*, 7(8), 1551-1568.

- Thomas-Chollier, M., Herrmann, C., Defrance, M., Sand, O., Thieffry, D., & van Helden, J. (2012). RSAT peak-motifs: motif analysis in full-size ChIP-seq datasets. *Nucleic Acids Research*, 40(4), 31.
- Tjian, R., & Maniatis, T. (1994). Transcriptional activation: a complex puzzle with few easy pieces. *Cell*, 77(1), 5-8.
- Todeschini, A. L., Georges, A., & Veitia, R. A. (2014). Transcription factors: specific DNA binding and specific gene regulation. *Trends in Genetics*, 30(6), 211-219.
- Tompa, M., Li, N., Bailey, T. L., Church, G. M., De Moor, B., Eskin, E., & Zhu, Z. (2005). Assessing computational tools for the discovery of transcription factor binding sites. *Nature Biotechnology*, 23(1), 137-144.
- Tran, N. T. L., & Huang, C. H. (2014). A survey of motif finding web tools for detecting binding site motifs in ChIP-Seq data. *Biology Direct*, 9(1), 4.
- Välimäki, N., Gerlach, W., Dixit, K., & Mäkinen, V. (2007). Compressed suffix tree—a basis for genome-scale sequence analysis. *Bioinformatics*, 23(5), 629-630.
- van Duijvenboden, K., de Boer, B., Capon, N., Ruijter, J., & Christoffels, V. (2016). EMERGE: a flexible modelling framework to predict genomic regulatory elements from genomic signatures. *Nucleic Acids Research*, 44(5), 42.
- van Heeringen, S., & Veenstra, G. (2011). GimmeMotifs: a de novo motif prediction pipeline for ChIP-sequencing experiments. *Bioinformatics*, 27(2), 270-271.
- van Helden, J., André, B., & Collado-Vides, J. (1998). Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *Journal of Molecular Biology*, 281(5), 827-842.

- Vanet, A., Marsan, L., & Sagot, M. F. (1999). Promoter sequences and algorithmical methods for identifying them. *Research in Microbiology*, 150(9), 779-799.
- Varki, A., & Altheide, T. K. (2005). Comparing the human and chimpanzee genomes: searching for needles in a haystack. *Genome Research*, 15(12), 1746-1758.
- Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning* (pp. 1096-1103). ACM.
- Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 3371-3408.
- Wang, C., Zhang, M., & Zhang, Z. (2013). Computational identification of active enhancers in model organisms. *Genomics, Proteomics & Bioinformatics*, 11(3), 142-150.
- Wang, D., & Lee, N. K. (2008). MISCORE: mismatch-based matrix similarity scores for DNA motif detection. In *International Conference on Neural Information Processing* (pp. 478-485). Springer, Berlin, Heidelberg.
- Wang, J., Zhuang, J., Iyer, S., Lin, X., Whitfield, T. W., Greven, M. C., Pierce, B. G., Dong, X., Kundaje, A., Cheng, Y., & Rando, O. J. (2012). Sequence features and chromatin structure around the genomic regions bound by 119 human transcription factors. *Genome Research*, 22(9), 1798-1812.
- Wei, Z., & Jensen, S. T. (2006). GAME: detecting cis-regulatory elements using a genetic algorithm. *Bioinformatics*, 22(13), 1577-1584.
- Weigend, A. S., Rumelhart, D. E., & Huberman, B. A. (1990). Generalization by Weight-Elimination with Application to Forecasting. *Neural Information Processing Systems*, 90, 875-882.

- Wijaya, E., Rajaraman, K., Yiu, S. M., & Sung, W. K. (2007). Detection of generic spaced motifs using submotif pattern mining. *Bioinformatics*, 23(12), 1476-1485.
- Wijaya, E., Yiu, S. M., Son, N. T., Kanagasabai, R., & Sung, W. K. (2008). MotifVoter: a novel ensemble method for fine-grained integration of generic motif finders. *Bioinformatics*, 24(20), 2288-2295.
- Won, K. J., Chepelev, I., Ren, B., & Wang, W. (2008). Prediction of regulatory elements in mammalian genomes using chromatin signatures. *BMC Bioinformatics*, 9(1), 547.
- Wong, Y. S., Lee, N. K., & Omar, N. (2016). GMFR-CNN: An Integration of Gapped Motif Feature Representation and Deep Learning Approach for Enhancer Prediction. In *Proceedings of the 7th International Conference on Computational Systems-Biology and Bioinformatics* (pp. 41-47). ACM.
- Workman, C. T., & Stormo, G. D. (2000). ANN-Spec: a method for discovering transcription factor binding sites with improved specificity. *Biocomputing*, 5, 467-478.
- Wunderlich, Z., & Mirny, L. (2009). Different gene regulation strategies revealed by analysis of binding motifs. *Trends in Genetics*, 25(10), 434-440.
- Xiong, H. Y., Alipanahi, B., Lee, L. J., Bretschneider, H., Merico, D., Yuen, R. K., Hua, Y., Gueroussov, S., Najafabadi, H. S., Hughes, T. R., & Morris, Q. (2015). The human splicing code reveals new insights into the genetic determinants of disease. *Science*, 347(6218), 1254806.
- Xu, Y., Mo, T., Feng, Q., Zhong, P., Lai, M., Eric, I., & Chang, C. (2014). Deep learning of feature representation with multiple instance learning for medical image analysis. In *Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1626-1630). IEEE.

- Yáñez-Cuna, J. O., Arnold, C. D., Stampfel, G., Boryń, Ł. M., Gerlach, D., Rath, M., & Stark, A. (2014). Dissection of thousands of cell type-specific enhancers identifies dinucleotide repeat motifs as general enhancer features. *Genome Research*, 24(7), 1147-1156.
- Yang, B., Liu, F., Ren, C., Ouyang, Z., Xie, Z., Bo, X., & Shu, W. (2017). BiRen: predicting enhancers with a deep-learning-based model using the DNA sequence alone. *Bioinformatics*, 33(13), 1930-1936.
- Yu, Q., Huo, H., Chen, X., Guo, H., Vitter, J. S., & Huan, J. (2015). An efficient algorithm for discovering motifs in large DNA data sets. *IEEE Transactions on Nanobioscience*, 14(5), 535-544.
- Yu, Q., Huo, H., & Feng, D. (2016). PairMotifChIP: a fast algorithm for discovery of patterns conserved in large ChIP-seq data sets. *BioMed Research International*, 2016, 1-10.
- Zambelli, F., Pesole, G., & Pavesi, G. (2012). Motif discovery and transcription factor binding sites before and after the next-generation sequencing era. *Briefings in Bioinformatics*, 14(2), 225-237.
- Zhang, H., Zhu, Q., & Jia, X. (2015). An Effective Method for Gender Classification with Convolutional Neural Networks. In *International Conference on Algorithms and Architectures for Parallel Processing* (pp. 78-91). Springer International Publishing.
- Zhang, Y., Wang, P., & Yan, M. (2016). An Entropy-Based Position Projection Algorithm for Motif Discovery. *BioMedical Research International*, 2016, 1-10.
- Zhu, B., Zhang, W., Zhang, T., Liu, B., & Jiang, J. (2015). Genome-wide prediction and validation of intergenic enhancers in Arabidopsis using open chromatin signatures. *The Plant Cell*, 27(9), 2415-2426.

- Zhu, L., Yang, J., Song, J. N., Chou, K. C., & Shen, H. B. (2010). Improving the accuracy of predicting disulfide connectivity by feature selection. *Journal of Computational Chemistry*, 31(7), 1478-1485.
- Zhu, Y., Sun, L., Chen, Z., Whitaker, J. W., Wang, T., & Wang, W. (2013). Predicting enhancer transcription and activity from chromatin modifications. *Nucleic Acids Research*, 41(22), 10032-10043.
- Zia, A. & Moses, A. M. (2012). Towards a theoretical understanding of false positives in DNA motif finding. *BMC Bioinformatics*, 13(1), 151.
- Zimin, A., Cornish, A., Maudhoo, M., Gibbs, R., Zhang, X., Pandey, S., Meehan, D.T., Wipfler, K., Bosinger, S.E., Johnson, Z.P., & Tharp, G.K. (2014). A new rhesus macaque assembly and annotation for next-generation sequencing analyses. *Biology Direct*, 9(1), 1.

APPENDICES

Appendix 1 – Python code for preprocessing stage

```
import numpy as np
import itertools
from numpy import matrix, ones
from itertools import product, chain, combinations, permutations, izip
from collections import Counter
from operator import sub, truediv
import sys
import math
#####

def diffs(a,b):
    # collect sliding window differences
    # length of window determined by the shorter array
    # if a,b are not arrays, need to replace b[...] - a with
    # a list comprehension
    n,m=len(a),len(b)
    if n>m:
        # ensure s is the shorter
        b,a=a,b # switch
        n,m=len(a),len(b)
        # may need to correct for sign switch
    result=[]
    for i in range(0,1+m-n):
        result.append(b[i:i+n]-a)
    return result

#####

def alldiffs(a,b):
    # collect all the differences for elements of a and b
    # a,b could be lists or arrays of arrays, or 2d arrays
    result=[]
    for aa in a:
        for bb in b:
            result.append(diffs(aa,bb))
    return result

#####|
```

```

def count_total(a,b):
    #count the total number of element for two arrays in different list
    #return [sum(map(len, i)) for i in product(a, b)]
    y= lambda x:len(x)
    result=[]
    for a1 in a:
        for b1 in b:
            result.append(y(a1) + y(b1))
    return result
#####
def count_zero(obj):
    #count the total number of zero for two arrays in different list
    if isinstance(obj,list):
        return list(map(count_zero,obj))
    else:
        return Counter(obj)[0]

#def total_minus_intersect(mat):
#####
def distance_matrix(pattern):
    # to get the side length, solve for n where len(pattern) = n*(n + 1)/2
    #(triangular number formula)
    side_length = (int(math.sqrt(1 + 8 * len(pattern))) - 1) / 2 + 1
    assert (side_length * (side_length - 1)) // 2 == len(pattern),
    "Pattern length must be a triangular number."

    # create the grid
    grid = [[0] * side_length for i in range(side_length)]

    # fill in the grid
    position = 0
    for i in range(0, side_length - 1):
        for j in range(0, side_length - 1 - i):
            element = pattern[position]; position += 1
            grid[i][i + j + 1] = element # fill in the upper triangle
            grid[i + j + 1][i] = element # fill in the upper triangle

    return grid
#####

```

```

# define the 3 arrays
# each is a list of 1d arrays

a=[np.array([1,2,3,4,1,3]),np.array([3,4,4,3])]
b=[np.array([4,3,3,1,3])]
#c=[np.array([1,2]),np.array([4,3])]

output=[]
#comb_set=list(itertools.combinations([a,b,c],2))
for i,j in itertools.combinations([a,b],2): #a,b,c
    all_diffs=alldiffs(i,j)
    total=count_total(i,j)
    zero=count_zero(all_diffs)
    total=np.array(total)
    #total=total[:,np.newaxis]
    #zero=np.array(zero)
    #union=total-zero
    union=map(sub,total,zero)
    #union=np.array(union)
    zero=np.array(zero).tolist()
    union=np.array(union).tolist()
    #union=union.tolist()
    #print zero
    #print union
    union=[list(x) for x in union]
    #print union

    sim=[[float(aaa) / bbb for (aaa, bbb) in itertools.izip(aa, bb)] \
          for (aa, bb) in itertools.izip(zero, union)]
    #print sim
    sim_comb=sum(sim,[])
    #print sim_comb
    sum_of_sim=sum(sim_comb)
    #print sum_of_sim
    number_sum=len(sim_comb)
    #print number_sum
    ave=sum_of_sim/number_sum
    one_ave=1-ave
    output+= [one_ave]

#print output
print union
print zero
print number_sum
print one_ave
def matrix_to_string(matrix): return "\n".join("\t".join(str(x) for x in row)
                                                for row in distance_matrix(output))

if __name__ == "__main__":
    print(matrix_to_string(distance_matrix(output)))

```

Appendix 2 – Learning of the stacked autoencoders

```
load creb_clust2;

train_x = double(train_x);
test_x = double(test_x);
train_y = double(train_y);
test_y = double(test_y);

%% ex1 train a 25 and 12 hidden units SDAE and use it to initialize a FFNN
% Setup and train a stacked denoising autoencoder (SDAE)
rand('state',0)
sae = saesetup([76 25 12]);
sae.ae{1}.activation_function = 'sigm';
sae.ae{1}.learningRate = 1;
sae.ae{1}.inputZeroMaskedFraction = 0.5;
sae.ae{2}.activation_function = 'sigm';
sae.ae{2}.learningRate = 1;
sae.ae{2}.inputZeroMaskedFraction = 0.5;
opts.numepochs = 1;
opts.batchsize = 5;
sae = saetrain(sae, train_x, opts);

% Use the SDAE to initialize a FFNN
nn = nnsetup([76 25 12 1]);
nn.activation_function = 'sigm';
nn.learningRate = 1;
nn.W{1} = sae.ae{1}.W{1};
nn.W{2} = sae.ae{2}.W{1};

% Train the FFNN
opts.numepochs = 1;
opts.batchsize = 5;
nn = nntrain(nn, train_x, train_y, opts);
[er, bad] = nntest(nn, test_x, test_y);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

labels = nnpredict(nn, test_x);

nn.testing = 1;
nn = nnff(nn, test_x, zeros(size(test_x,1), nn.size(end)));
nn.testing = 0;
[dummy, i] = max(nn.a{end}, [], 2);
labels = i;

dummy1=round(dummy);
[I,J]=find(test_y==1);
expected = J;
```

```

addition=dummy1+test_y;
minus=dummy1-test_y;
%addition=expected+dummy;
%minus=expected-dummy;
fp=sum(minus==1);
fn=sum(minus==-1);
tp=sum(addition==2);
tn=sum(addition==0);

fprintf('True positive appears %d times\n',tp);

fprintf('True negative appears %d times\n',tn);

fprintf('False positive appears %d times\n',fp);

fprintf('False negative appears %d times\n\n',fn);

Precision=tp/(tp+fp);
fprintf('Precision : %d\n\n', Precision);

Recall=tp/(tp+fn);
fprintf('Recall : %d\n', Recall);
FPR=fp/(fp+tn);
fprintf('False positive rate: %d\n', FPR);

```

Appendix 3 – Conversion of DNA sequence into binary format

```
# convert sequence into binary 1,0
import re
import numpy as np
import csv

with open('chr18_dbgkm_p300_merge_100bp.txt') as file: #gm12878_pos_all_2.txt
    seqs=file.read().splitlines()
    seqs=[val[:] for val in seqs]

d = {'A' : '1000', 'C' : '0100', 'G': '0010', 'T': '0001', 'a': '1000', 'c': '0100', 'g': '0010', 't': '0001'}

with open('chr18_dbgkm_p300_merge_100bp.csv', 'wb') as testfile:
    csv_writer=csv.writer(testfile)
    for p in seqs:
        for c in p:
            p = re.sub(c,d[c],p)

        #print p
        csv_writer.writerow(p)
```

Appendix 4 – Computation and normalization of the position weight matrix

```
from Bio import SeqIO
import numpy as np
import csv
from sklearn import preprocessing
from Bio.Seq import Seq
from Bio.SeqRecord import SeqRecord
from collections import defaultdict

#Open a FASTA input file of gene nucleotides sequences:
input_file = open('chr18_dbgkm_p300_merge_100bp.fasta', 'r') #gm12878_pos_all_2.fa
output_file = open('chr18_dbgkm_p300_merge_100bp.csv', 'w') #gm12878_pos_all_2
for cur_record in SeqIO.parse(input_file, "fasta") :

    A_count=cur_record.seq.count('A')
    C_count=cur_record.seq.count('C')
    G_count=cur_record.seq.count('G')
    T_count=cur_record.seq.count('T')

    output_line = '%f,%f,%f,%f\n' % \
        (A_count,C_count,G_count,T_count)

    output_file.write(output_line)
output_file.close()
input_file.close()
#norm_code.py
#normalize the dimercount using feature scaling
#normalization = x-min/max-min

all_data=np.loadtxt(open("c:/Python27/chr18_dbgkm_p300_merge_100bp.csv","r"), #gm12878_pos_all_2
delimiter=",",
skiprows=0,
dtype=np.float64)

x=all_data[:]
x=x/101
np.savetxt("chr18_dbgkm_p300_norm.csv", x, delimiter=",")#gm12878_pos_all_norm_2.csv
#print('total number of samples (rows):', x.shape[0])
#print('total number of features (columns):', x.shape[1])
#minmax_scale = preprocessing.MinMaxScaler(feature_range=(0, 1)).fit(x)
```

Appendix 5 – Learning of the convolutional neural network taking GMFR as input

```
load elk1_3_13; %y_cebpb_try5 121000 - 1000

train_x = double(reshape(train_x',10,10,10160));
test_x = double(reshape(test_x',10,10,1000));
train_y = double(train_y');
test_y = double(test_y');

%% ex1 Train a 6c-2s-12c-2s Convolutional neural network
%will run 1 epoch in about 200 second and get around 11% error.
%With 100 epochs you'll get around 1.2% error
rand('state',0)

cnn.layers = {
    struct('type', 'i') %input layer      3   3
    struct('type', 'c', 'outputmaps', 6, 'kernelsize',3, 'activation', 'ReLu') %convolution layer %2 1
    struct('type', 's', 'scale', 2, 'method','m') %sub sampling layer 1
    struct('type', 'c', 'outputmaps', 12, 'kernelsize',3, 'activation', 'ReLu') %convolution layer %4 1
    struct('type', 's', 'scale', 2, 'method', 'm') %subsampling layer 1
};

opts.alpha = 0.8 ; % 0.8
opts.batchsize = 10;%cebpb-24, tfap2a-10, yy1- 8, ctcf-23,ep300-2 % bs - 2
opts.numepochs = 1; %200

cnn = cnnsetup(cnn, train_x, train_y);
cnn = cnnttrain(cnn, train_x, train_y, opts);

[er, bad] = cnntest(cnn, test_x, test_y);
```

```
% feedforward
cnn = cnnff(cnn, test_x);
[~,h] = max(cnn.o);
[~,a] = max(test_y);
bad = find(h ~= a);

er = numel(bad) / size(test_y, 2);

dummy1=round(cnn.o');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

addition=dummy1+test_y';
minus=dummy1-test_y';
%addition=expected+dummy;
%minus=expected-dummy;
fp=sum(minus==1);
fn=sum(minus==-1);
tp=sum(addition==2);
tn=sum(addition==0);

fprintf('True positive appears %d times\n',tp);

fprintf('True negative appears %d times\n',tn);

fprintf('False positive appears %d times\n',fp);

fprintf('False negative appears %d times\n\n',fn);
```

```
Precision=tp/(tp+fp);
fprintf('Precision : %d\n', Precision);

Recall=tp/(tp+fn);
fprintf('Recall : %d\n', Recall);
FPR=fp/(fp+tn);
fprintf('False positive rate: %d\n', FPR);

f_measure=2*((Precision*Recall)/(Precision+Recall));
fprintf('F-measure: %d\n', f_measure);

accuracy= (tp+tn)/1000;
fprintf('Accuracy: %d\n', accuracy);
```

Appendix 6 – Merits

List of Publications

- Lee, N. K., Farah, A. L., Wong, Y. S., & Omar, N. (2018). DeepFinder: An integration of feature-based and deep learning approach for DNA motif discovery. *Journal of Biotechnology and Biotechnological Equipment*. doi: 10.1080/13102818.2018.1438209
- Wong, Y. S., Lee, N. K., & Omar, N. (2016). GMFR-CNN: An Integration of Gapped Motif Feature Representation and Deep Learning Approach for Enhancer Prediction. In *Proceedings of the 7th International Conference on Computational Systems-Biology and Bioinformatics* (pp. 41-47). ACM.
- Omar, N., Wong, Y. S., Xi, L., Chong, Y. L., Abdullah, M. T., & Lee, N. K. (2017). Enhancer Prediction in Proboscis Monkey Genome: A Comparative Study. In *10th International Conference on Information Technology in Asia (CITA'17) 2017*, 23-24 May. Kuching.