Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Department of Software

Design a System for Prediction Transcription Factor Binding Sites of DNA Sequence Based on CNN Algorithm

A Thesis

Submitted to the Council of the College of Information Technology for Postgraduate Studies of University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology-Software

By

Faisal Abdullah Aziz Laftah

Supervised by

Asst. Prof. Dr. Sura Zaki AL-Rashid

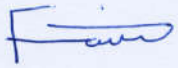
2022 A.D. **1444 A.H.**

﴿ فَتَعَالَى اللَّهُ الْمَلِكُ الْحَقُّ وَلا تَعْجَلْ بِالْقُرْآنِ مِن قَبْلِ أَن يُقْضَى إِلَيْكَ وَحْيُهُ وَقُل رَّبِّ زِدْنِي عِلْمًا ﴾

سورة طه - آية ١١٤

Declaration

I hereby declare that this thesis entitled "**Design a System for Prediction Transcription Factor Binding Sites of DNA Sequence Based on CNN Algorithm**", submitted to the University of Babylon in partial fulfillment of the requirement for the degree of Master in Information Technology \ Software, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose source is appropriately cited in the references.

Signature: 

Name: Faisal Abdullah Aziz

Date: 29 / 9 /2022

Supervisor Certification

I certify that the thesis entitled (**Design a System for Prediction Transcription Factor Binding Sites of DNA Sequence Based on CNN Algorithm**) was prepared under my supervision at the department of Software/ College of Information Technology/ University of Babylon as partial fulfillment of the requirements of the degree of Master in Information Technology-Software.

Signature:

Supervisor Name: Asst. Prof. Dr. Sura Zaki AL-Rashid

Date: 2 /10/2022

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled "Design a System for Prediction Transcription Factor Binding Sites of DNA Sequence Based on CNN Algorithm" for debate by the examination committee.

Signature:

Asst. Prof. Dr. Ahmed Saleem Abbas

Head of Software Department

Date: 2 / 10 /2022

Certification of the Examination Committee

We hereby certify that we have studied the dissertation entitled (**Design a System for Prediction Transcription Factor Binding Sites of DNA Sequence Based on CNN Algorithm**) presented by the student (**Faisal Abdullah Aziz**) and examined him/her in its content and what is related to it, and that, in our opinion, it is adequate with (Viva Result) standing as a thesis for the degree of Master in Information Technology-Software.

Signature:
Name: Asaad Noori Hashim
Title: Asst. Prof. Dr.
Date: 2 / 10 / 2022
(Chairman)

Signature:
Name: Enaas Hamood Mhasin
Title: Asst. Prof. Dr.
Date: 2 / 10 / 2022
(Member)

Signature:
Name: Nashwan Jasim Hussein
Title: Asst. Prof. Dr.
Date: 2 / 10 / 2022
(Member)

Signature:
Name: Sura Zaki AL-Rashid
Title: Asst. Prof. Dr.
Date: 2 / 10 / 2022
(Member and Supervisor)

Approved by the Dean of the College of Information Technology, University of Babylon.

Signature:
Name: Hussein Atiya Lafta
Title: Professor
Date: / / 2022
(Dean of Collage of Information Technology)

Dedication

To my mother who facilitates my life,

To my father and sisters who supported me, and finally, to my supervisor Assist.

Prof. Dr. Sura Zaki AL-Rashid

Acknowledgements

In the first place, I honor and thank Almighty God for his blessing showers during my work of investigation in order to complete the study. I offer my warm thanks to my parents, too, for their sacrifices and help in creating the proper environment for research completion and preparing me for my future.

I would like to express my deepest thanks to my supervisor Asst. Prof. Dr. Sura Zaki Al Rashid for his valuable advice, motivation, guidance, and so many fruitful discussions throughout the preparation of this thesis.

I would like to extend my respect and deepest gratitude to the College of Information Technology and the University of Babylon.

Finally, sincere thanks and appreciation go to my true friends for their encouragement and friendship.

Abstract

Computer science is a scientific branch that studies computers and computing, including their theoretical and algorithmic frameworks, hardware and software, and their roles in information processing. Computer science emerged as a distinct discipline in the early 1960s, despite the fact that the electronic digital computer that it studies was invented nearly two decades earlier. Computer science is used to solve many problems in various fields, including industry, medicine, and biology. One of the most significant biological problems is finding a so-called consensus motif inside a DNA molecule. Motif discovery has long been a challenge for biologists because it indicates transcription factor binding sites (TFBS) on DNA sequences. Unfortunately, biological studies have only found some of these sites. In order to do this, a prediction model that accurately predicts binding relationships between transcription factors (TFs) and DNA sequences must be created based on deep learning (DL) algorithms. Deep learning methods have recently been suggested, and competitive results for a transcription factor binding site forecasting task have been shown. Within deep learning, a convolutional neural network (CNN) is a kind of artificial neural network that is widely utilized in DNA's binding site finding.

The proposed system initializes with a data preprocessing stage. After that, the results of this phase will proceed along two paths, with each path consisting of three steps and involving methods like k-mer embedding, one hot encoding, the z-score motif method, and others. Then, the output from these paths will be input into the encoding phase. Following that, the encoded features are entered into CNN for DNA's binding site classification. Finally, the evaluation step has been performed depending on different measures. The results show that the performance of the proposed system is effective. The model achieved a prediction accuracy of 99.26%, a loss ratio of between 5 and 9%, precision, recall, and an F1-score of 100%. In addition, the AUPRC has achieved a ratio of 100%.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Meaning
A	Adenine
ACC	Accuracy measure
AGRIS	Arabidopsis Gene Regulatory Information Server
ANN	Artificial Neural Network
AtcisDB	Arabidopsis cis-regulatory element Database
AtRegNet	Arabidopsis thaliana regulatory network dataset
AtTFDB	Arabidopsis Transcription Factor Database
AUC	Area Under the Curve
AUPRC	Area under the precision-recall curve
C	Cytosine
CNN	Convolution Neural Network
CREs	Cis-regulatory elements
DBN	Deep Belief Networks
DL	Deep Learning
DNA	Deoxyribonucleic acid
ENCODE	Encyclopedia of DNA Elements
FA	Factor analysis
FC	Fully connected layer
FFL	Feed-forward loop
FN	False Negative
FP	False Positive
FSM	Finite-state machine
G	Guanine
GANs	Generative Adversarial Networks

GDVTK	Genome Data Visualization Toolkit
KEGRU	K-mer embedding with Bidirectional Gated Recurrent Unit (GRU) network model
LSA	Latent Semantic Analysis
LSTM	Long-short term memory
MIM	Multi-input module
ML	Machine learning
mRNA	Messenger RNA
MSE	Mean squared error
NMs	Network motifs
PCA	Principal component analysis
PDNAsite	Protein-DNA binding site model
Pre	Precision measure
PWMs	Position weight matrices
RBM	Restricted Boltzmann Machines
Rec	Recall measure
ReLU	Rectified Linear Unit
RF	Random Forest
RNA	Ribonucleic Acid
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SIM	Single-input module
SVM	Support vector machine
T	Thymine
TFBS	Transcription factor Binding Sites
TFs	Transcription factors
Theano or CNTK	TensorFlow

TN	True Negative
TP	True Positive
TRN	Transcription regulation networks
tRNA	Transfer RNA
U	Uracil

List of Algorithms

Chapter One

GENERAL INTRODUCTION

1.1. Introduction

The deoxyribonucleic acid (DNA) molecule is made up of two polynucleotide chains which roll whole to construct a dual helix that contains genetic instructions for all known species to create, operate, grow, and reproduce [1]. There are small pieces of DNA that are selectively attached by one or more proteins (such as TFs) with different activities, called TFBS binding sites [2]. Bioinformatic approaches have been developed for TFBS prediction. Bioinformatics is an interdisciplinary sector that creates methods and software for biological data knowledge, particularly in huge and complex data sets. In general, the aim of bioinformatics is to organize data in a way that allows researchers to access existing information and to submit new entries as they are produced [3]. Bioinformatics utilizes a broad range of computational tools, including sequence and structural alignment, database design and data mining, macromolecular geometry, phylogenetic tree construction, protein structure and task prediction, and gene discovery [4].

Deep learning has been widely implemented and has achieved better performance than traditional statistical and machine learning approaches for many bioinformatic problems. Deep learning structures like deep neural networks, convolutional neural networks, and recurrent neural networks are demonstrated to produce state-of-the-art outcomes on a wide range of tasks in domains like computer vision, natural language processing, audio extraction, and bioinformatics. Deep learning is a relatively new methodology that operates in a hybrid multiple-layer conceptual mode by converting data to a much higher-level abstraction space where the prediction model will be built. This new approach has provided much more appealing solutions for merging heterogeneous data

and is capable of learning complex patterns from multiple simple inputs. One typical deep learning approach is known as the convolutional neural network (CNN). The benefit of CNN is that it does not isolate feature extraction and model learning into two separate phases. In the current work, a model based on CNN structure has been proposed for discovering DNA TF-binding motifs [5].

1.2. Related Works

In 2015, Jian Zhou and Olga G. Troyanskaya presented the DeepSEA model. The goal of this model was to predict non-coding TF-binding sites in a DNA sequence. This study relied on data from the Encyclopedia of DNA Elements (ENCODE), which included 690 TF binding profile information for 160 unique TFs, 125 DHS features, and 104 histone-mark features. DNase I-hypersensitive sites (DHSs) and histone marks are estimated to have more complicated strategies involving numerous chromatin proteins. Thus, accurate sequence-based prediction of chromatin features needs a robust model capable of modeling such complex dependencies. DeepSEA model depends on deep learning algorithms and specifically on deep convolutional networks. A deep convolutional network is a class of multilayer neural network. The model is arranged in a deep neural network-like manner, with a layer-by-layer system that executes a series of functional changes. The DeepSEA model obtained an accuracy of 81.58%, a precision of 94.81%, and a sensitivity of 64.66% [6].

In 2016, Zeng et al. presented the Zeng model. They created a methodical investigation into CNN structures for DNA binding prediction. For the dataset, the scientists utilized 690 TF ChIP-seq studies from Encyclopedia of DNA Elements (ENCODE). The researchers were able to explore the best performing schemes via modifying CNN size,

depth, and other styles. They also discovered that incorporating convolutional filters into a network benefits motif-based activity. They demonstrated the advantages of CNNs in teaching higher sequence characteristics like additional motifs and localized sequence context via trying to compare network operation on a variety of modelling activities that range in complexity. The Zeng method obtained an accuracy of 81.69%, a precision of 83.69%, and a sensitivity of 76.20% [7].

In 2016, Zhou et al. designed the PDNAsite model. In that design, two protein datasets were utilized to test the efficiency of PDNAsite. PDNA-62 was created using the data structure of 62 protein-DNA, whereas PDNA-224 was created using the data of 224 protein-DNA. A protein-DNA binding site is referred to as a PDNAsite. PDNAsite is a valuable method for finding DNA-binding sites. The PDNAsite approach is divided into three phases: feature detection, LSA operation, and ensemble learning. Feature detection utilizes the sequence and sliding windows to recover seven kinds of characteristics from the sequential and structural contexts, respectively. Then, to decrease duplication and lower dimensionality, LSA was practiced to the sub-feature region occupied by PSSM features. Ensemble learning combines numerous SVM base classifiers for prediction. PDNAsite's efficiency has an AUC of 0.928 on average [8].

In 2016, May D. Wang and Hamid Reza presented the DeeperBind technique. The algorithm was trained using data from PBM experiments obtained from the UniProbe database. In addition, a long short-term recurrent convolutional network was applied to predict protein binding specificities with regard to DNA probes, with the goal of adding a positional dimension to the core design by including recurrence into its model. The goal of this effort was to use CNN as a visual feature collector.

It is hoped that combining CNN and LSTM can improve the prediction of protein-DNA binding specificities. DeeperBind's performance has an AUC of 0.86 on average [9].

In 2018, Zhang et al. developed the DeepBind model. This model is based on the collection of 214 available ChIP-seq datasets. DeepBind is the first approach that uses deep CNN to solve the problem of motif extraction, with the goal of identifying TF binding preferences to DNA sequences. DeepBind used one-hot encoding to convert DNA sequences to one-hot arrays, which were later used to learn these vectors and repeatedly upgrade the coefficients of kernel arrays (PWMs). This method's performance relies on an AUC of 0.85 on average [10].

In 2018, Shen et al. proposed a model, named KEGRU. They utilized 125 TF binding sites in ChIP-seq testings to evaluate the performance. In their model, they utilized a bidirectional gated recurrent unit neural network and a k-mer embedding method to discover DNA binding sites. To overcome the dimensional curse induced by one-hot encoding, word embedding was used in the system and implemented in k-mer sequence depiction with the use of the unsupervised learning technique word2vec. The KEGRU network is not just used to improve the suitability of variable-length sequences, but it also allows for the extraction of complicated context from the k-mer sequence. This model achieved an average AUC of 0.959 [11].

In 2019, DeepRam was presented by Chaabane et al. A system based on data from ChIP-seq and CLIP-seq investigations. DeepRam provided a full examination of various deep learning frameworks for predicting DNA and RNA protein binding locations. DeepRam is a deep learning toolset for guessing protein binding sites and motifs from start to finish. It enables users to execute experiments using a variety of cutting-

edge methodologies and handles the difficulty of selecting model parameters in deep learning models with a completely automated model selection strategy. Customers can adjust a deep learning model by modifying its various pieces, such as the input sequence description (one-hot or k-mer embedding), if either to utilize a CNN and the number of layers, or to utilize a recurrent neural network (RNN) and the type and quantity of layers. This approach had an AUC of 0.864 on average [12].

In 2020, Jeon et al. proposed the TbiNet model. The method was based on TF-DNA binding data gained via the ChIP-seq methodology. In TbiNet, a deep neural network founded on attention, predicts transcription factor binding sites in a DNA sequence. The attention mechanism may assign different weight grades to every segment of an input sequence when generating outputs in order to concentrate on significant parts. The CNN layer kernels act like motif scanners, that capturing data from TF-binding motifs in input sequences. The Bi-LSTM layer of TbiNet enables it to understand the regulating grammars of the TF binding motifs acquired by CNN. This approach had an AUC of 0.946 on average [13].

In 2021, Maiada et al. proposed the KNN model. The TFBS dataset utilized in this research was acquired from Kaggle.com. It comprises 2400 TF binding and non-binding site records. Transcription factor binding sites are not yet completely discovered, and this is deemed as a dilemma that could be addressed computationally. This model predicts the transcription factor binding sites of SP1 on the human chromosome utilizing the classification method K-Nearest Neighbors (KNN). This study demonstrates the use of the voting procedure for predicting binding sites, as well as the outperformance of KNN on this kind of data. This technique achieved an average AUC of 0.97 [14].

In 2022, Yuan et al. presented the IBPred model. The original data were split into 187 IBPs and 299 non-IBPs as per ion-binding proteins in phage viruses. IBPs can associate with ions in a non-covalent manner and are significant in biological processes. Because molecular biology experimental approaches for discovering IBPs are still labor-intensive and expensive, it is beneficial to develop computational methods for identifying IBPs rapidly and efficiently. A random forest (RF)-based approach was built to quickly identify IBPs. Depending on the protein sequence information and residues' physicochemical properties, the dipeptide composition combined with the physicochemical correlation among two residues was suggested for obtaining the features. This model achieved an average AUC of 0.865 [15]. The outline of the related works is provided in Table (1.1).

TABLE 1.1: SUMMARY OF THE RELATED WORKS

Paper	Year	Dataset	Preprocessing	Classification	Evaluation (AUC)
[6]	2015	ENCODE	data preprocessing	deep convolutional network	**0.958**
[7]	2016	690 ChIP-seq experiments	one-hot encoding	CNN	**0.886**
[8]	2016	two protein sequences	training and testing phase	SVM classifier	**0.928**
[9]	2016	Protein Binding Microarrays (PBM) experiments	data preprocessing	CNN + LSTM	**0.86**
[10]	2018	ChIP-seq datasets	one-hot encoding	Deep CNN	**0.85**
[11]	2018	TF-binding sites ChIP-seq	Word2vec	CNN + BiGRU	**0.959**
[12]	2019	ChIP-seq and CLIP-seq	one-hot vector	CNN or RNN	**0.864**
[13]	2020	TF-DNA binding data	one-hot encoded matrix	Attention-based CNN	**0.946**
[14]	2021	2400 TF binding and non-binding data	data preprocessing	KNN	**0.97**
[15]	2022	187 IBPs and 299 non-IBPs	Data splitting	RF	**0.865**

1.3. Problem Statement

Transcription factors (TFs) control gene expression by connecting with certain DNA sequence locations known as TFBS. An earlier study found that TFs are considerably preserved in long-time growth and have a direction for binding to specific DNA sequences known as motifs. The discovery of such motifs can aid not in just understanding gene expression, but also in identifying the underlying disease variations and designing treatment drugs.

1.4. Aim of Thesis

The finding of DNA motifs is a critical step in several systems for researching gene activity. The identifying of motifs is critical for locating transcription factor binding sites (TFBSs), which facilitates understanding the principles governing gene expression control. As a consequence, the focus will be on designing a more efficient and accurate method for mining forms.

1.5. Thesis Objectives

- Developing a method for mining motifs where it is more efficient and accurate than previous models.
- Improving the accuracy of classification by using machine learning methods.
- Provide a comprehensive examination for CNN schemes in order to predict DNA binding sites utilizing a huge number of transcription factors.

1.6. Research Contributions

- The 2D CNN system is designed to classify DNA sequences, whether they have a binding region or not.
- The model's performance is improved with padding, which increases the accuracy rate and reduces the loss value.
- Merging the z-score motif finding method with CNN in order to predict TFs that are found in each binding site.

1.7. Research Challenges

This research has various challenges, including:

- Assign *k* length in the k-mer embedding function.

- The spatial and huge data need to be minimized in size and reduced in dimensionality to avoid overfitting.
- The location of the consensus motif in each sequence is unrelated to the other ones.

1.8. Thesis Organization

The thesis is partitioned into five chapters. Each chapter begins with a short background that underlines the key contributions and offers an impression of the chapter. The summaries of the chapters are as follows:

- Chapter Two gives a historical review of (TFBS) previous methods and explains critical aspects such as DNA sequence transcription and gene expression, k-mer embedding, one-hot encoding, PWM matrices, and the z-score consensus motif method. Finally, the classification technique and performance measures are explained.
- Chapter Three explains the proposed model for transcription factor binding sites prediction depending on Deep Learning and Convolutional Neural Networks (CNN) and illustrates full details of each step, beginning with the initial step that deals with dataset operations and concluding with the final step dedicated to describe the neural network that is utilized for motif classification.
- Chapter Four reveals the experimental results and discussion of the suggested model. In this section, a variety of actual dataset samples will be utilized to test the proposed technique, and the results will be analyzed, evaluated, and argued over.
- Chapter Five contains conclusions and suggestions for future studies on this topic, as well as other recommendations for further studies on the proposed approach.

Chapter Two

THEORETICAL BACKGROUND

2.1. Introduction

This chapter describes DNA's operations such as DNA transcription and translation, as well as RNA splicing and protein structure. It also demonstrates the used dataset. Moreover, the concentration will be placed on significant methods that the thesis hugely depends on, like k-mer embedding, one-hot encoding, and probabilistic and enumeration motif mining methods. Furthermore, it outlines the conceptual foundations of deep learning and artificial neural networks and how to use neural networks to predict DNA's binding sites.

2.2. Biological Concepts

2.2.1. Deoxyribonucleic Acid (DNA)

Deoxyribonucleic acid (DNA) is a type of nucleic acid such as RNA and proteins. Within biological cells, DNA is contained within large schemes called chromosomes. They are made up of basic polymeric groups called nucleotides [1]. Nucleotides join together to form larger structures known as polynucleotides. The two polynucleotides are known as DNA strands. Every nucleotide involves a single of four nucleobases (adenine [A], cytosine [C], guanine [G], or thymine [T]), a sugar known as deoxyribose, and a set of phosphates (see Figure 2.1) [16]. A gene is a small piece of DNA that takes into consideration the inheritance functional unit, which contains proteins that control the transcription rate of genetic data called TFs. One of the significant factors in the gene expression process includes transcription factors. It can regulate the transmission of genetic information between DNA and messenger RNA through binding a DNA sequence to a region known as Transcription Factor Binding Sites (TFBS). A TATA box is a form of promoter sequence that guides other molecules where to start transcription process (see Figure 2.2). The regulatory areas (known as promoters) that have a

transcription factor binding site and a TATA box are located directly above transcription start sites, which are where transcription factors and RNA polymerase II (Pol II) are assembled to start transcription [17]. Introns are non-coding DNA regions that are deleted via RNA splicing during RNA product generation. Exons are DNA sequences that contain protein-coding information and need the required codons or information for protein production.

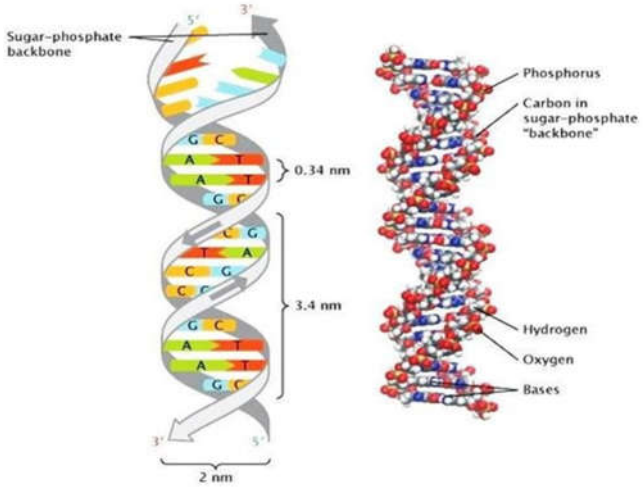


Figure 2.1: Structure of DNA sequence [16]

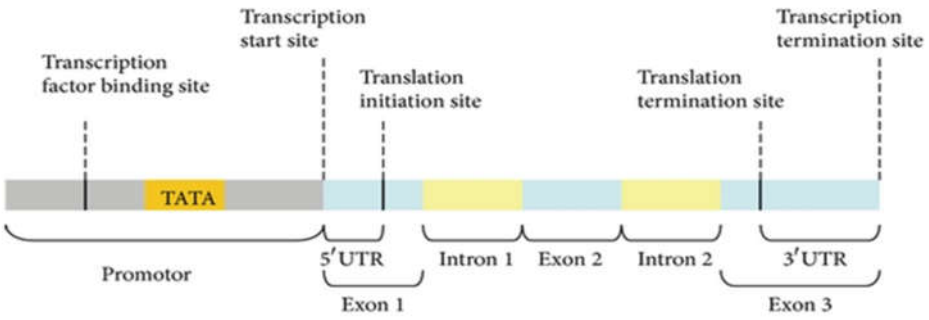


Figure 2.2: TF Binding sites in DNA sequence [17]

2.2.2. Ribonucleic Acid (RNA)

Ribonucleic acid (RNA) is a polymeric element involved in gene coding, decoding, regulation, and production. The core of RNA is made of a bit more distinct sugar than DNA, ribose rather than deoxyribose, and one of the bases is somewhat dissimilar, Uracil (U)

rather than thymine (T) [1]. The other bases, however, are the same: A, C, and G (see Figure 2.3) [18]. RNA and DNA are both nucleic acids and are essential elements of an organism, as are proteins and carbohydrates. Cellular organisms utilize mRNA to transfer genetic information (represented by the nitrogenous bases G, U, A, and C) that controls the creation of some proteins. mRNA is generated during the transcription operation, when an enzyme (RNA polymerase) changes the gene into main transcript mRNA (also known as pre-mRNA).

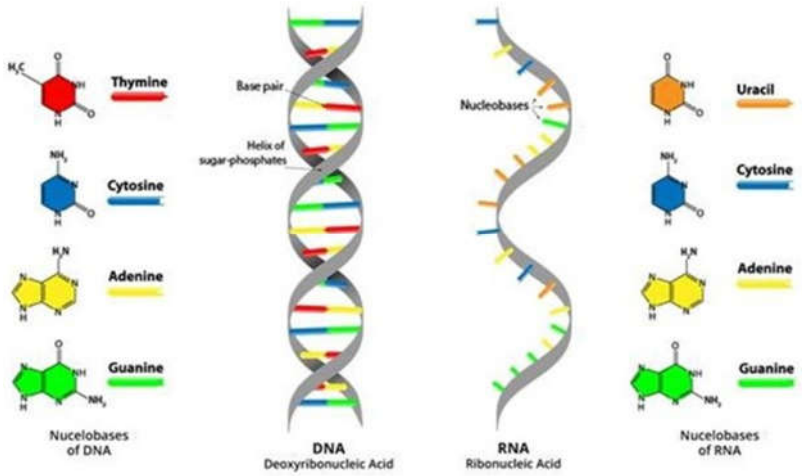


FIGURE 2.3: DIFFERENCES BETWEEN DNA AND RNA MOLECULES [18]

2.2.3. Transcription Factors (TFs)

Proteins are huge macromolecules that are made up of one or more lengthy chains of amino acid residues [1]. Proteins play a wide range of roles within organisms, including DNA replication, transcription, and catalyzing metabolic activities. As shown in Figure 2.4, a polypeptide is a linear chain of amino acid residues that has fewer than 20–30 residues [19]. Proteins differ significantly from one another in their sequence of amino acids. There are 20 amino acids in the string of letters that constitute the protein sequence. These are from A-Z except for these letters (BOX, JUZ).

Examples of proteins are DNA-binding proteins such as TFs and NAC-domain proteins that are responsible for transcriptional regulation in specific plants. Transcription factors are proteins that identify and bind to certain chromosomal DNA sequences and guide their transcription [20]. So, they are responsible for the regulation of gene expression. They are created at various periods and locations throughout an organism's life.

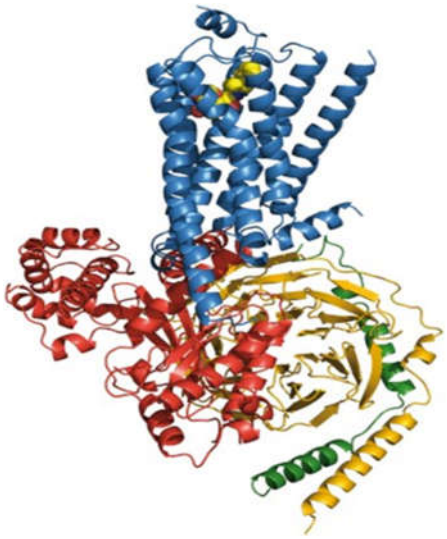


FIGURE 2.4: PROTEIN SHAPE [19]

2.2.4. Motif and its types

Sequence motifs (or consensus motifs) are small, repetitive DNA patterns that are thought to serve a biological purpose. They frequently denote sequence binding sites for proteins such as transcription factors (TFs) and other molecules (as shown in Figure 2.5) [21]. Other than the sequence motif, there are other types of motifs, such as shape motif and network motif. The shape motif represents the value of a feature that is within the estimated range at each location [22].

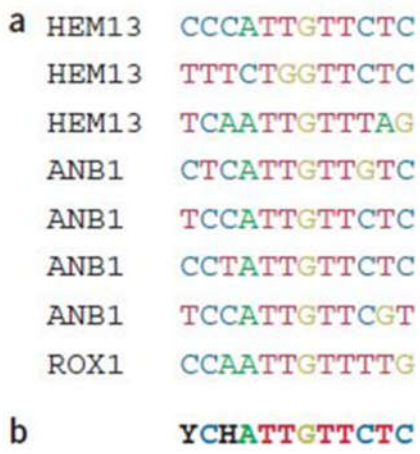


FIGURE 2.5: CONSENSUS MOTIF (A) THREE S. CEREVISIAE GENES CONTAIN EIGHT GENETIC BINDING SITES. (B) NEW CONSENSUS SEQUENCE [21]

Shape motif detection includes comparing shape-feature profiles from DBP-bound areas (positives) with non-bound areas (negatives). The third and significant type is network motifs (NMs), which are transcription regulation networks (TRN) of studied organisms that seem to be composed of a small number of repeating regulatory patterns [23]. NMs are classified into four types, as shown in Figure 2.6:

a) Auto regulation: If transcription factor Y regulates gene X without any other interactions, this is referred to as "auto regulation". Auto regulation can be used as a starting point for interpreting the functions of network motifs.
b) Feed-forward loop: The second type of network motif is the feed-forward loop (FFL). This network is made up of a regulator X, which regulates Y, and a gene Z, which is controlled by both X and Y.
c) Single-input module (SIM): The SIM is the NM family's third form. A single input module has a straightforward pattern in which a regulator X regulates a collection of gene products.

d) Multi-input module (MIM): The last and fourth type is MIM. A multi-input module is a transcription network that is made up of a group of regulators that work together to regulate a group of output genes.

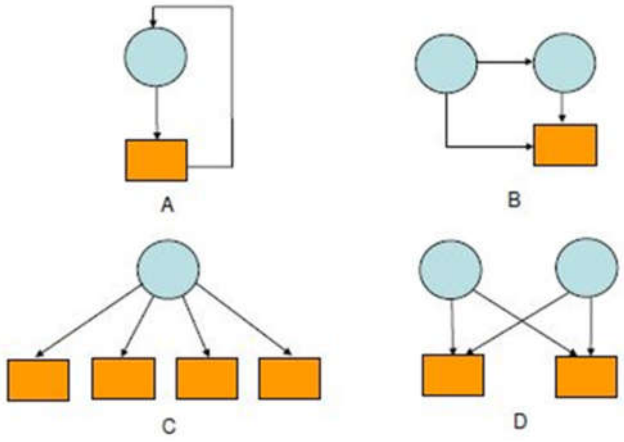


FIGURE 2.6: THE FOUR TYPES OF NETWORK MOTIFS (NMS) [23]

2.2.5. DNA Transcription and Gene Expression

DNA transcription is the first step of the gene expression synthesis process. The process through which the commands in DNA are transformed into a functioning output, like a protein, is known as gene expression. Figure (2.7) illustrates the steps of gene expression production [24]. DNA transcription is the process of transcribing the genetic code in DNA to produce mRNA, or messenger RNA. This is performed via an enzyme named RNA polymerase that transforms existing nucleotides from the cell's nucleus into mRNA. Translation happens after messenger RNA (mRNA) transports the copied "message" from DNA to the cell's protein-creating components known as ribosomes. The message conveyed via mRNA is read by transfer RNA, a carrying molecule (tRNA). Three characters (a codon) are scanned at a moment's notice from the mRNA. Every codon indicates a different amino acid. For instance, the three bases "GGU" represent the amino acid glycine. Every amino acid is then bounded to its tRNA molecule. When this mRNA sequence is examined,

every tRNA molecule offers an amino acid to a ribosome and immediately connects to the relevant codon on the mRNA molecule. Once the tRNA is linked, its amino acid is released, and the adjacent amino acids join together to create a polypeptide, which is a long string of amino acids. This step is repeated until a protein is formed [25].

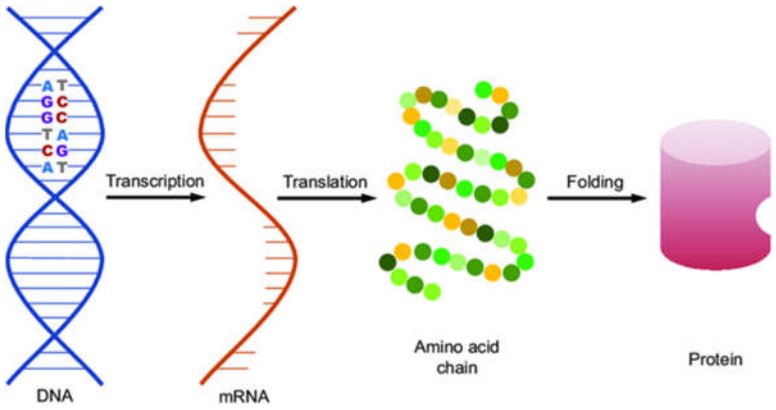


FIGURE 2.7: THE STEPS OF GENE EXPRESSION SYNTHESIS [24]

2.3. Dataset

There are many biological datasets. The database used for this thesis is obtained from a publicly available data resource, namely the Arabidopsis Gene Regulatory Information Server (AGRIS), which gives a complete reference for gene creation investigations in the plant Arabidopsis thaliana (see Figure 2.8) [26]. Since it is a helpful model for understanding cellular and molecular biology, Arabidopsis thaliana is already usually employed in plant studies, including genetics and evolution. Inside the AGRIS server, there are three data sources, which are AtTFDB, AtcisDB, and AtRegNet. In addition, this dataset makes major contributions to the detection of the whole collection of TF-DNA interactions [27].

FIGURE 2.8: ARABIDOPSIS THALIANA PLANT [26]

2.3.1. AtTFDB Database

Since the last major release, the Arabidopsis Transcription Factor Database (AtTFDB) has included 83 TFs, which were found by using the same criteria as previously mentioned. Single TF information can be retrieved by browsing or searching with particular gene locus identifiers (AGI ID, Atxgxxxxx) or equivalent genetic designations. The output table includes the AGI ID, gene variants, and connects to MIPS (http://mips.helmholtz-muenchen.de/plant/athal/), SALK (http://signal.Salk.edu/cgi-bin/tdnaexpress/), and TAIR (http://Arabidopsis.Org/), nucleotide and protein sequences, and information on the TF's activity in a regulation [27].

2.3.2. AtcisDB Database

The Arabidopsis cis-regulatory element Database (AtcisDB) is a navigable related database that contains a variety of data kinds, such as TF-binding sites and entire promoter sequence information. To improve flexibility and expansion, the Genome Data Visualization Toolkit (GDVTK) is utilized to show gene regulatory data on locations of the genome in prior AGRIS releases which are substituted by the Genome Browser (http://gmod.Org/wiki/GBrowse). This browser facilitates the combination of AtcisDB with the display and merging of every genomic

data type, such as word counts and CREs (cis-regulatory elements) position information. The most recent AGRIS version incorporates genome-wide word counting into AtcisDB [27].

2.3.3. AtRegNet Database

In the Arabidopsis thaliana regulatory network dataset (AtRegNet), only TFs and their direct target genes are recorded and shown. At the moment, AtRegNet contains information on 8070 target genes, 64 transcription factors, and three transcription factor groups that have physical direct regulatory links. In AtRegNet, complexity is described when more than one transcription factor is attracted to DNA at the same time, generally in a synergistic form. The data comprises high-throughput in vivo DNA-binding approaches like ChIP–Chip and ChIP-Seq [27].

2.4. Data Preprocessing

Data preprocessing is a data mining approach that includes converting raw data into a useful form. Data preprocessing is used for representing complicated schemes with attributes.

2.4.1. Data Cleaning

Data cleaning is the act of detecting and correcting errors and conflicts in data so as to ensure data quality. Also, missing values should be resolved because many systems do not accept them [28]. Handling missing values is done by removing missing and unrelated values to produce sequences that only comprise letters (AGCT) via:

1. Remove sequences that match the first value within that is not one of the four characters (A, G, C, T), regardless of whether the value is missing (gap) or irrelevant.

2. Receiving sequences that included only letters (AGCT)

The steps for handling missing values are depicted in Algorithm (2.1) [29].

ALGORITHM 2.1: THE STEPS OF HANDLING MISSING VALUES	
Input: *two dimensional array of nucleotides(AGCT),* $a_{(ij)}$*, Where i is index for sequences and j is index for nucleotides (AGCT)* DNA_i *is an array of Dataset (AGCT)for one dimensional*	
Output: *two dimensional B the new dataset only has [A, G, C, T]*	
Begin	
Step 1	*for i in N, where N is no. of sequences*
Step 2	*flag =true*
Step 3	*for j in* seq_i *from DNAi*
Step 4	*if* a_{ij} *not in [A,G,C,T]*
Step 5	*flag =false*
Step 6	*end if*
Step 7	*end j*
Step 8	*if flag = true*
Step 9	*B.append =* seq_i
Step 10	*end if*
Step 11	*end i*
End	

2.4.2. Data Integration

Data integration is the task of merging data from several places and giving the user a uniform representation of this data. The subject of creating data integration systems is essential in recent real-world applications and is characterized by a variety of theoretically fascinating issues. The definition of the correlation between the data at the resources and those in the global template is one of the most significant components of the design of a data integration framework. In principle, numerous global databases are authorized for such a data integration system in relation to a particular source database [30].

2.4.3. Data Reduction

The transition of numeric or alphabetic digital information obtained empirically or experimentally into a rectified, organized, and simpler form is known as data reduction. The primary idea is to reduce

the large volumes of data in the used database to the important sections through dimensionality reduction, in which the data properties or dimensions are decreased. Data reduction has been frequently utilized in data mining to facilitate analysis. Widely used techniques include principal component analysis (PCA) and factor analysis (FA) are used [31].

2.5. Alignment-free Methods

Alignment-free sequence analysis techniques on molecular sequence and structure data give alternatives to alignment-based techniques in bioinformatics. Alignment-free methods are divided into five types: a) k-mer/word frequency methods, b) common substring length methods, c) number of (spaced) word matches methods, d) micro-alignment methods, and e) information theory methods [32].

2.5.1. K-mer Embedding

Recent studies have focused on *k*-mer embeddings, or genuine depictions of words or k-mers generated via distributional semantic models [33]. In this phase, the DNA sequence is split into overlapped k-mers of a certain length and stride window. All subsequences having a *k*-length and stride s have recovered, obtaining a k-mer sequence of length $L = [(L_0 - k)/s] + 1$ (L: number of k-mers, L_0: DNA sequence length k: k-mer length, s: stride window), where each k-mer is archived by a positive number in set C = [1, 2, ….,4^k]. A study will be carried out to determine how to extract feature for these kinds of sequence data belong to C^L with variable length L and how to build a feature map. The embedding phase calculates k-mer co-occurrence statistics and acquires how to map them onto a D-dimensional domain R^D [34]. Additionally, the distributed depiction of k-mers might be effective for predicting DNA methylation and histone occupancy [33].

2.6. State encoding

State encoding generates a distinct template of ones and zeros for every state of a finite-state machine (FSM). Speed, area, or both have traditionally been utilized as design criteria for FSM synthesis [35].

2.6.1. One-hot Encoding

One-hot encoding is a type of state encoding. It is the typical technique for turning DNA sequences into vectors. As a result, the DNA sequence will be vectorized into a binary matrix. As illustrated in Figure 2.9 (A), every location might be depicted via a four-state vector with bases of *A*, *C*, *G*, and *T*. In this case, a DNA sequence termed $S = (s_1, s_2, .. s_i, \ldots, s_n)$ with n nucleotides and a sequence motif size of m, the binary matrix S for DNA sequence is expressed as a $4 \times n$-dimensional column vector, as shown in the equation below.

$$S_{i,j} = \begin{cases} 1 \;\; if \; S_{i-m+1} = \; j^{th} \; base \; in \; \{A, C, G, T\} \\ 0 \;\; otherwise \end{cases} \tag{2.1}$$

where i is the nucleotide index and j are the column number according to *A, C, G,* and *T*.

k-mer encoding is frequently used to describe DNA sequences. In DNA sequences, the k-mer is frequently non-random and follows specific patterns. The k-mer is a sequence substring made up of k nucleotides. k-mer frequency data is commonly used in k-mer coding. All potential motifs of length k are represented in the subsequence space. Each conceivable theme is created by continually assigning k elements from the letters *A, C, G*, and *T* [36]. Assuming k is 3, each k-mer subset is mapped to high-dimensional array, as seen in Figure 2.9 (B). As a result, the size of the subsequence space is 4^k.

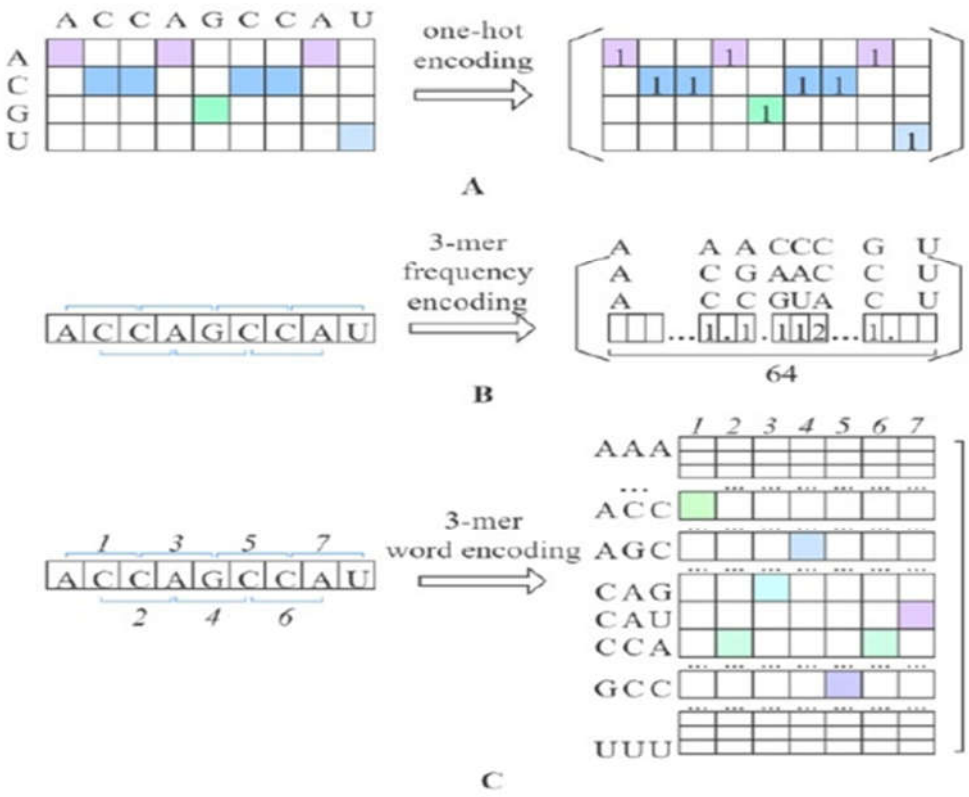


FIGURE 2.9: ONE HOT ENCODING METHOD [36]

2.7. Consensus Motif Finding

The discovery of DNA motifs is a crucial stage in several frameworks that explore gene performance. Motif searching is critical for discovering transcription factor binding sites (TFBSs), which assists in understanding the techniques governing gene creation control [37].

2.7.1. Probabilistic-based Motifs Finding Method

A probabilistic technique creates a probabilistic model known as Position Weight Matrices (PWMs), which is a matrix that provides the frequency at which each nucleotide is located at the sites of the TFBS to identify motifs from non-motifs, hence the need for fewer search terms [37]. PWMs can be built in a variety of ways. The following are the most commonly used approaches, which are comparable to the one proposed by Staden. A base frequency table is constructed by counting how many

times each base occurs at each location in a group of aligned TFBS (see Figure 2.10). The base frequency table comprises four rows (one per letter of the alphabet: *A, C, G*, and T), and the number of columns is similar to the motif length [38]. Though convenient, the PWM has considerable drawback because this method assumes the independence of places within the binding motif. Non-independence of nucleotide distributions in multiple places could possibly mean non-additivity of the binding energy [39].

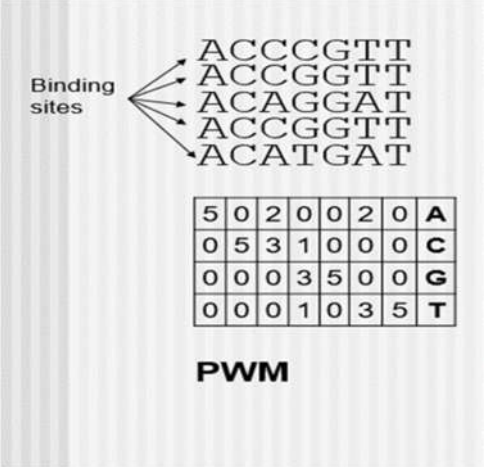


FIGURE 2.10: A DESCRIPTION OF PWMS OPERATION

2.7.2. Enumeration-based Motifs Finding Method

The enumeration method seeks consensus sequences; motifs are predicted depending on word enumeration and word matches. The word enumeration technique algorithms comprehensively scan the entire search space to discover which ones emerge with potential modifications and so commonly find the global optimum. But this also implies that they are exponential-time algorithms that take a lot of time to detect longer motif lengths and are ineffective when dealing with thousands of sequences. There are several types of enumerative approaches, such as simple word enumeration, clustering-based method, tree-based method, etc. Within the

simple word enumeration category, there are two types of algorithms, which are z-score and DREME [37].

In the biological fields, z-scores are commonly employed to measure the importance of pairwise similarities between DNA, RNA, or protein sequences. A high z-score indicates an alignment that is less likely to happen by coincidence. This form of alignment is more likely to be biologically significant. Z-score calculates the distance (in standard deviations) between the specified alignment and the mean value of the other alignments that can be acquired by a permutation of each sequence. Those scores are frequently used to assign a level of importance based on values taken from a standardized statistical distribution. Additionally, higher z-scores indicate a lower likelihood of seeing a value equal to or greater than such values (and thus of greater statistical significance) [40]. The z-score equation is as follows [41]:

$$\mathrm{Z(m)} = \frac{obs(m) - E(m)}{\sigma(\mathrm{m})} \tag{2.2}$$

Where ***obs(m)*** denotes the actual number of occurrences of the motif *m*, ***E(m)*** denotes the expected value of obtaining a specific nucleotide motif across n equal-length sequences, and $\boldsymbol{\sigma(m)}$ is the standard deviation that measures the distance between the given alignment.

2.8. Deep Learning

Deep Learning (DL) (sometimes called Deep Structured Learning, or Deep Machine Learning) is the science of artificial neural networks and related machine learning techniques with multiple hidden layers. For feature obtaining and transformation, it employs a hierarchy of several layers of nonlinear unit operations. Deep learning methods can be supervised or unsupervised [5].

2.8.1. Deep Learning Applications

There has been a massive development in system design and intelligence after the launch of the earlier models for deep learning. Deep learning applications nowadays are diverse but limited. Deep learning can now provide us with better cars, possibly even self-driving cars, better medical image analysis, personalized medical care, sufficient language translation, search recovery filtering, and a multitude of other applications in energy, banking, industrial production, environmental protection, and artistic production [42].

2.8.2. Deep Learning Models

Deep learning models are divided into two types: supervised and unsupervised, each with their own network structure. Because of their effectiveness in solving complex issues, the utilize of supervised and unsupervised deep learning methods has expanded rapidly. Deep learning is becoming more accessible for a wide range of applications due to high-performance computer capabilities, the availability of enormous volumes of data (labelled and unlabeled), and state-of-the-art accessible tools [42].

2.8.2.1. Supervised Deep Learning

Supervised deep learning structures are trained using labelled data. It trains the learning algorithm to generalize from training data and to apply to previously unknown data. After the training phase is completed, the model is tested on a portion of the testing set to predict the output. CNNs are among the most widely utilized supervised deep learning methods. Several CNN-based supervised deep learning structures are AlexNet (5 convolution, 3 max-pooling, 3 fully connected layers, and softmax classifier), LeNet-5 (3 convolution, 2 subsampling (pooling), fully connected layers, and softmax classifier), VGGNet (13 convolution,

5 max-pooling, 3 fully connected layers, and softmax classifier), GoogleNet (with specific architecture) and ResNet (with specific architecture) and others [43].

2.8.2.2. Unsupervised Deep Learning

Unsupervised deep learning methods are essential since unlabeled data is larger than labelled data and they are specifically designed for applications that involve large amounts of unlabeled data. Examples of unsupervised deep learning networks are Restricted Boltzmann Machines (RBM), Deep Belief Networks (DBN), and Generative Adversarial Networks (GANs) [43].

2.8.3. Convolutional Neural Network

Convolutional Neural Network, also named as ConvNet, is a deep learning approach that uses to analyze visual imagery. CNN is recommended because of its capacity to deal with sparse data and improve classification efficiency [44]. Neurons in a typical neural network are completely linked between layers. Hidden layers are those found among the input and output layers. Every hidden layer is made up of multiple neurons, each of which is completely attached to all of the nodes in the previous layer. The fully connected neural network's deep linked network topology doesn't really adapt well to huge images. The most popular solution for huge images is to utilize a convolutional neural network. ConvNet is made up of a series of different sorts of layers that work together to accomplish various tasks. The layers of a standard convolutional neural network are as follows:

- Convolutional layer
- Activation function layer (ReLU)
- Pooling layer

- Fully connected layer (Dense Layer)
- Dropout layer

These components are used to build the CNN scheme. Convolutional and activation layers are typically mixed, succeeded via an optional pooling layer. The network's final layer is a fully connected layer, and the output of the final fully connected layer returns the input image's class results [45].

2.8.3.1. Basic Components of CNN Architecture

One of the most significant features of CNN is prediction. In public, the distinct CNN is composed of two main elements: the feature extractor and the classifier, which are described as follows:

- **Feature Extractor**

Feature Extraction is the preliminary stage of CNN operations; it extracts features and converts them into feature maps. CNN is composed of numerous filters, each of which performs a unique function. As a result, multiple feature maps are produced, each of which corresponds to a different filter. The feature extraction process involves several phases so as to gain the final low-dimensional feature vector, which is then fed into a classifier. The feature extractor is made up of several layers (multiple convolution layers with optional pooling layers). First, it runs through a convolution layer to convolve the filter with the input, producing feature maps that are subsequently reduced with a pooling layer. Afterwards, it utilizes the previously generated feature maps as input feature maps and repeats the process, going layer by layer to extract advanced features and obtain smaller-sized feature maps. The final advanced feature, reduced dimensions of feature maps, is then flattened to form a low-dimensional feature vector to be input into the classifier [45].

- **Classifier**

After extracting feature maps and reducing dimensions by picking the best features among them, the low-dimensional feature vector is fed into a classifier. The classifier provides the likelihood that the input belongs to that class. To accomplish this, the classifier is made up of one or more fully connected layers [45].

2.8.3.2. CNN Infrastructure

The input layer of a CNN model reflects the model's entries (the selected features). The input layer is a matrix of size ($n \times m$), where n denotes the number of samples and m denotes the number of features. The whole CNN architecture is illustrated layer by layer as follows:

a) Convolution Layer

The convolution layer, which employs the convolution function (represented by *) instead of general matrix multiplication, is the primary element of a convolutional neural network. Its parameters are made up of a collection of learnable filters called kernels. The basic goal of the convolutional layer is to find characteristics that are common across the dataset that are found in certain regions of the input image and convert their presence to a feature map. A feature map is constructed for any filter within the layer via iteratively running the filter over sub-sections of the entire image, i.e., by convolving the filter with the given image, incorporating a bias element and then using an activation task [43]. The input region where a filter is implemented is referred to as the local receptive field (see Figure 2.11) [46].

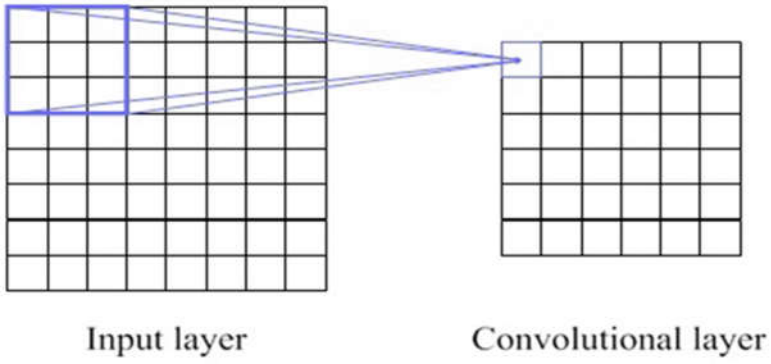


FIGURE 2.11: LOCAL RECEPTIVE FIELD ON INPUT LAYER [46]

The receptive field has the same dimensions as the filter. Figure 2.12 depicts how an S-shaped filter and input are convolved to produce the feature map [47]. The result of the convolution is transformed into a feature map by inserting a bias factor and implementing a nonlinear function.

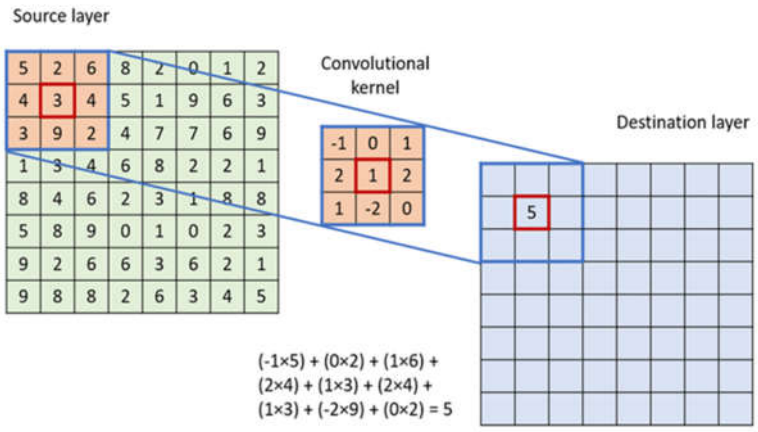


FIGURE 2.12: THE CONVOLUTION OPERATION [47]

Weight sharing refers to the usage of similar filters or weights for all receptive fields in a layer. Each convolutional layer's weights represent the convolution filters, and each convolutional layer may have several filters. Every filter has some feature, such as an edge or a corner, and every filter is dragged over the width and height of the input throughout the front pass, generating a feature map for this filter [43].

A convolutional neural network structure contains multiple hyper-parameters that are utilized to regulate the model's activity. The four most essential hyper-parameters in the ConvNet's convolution layer are listed below:

a) Filter size: filters could be any size larger than 2×2 but less than the input volume. The size of a filter is unrelated to the volume of the input.
b) Number of filters: every appropriate number of filters can be employed with different sizes.
c) Stride: it is the number of pixels that must be shifted altogether to generate the local receptive field of a filter. Moving across and bottom a pixel value is referred to as a stride with one. Overlap will occur if the stride is too short, and vice versa.
d) Padding: this hyper-parameter specifies the number of pixels that will be used to pad the input data. There are two possible values: "valid" or "same". "Valid" means there is no padding, while "same" creates padding with zeros [48].

The general convolution layer output involves a number of feature maps, each of which has been convolved with a different kernel to describe unique features. Let's denote the input source data I, the convolution kernel or weight W, and the bias b. Then, suppose $L \times L$ k-sized filters, W, are convolved with the input data to yield k feature maps termed, each with i and j indices. The following equation describes the convolution operation mathematically:

$$C_{i,j}^{k} = \sum_{m=0}^{L-1} \sum_{n=0}^{L-1} W_{m,n}^{k} * I_{i+m,j+n} + b^{k} \qquad (2.3)$$

As a result, many feature maps are generated, as shown in Figure 2.13 [49]. The convolution process is illustrated in the algorithm (2.2) [50].

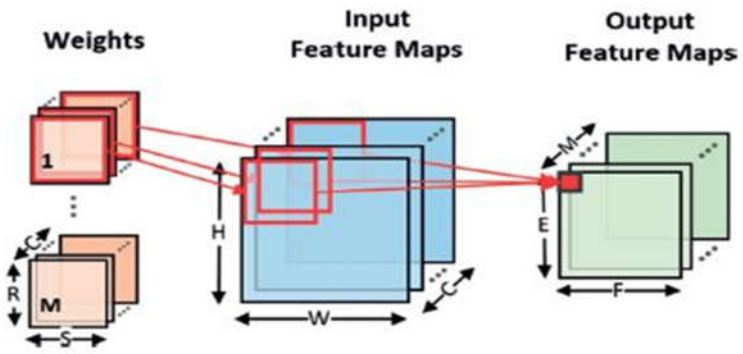


FIGURE 2.13: THE CONVOLUTION LAYER PRODUCED M FEATURE MAPS [49]

ALGORITHM 2.2: THE CONVOLUTION OPERATION
Input: *Two-dimensional array R [n * m] where n is the number of samples and m is the number of features. // Output of Algorithm (3.4)*
Output: *Feature map of the dataset*
Begin
step 1 *for k from 1 to number of filters*
step 2 *initialize mask (W) randomly, initialize bias (b) equal to 1 // b is a constant*
step 3 *for i from 1 to row step 1*
step 4 *for j from 1 to column step 1*
step 5 *net = 0*
step 6 *for m from i-d to i+d // rows and columns of kernel, d specify kernel size*
step 7 *for n from j-d to j+d // d=1 equal to L*L kernel (W) of size 2*2*
step 8 *net = net + [I (m, n) * W (m, n) + b]*
step 9 *end n*
step 10 *end m*
step 11 *F(net) = Max (net, 0) // Relu activation function*
step 12 *Feature-map [i, j, k] = f(net) // k feature map resulted that equal to the number of filters where each feature map corresponds a specific filter*
step 13 *end j*
step 14 *end i*
step 15 *end k*
End

b) Activation Layer

There are numerous activation functions that are utilized with neural networks, and several of the most widely employed activation methods are Rectified Linear Unit (ReLU), softmax, and sigmoid. Every

convolutional layer's output is transmitted to an activation function layer to convert the linear output into nonlinear. The activation layer is made up of an activation mission that receives a feature map generated via the convolutional layer and produces an activation map just like its output. ReLU-used neural networks train much more quickly than sigmoid and tanh activation functions. Setting the input threshold to zero yields the ReLU activation. In other words, if the input is less than zero, the output of a rectified linear unit is zero; otherwise, the output is the same as the input [43]. The activation function of the ReLU is given by equation (2.4) and is shown in Figure (2.14) [51].

$$ReLU(x) = \max(0, x) \tag{2.4}$$

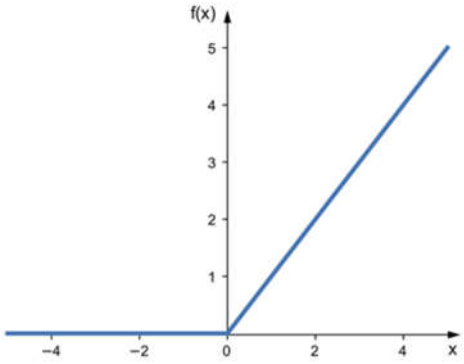


FIGURE 2.14: THE PLOT OF THE RELU ACTIVATION FUNCTION [51]

If the equation (2.3) is substituted, which is the linear output result from the convolution layer, with the ReLU activation function equation (2.4), the outcome is the following equation:

$$C_{i,j}^{k} = max\,(0, \sum_{m=0}^{L-1} \sum_{n=0}^{L-1} W_{m,n}^{k} * I_{i+m,j+n} + b^{k}) \tag{2.5}$$

c) Pooling Layer

ConvNets have an unnecessary pooling or down-sampling layer after the convolution and activation layers to shrink the volume of the

input and, hence, the set of variables in the network. A pooling layer reduces the dimensionality of its input by minimizing the feature maps created by the convolutional layer. There are just a few pooling strategies available, with max-pooling being the most frequent. In the max-pooling method, the pooling layer causes the input feature map to be divided into non-overlapping chunks, and the highest value in each block is simply the output. The max-pooling operation is presented in Figure 2.15 [52].

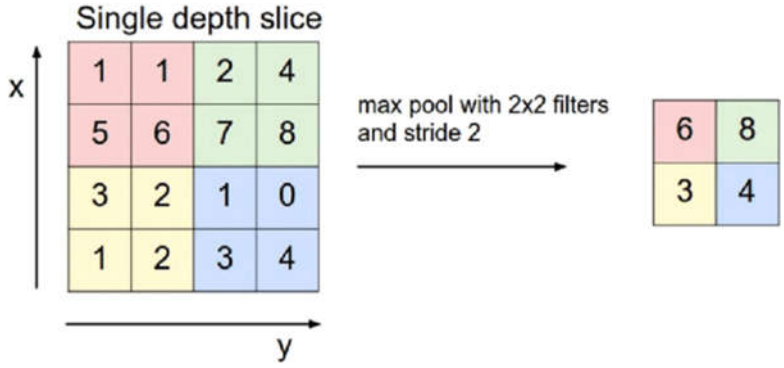


FIGURE 2.15: MAX-POOLING PROCESS ON 4X4 INPUT SIZE [52]

The max pooling layer repeats the same process with each of the previously obtained feature maps from the previous convolution layer. The mathematical form of the max-pooling layer, *P*, on the previously activated feature maps is described in equation (2.6).

$$P_{i,j}^{k} = max \begin{pmatrix} C_{(2i,2j)}^{k} & C_{(2i+1,2j)}^{k} \\ C_{(2i,2j+1)}^{k} & C_{(2i+1,2j+1)}^{k} \end{pmatrix} \tag{2.6}$$

The pooling layer process deletes less significant input while retaining discovered features in a reduced depiction. The main principle underlying the pooling process is that feature recognition is more essential than feature position [43]. The pooling operation is depicted in the algorithm (2.3) [50].

ALGORITHM 2.3: THE POOLING OPERATION	
Input: *Feature map resulted from algorithm 3.5*	
Output: *Down sampled feature map*	
Begin	
Step 1	*k1 = 0 // initial row index of the resulted down-sampled feature map*
Step 2	*for r from 1 to rows of the feature map step 2 // stride 1*
Step 3	*k2 = 0 // initial column index of the resulted down-sampled feature map*
Step 4	*for c from 1 to columns of the feature map step 2 // stride 1*
Step 5	*Max = feature-map (r, c)*
Step 6	*for i from r to r+1*
Step 7	*for j from c to c+1*
Step 8	*if feature-map (i, j) > Max*
Step 9	*Max = feature-map (i, j)*
Step 10	*end if*
	end j
Step 11	*end i*
Step 12	*Down-sampled-fm [k1, k2] = max // the resulted index within the down sampled feature map achieved from the pooling operation*
Step 13	*k2 = k2 + 1*
Step 14	*end c*
Step 15	*k1 = k1 + 1*
Step 16	*end r*
End	

d) Fully Connected Layer

The network's last layer is a fully connected layer (FC), also known as a dense layer. Every neuron in the previous layer is coupled to each node in the following layer in a fully connected layer. The generated feature map of the former layer should be flattened to become a feature vector before being fully connected with the output layer, which consists of neurons based on the number of classes specified by the SoftMax or sigmoid activation tasks for multi and binary classification, in the last CNN layer to classify the trained data. Figure (2.16) depicts the overall relationship between final feature maps and a fully connected layer [43].

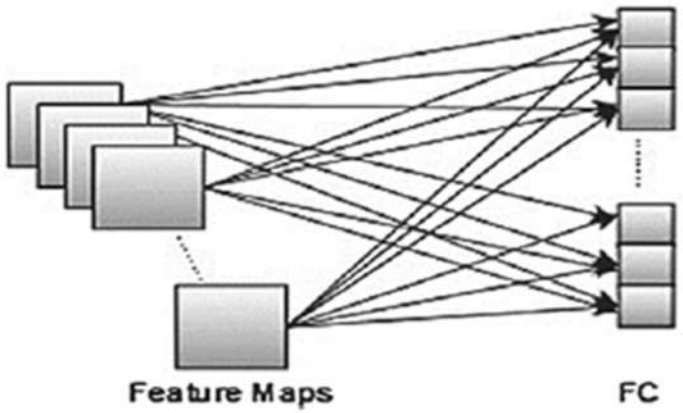


FIGURE 2.16: THE LINK THROUGH THE CONVOLUTION LAYER AND THE FULLY CONNECTED LAYER [43]

Assume the previous layers produced a number of feature maps sorted from 1 to *R*, where *C* are the feature maps created from the previous layers, *k* of *W* are the weights that multiply with the formerly generated feature maps *C* to classify objects. Equation (2.7) depicts the mathematical process of fully connecting *F*, which connects each of the previously generated feature maps with all of the output classes:

$$F^k = \sum_{i=1}^{R} W_i^k * C^i \tag{2.7}$$

The final fully connected layer's output is passed into a classifier that produces class scores. In this research, a sigmoid classifier will be implemented because it is appropriate mathematically for the learning algorithm [43]. The sigmoid activation function, also termed the logistic function, is a well-known activation job for binary classification in neural networks. The input to the function is changed to a value from 0.0 to 1.0. Inputs that are considerably larger than 1.0 are changed to 1.0, whereas values that are much less than 0.0 are converted to 0.0. The application of the sigmoid activation to the previously fully connected output is shown in the following equation:

$$\text{Sigmoid}\,(F^k) = \frac{1}{1 + e^{-F^k}} \tag{2.8}$$

e) Dropout Layer

The training dataset is likely to be overfit when all of the features are linked to the fully connected layer. When a model is trained, it works so well on training data that it has a bad impact on the model's function on new data. This is referred to as overfitting. To address overfitting, the model can include a dropout layer in which some neurons and their links are arbitrarily deleted from the network through training [52]. The dropout operation described in Figure (2.17) and Figure (2.18) depicts the overall architecture of CNN consecutively [53].

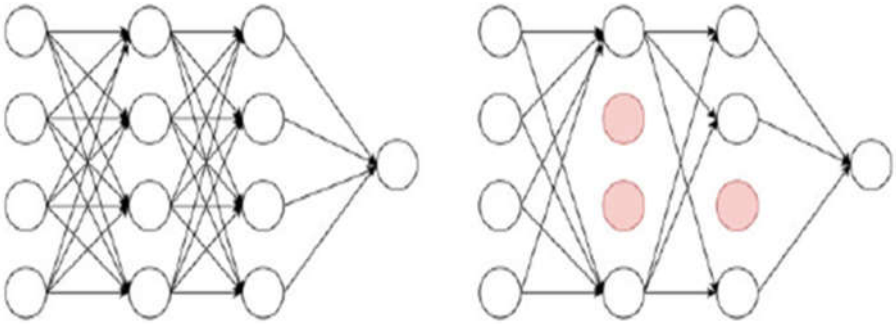


FIGURE 2.17: DROPOUT PROCESS [43]

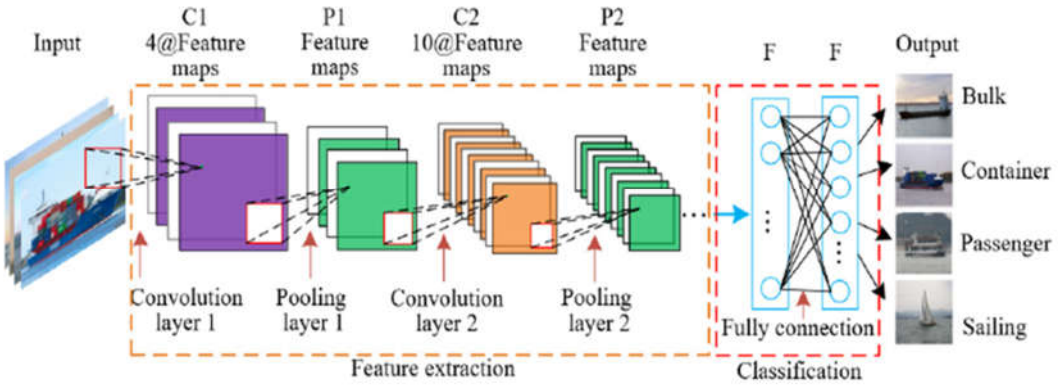


FIGURE 2.18: TYPICAL CONVOLUTIONAL NEURAL NETWORK (CNN) STRUCTURE [53]

2.9. Evaluation the Performance of the Prediction Model

More performance metrics have been utilized to measure the capability of generalization for the trained model and its quality when tested with unlabelled data so as to assess the execution of the prediction model. There are numerous performance measures in the deep learning domain, such as accuracy, loss, precision, and recall.

The most common measure in deep learning and data mining for the evaluation process is the accuracy metric. The accuracy of a set of data is the proportion of test set tuples successfully classified by the classifier. It indicates how accurately the classifier identifies tuples of different classes. The accuracy of a classifier or predictor is generally measured using a confusion matrix. This matrix displays the number of cases predicted either wrongly or correctly by a prediction model (as Table 2.1) [54].

TABLE 2.1: TWO-DIMENSIONAL CONFUSION MATRIX

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

1. True Positive (TP): The positive examples which are correctly classified.
2. False Negative (FN): The positive examples which are wrongly classified.
3. False Positive (FP): The negative examples which are wrongly classified.
4. True Negative (TN): The negative examples which are correctly classified.

Performance metrics utilized in this study are discussed here:

- Accuracy: is the number of right predictions divided by the entire number of predictions. Accuracy can be computed based on the equation below:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{2.9}$$

- Precision: is the percentage of related instances in the collection of received examples. The precision measurement is computed in equation (2.10).

$$Precision = \frac{tp}{tp + fp} \tag{2.10}$$

- Recall: also called sensitivity, it is the percentage of relevant instances that were received. Equation (2.11) clarifies how to compute the recall.

$$Recall = \frac{tp}{tp + fn} \tag{2.11}$$

- F1-score: integrates a classifier's precision and recall into a single measure by calculating their harmonic mean. F1-score can be calculated based on the equation below:

$$F1 - score = \frac{2\,(precision \times recall)}{precision + recall} \tag{2.12}$$

- AUPRC: the area under the precision-recall curve (AUPRC) is a beneficial performance parameter for imbalanced data in a problem where locating positive examples is important. AUPRC is computed from the equation below:

$$AUPRC = \sum_{n} (R_n - R_{n-1}) P_n \tag{2.13}$$

- Loss: during the deep learning model's training procedure, a loss function is applied to compute the error (the space between the prediction and the provided real class label). There are numerous loss functions that can be utilized to measure the error of the final prediction, depending on the model. The loss measure used here is mean squared error (MSE). An MSE is a metric that calculates the quantity of error in statistical models. The mathematical form of the loss function in the equation below:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \qquad (2.14)$$

Where n refers to the volume of samples, $\tilde{y}_i$ is the target (or predicted) label, and y_i is the actual target of the classifier [43].

Chapter Three

THE SYSTEM DESIGN

3.1. Introduction

In this chapter, the steps taken to meet the main goals of this thesis are discussed. It involves designing a model to predict transcription factor binding sites (TFBSs) in a DNA sequence. The architecture of this model is depicted first. Then, data preprocessing is discussed to prepare data for subsequent operations. After that, the model goes through two different paths that include multiple methods such as k-mer embedding, one-hot encoding, z-score motif extraction, and others. Next, the encoding of features into multiple numbers will be explained. Finally, the CNN model and evaluation methods are described subsequently.

3.2. The System Design

The proposed system architecture includes multiple phases for achieving the aim of this thesis. The next sub-sections offer full discussion of these phases, while this section provides a basic overview of them. Firstly, the data preprocessing stage includes three steps: data cleaning, integration, and reduction. Then, the work goes in two ways, each involving three steps. The first way includes the k-mer embedding method, the one-hot encoding method, and connecting each DNA sequence with its TFs. In the second way, there are three operations: k-mer generation, z-score consensus motif finding, and connecting each consensus motif with the existing dataset. The outcomes of these two paths will be input into the encoding phase, which encodes the features. Then, the feature data will be split into training and testing sets. The training data will be entered into a convolutional neural network (CNN) to build the prediction model. While the testing data is transmitted to the evaluation phase to assess the predicted class via different measures. The general architecture of the suggested system is visualized in Figure 3.1.

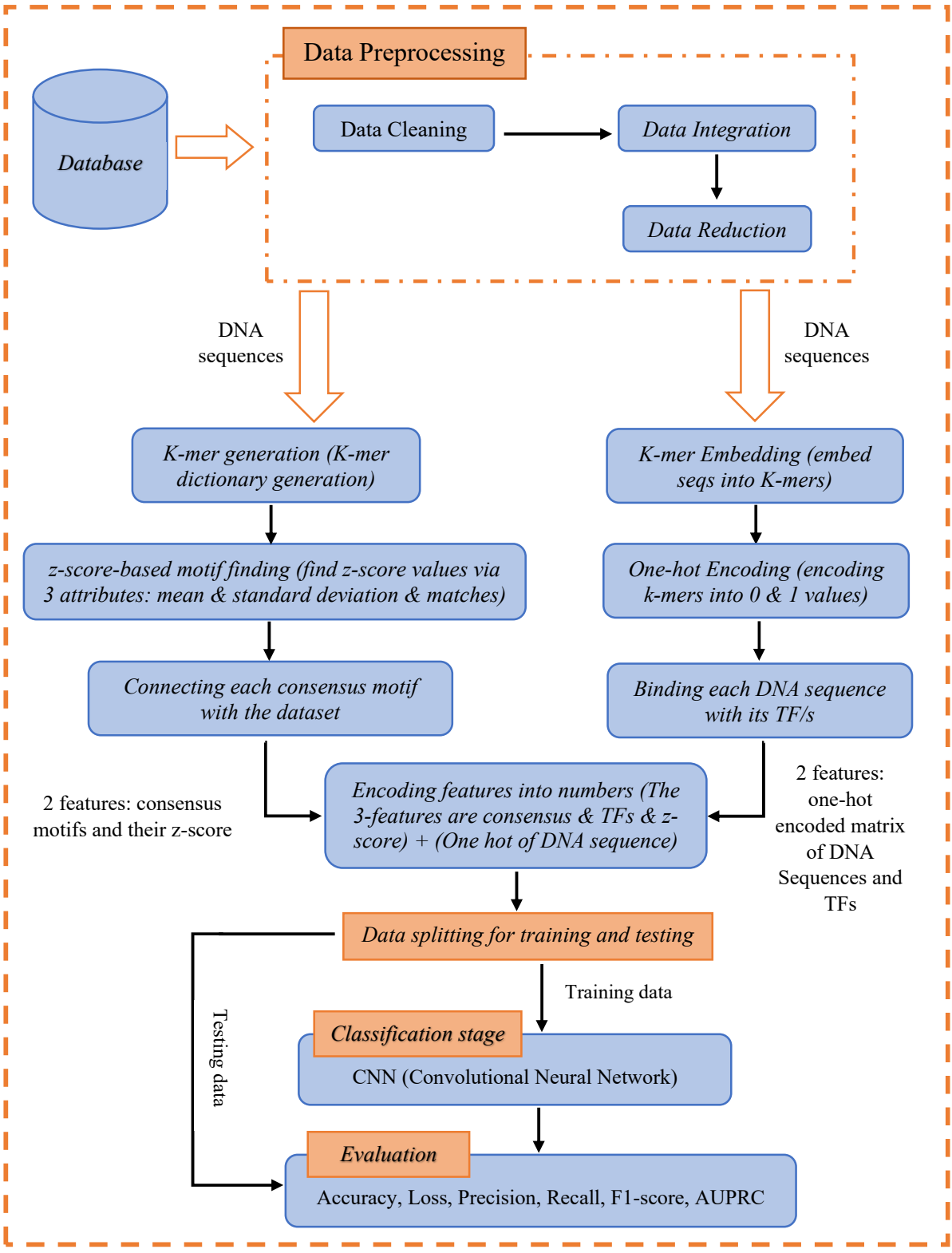


FIGURE 3.1: THE PROPOSED SYSTEM

3.2.1. Dataset

The Arabidopsis Gene Regulatory Information Server (AGRIS) is a great reference for gene regulatory research in Arabidopsis thaliana, the model plant. AtTFDB, AtcisDB, and AtRegNet are three interconnected databases that provide extensive and up-to-date information on transcription factors (TFs), predicted and practically proven cis-regulatory elements (CREs), and their relationships, respectively. Of the three databases mentioned, the focus will only be on three files that will enter the preprocessing stage, which are PromoterSeq.csv, BindingSite.csv, and Bindingsite_data .csv.

3.2.2. Data Preprocessing

Preprocessing is the initial stage of the proposed model, which aims to convert the raw dataset into a simple and efficient format. As a result, it is an important procedure with the main vision of making a dataset that may be considered reliable and useful for machine learning methods (prediction algorithms). The data preprocessing phase consists of three significant steps:

3.2.2.1. Data Cleaning

In this step, a missing value is an empty letter in a DNA sequence, sometimes known as a "gap" between two letters or two sequences. On the other hand, there are letters that are not labelled as having irrelative value (A, G, C, T). Therefore, these errors caused by a lack of dataset values will be handled by using a handling missing value algorithm that is depicted in Algorithm (2.1). Data cleaning involves other steps that will be implemented on the AGRIS database, such as removing duplicates and deleting unwanted data.

3.2.2.2. Data Integration

This step involves merging data from multiple data sources and aggregating them into one representation form. So, the primary key attributes are connected in one table with the foreign keys of the other tables. The goal of this step is to combine features that help in the prediction task into one table file. Figure 3.2 describes the relationships between AtcisDB database attributes, and Figure 3.3 shows the connection between AtTFDB database attributes.

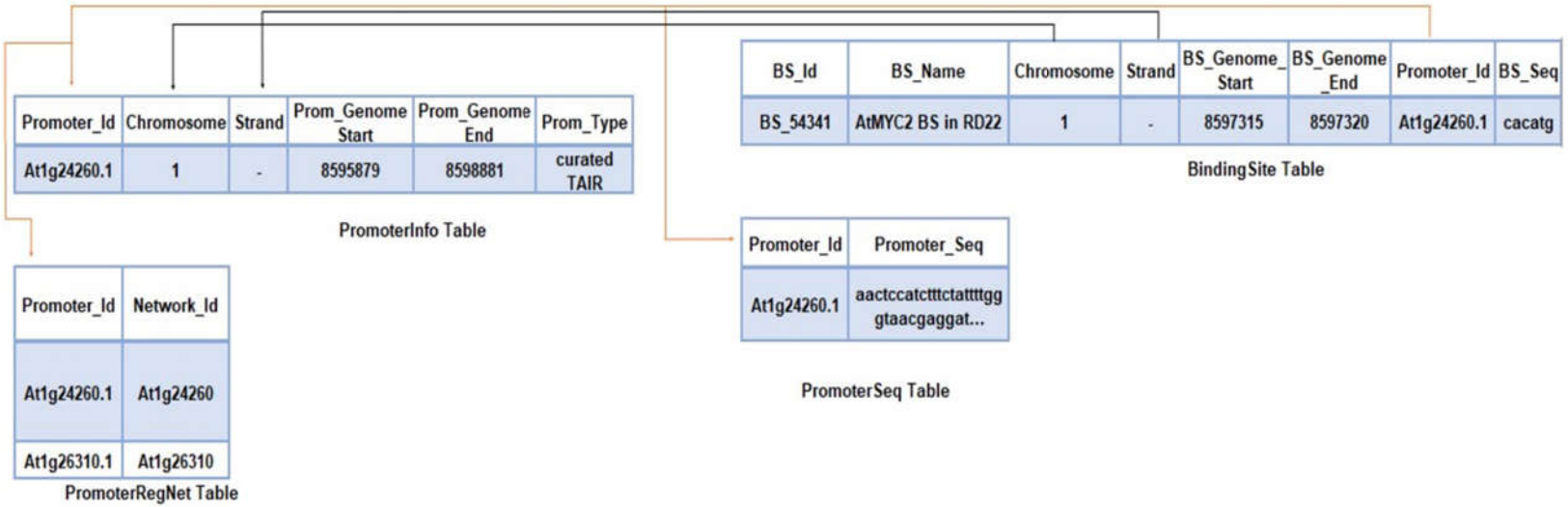


FIGURE 3.2: RELATIONSHIPS BETWEEN ATCISDB DATABASE FILES

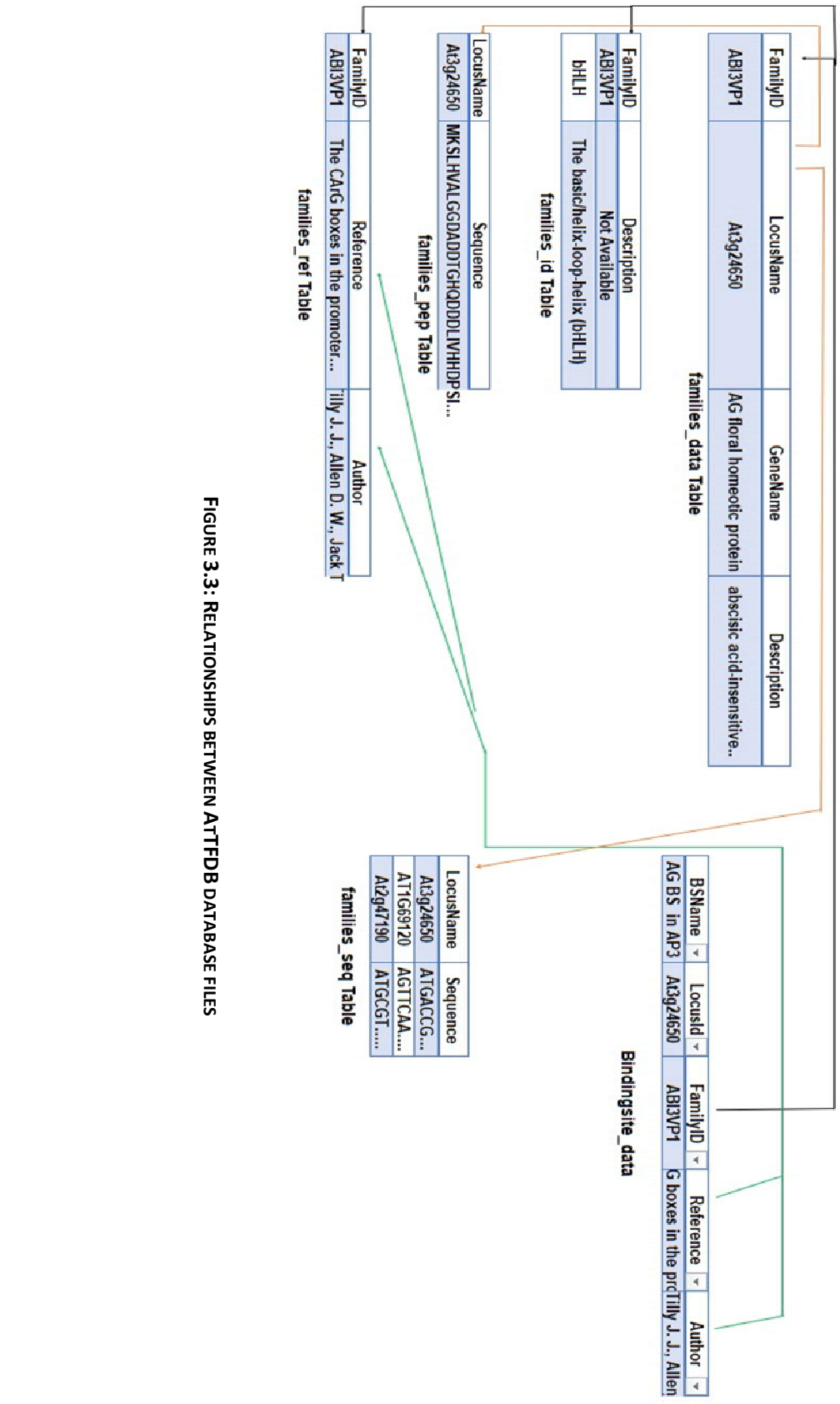


FIGURE 3.3: RELATIONSHIPS BETWEEN ATTFDB DATABASE FILES

3.2.2.3. Data Reduction

This step complements the previous one by depending on primary keys to minimize the size of the AGRIS database. Additionally, in data reduction, data attributes (columns) that do not indicate information regarding DNA-binding sites or description of transcription factors will be deleted. Therefore, the database named "AtRegNet" that contains a single file will not be used. The goal of this step is to reduce the dimensionality of the dataset and therefore prevent overfitting during the CNN training phase.

3.2.3. Computing One-hot method

3.2.3.1. K-mer embedding

The purpose of this step is to convert the DNA sequence that resulted from the previous phase into k-mers. Before that, the length of the k-mer (k) and the number of sequences (n) must be determined. The DNA sequences that will be input in this step are from the PromoterSeq file. The process of k-mer embedding is shown in Algorithm (3.1).

ALGORITHM 3.1: K-MER EMBEDDING
Input: *L,k,n: where L is length of DNA-sequence and k is length of k-mer (word) and n is number of sequence DNA sequences that resulted from the data preprocessing stage*
Output: *2D array1[m.n] where m is no. of k-mer and n is no. of sequences*
Begin
step 1 k-mers(seq_{bj}:string, k:integer)
step 2 L ← length(seq)
step 3 m = 0
step 4 for b = 1 to n
step 5 for j = 1 to L
step 6 k-mers(seq_{bj}, k)
step 7 m++
step 8 if m < L – k + 1 then
step 9 return to step 4
step 10 else
step 11 array1 [m, n] ← k-mers(DNA sequences, k=3)
step 12 end if
step 13 end j
step 14 end b
step 15 return array1
End

3.2.3.2. One – hot Encoding

The embedded k-mers will be transformed into one hot binary matrix. As it was explained in the previous chapter, each position contains four states, so if the nucleotide matches the state, it takes the number 1, while the others are assigned to zero. The aim of the one-hot encoding step is to represent the DNA sequence data in an appropriate form before feeding it as input to the CNN neural network. The Algorithm (3.2) depicts the steps of one hot encoding method.

ALGORITHM 3.2: ONE HOT ENCODING
Input: *Two dimensional array1 [m,n] Where m is no. of embedded k-mers and n is no. of input sequences //Output of algorithm (3.1)*
Output: *One-hot encoding matrix (E)*
Begin
step 1 for each input sequence
step 2 for each k-mer in sequence
step 3 for each char in k-mer
step 4 if the column number correspond to char
step 5 assign char to 1 and other chars in k-mer assign to 0
step 6 end if
step 7 end for
step 8 end for
step 9 end for
step 10 store all char encoding in the array E
step 11 return E
End

3.2.3.3. Binding each DNA sequence with its TF/s

In this step, the DNA sequences that entered the k-mer embedding step will be connected with their transcription factor(s). The number of transcription factors is different from one sequence to another. Some sequences have 8 TFs, others have 20 TFs. So, this step is significant to extract the TFs feature before the CNN classification step.

3.2.4. Consensus Motif Finding

3.2.4.1. K-mer generation

In this step, a k-mer dictionary will be generated by assigning a k-length to mers. The number of generated words is calculated by 4^k. This step is important because it comes before the z-score motif method, which is calculated for each k-mer result from the generation step.

3.2.4.2. Z-score-based Motif finding

The DNA sequences that resulted from the data preprocessing phase will be the input to the z-score motif method step. The goal of this step is to calculate z-score values for each k-mer from the previous

generation step. The Algorithm (3.3) explains how to calculate the z-score function depending on equation (2.2).

ALGORITHM 3.3: Z-SCORE-BASED MOTIF FINDING	
***Input**: L, n: where L is length of DNA-sequence and n is number of sequences, DNA sequences that result from the data preprocessing stage*	
***Output**: ranking z-score for all k-mers*	
Begin	
Step 1	*Divide (n) sequences into two groups: positive sequence (p) & negative sequence (g)*
Step 2	*n_p = number of positive sequences*
Step 3	*n_g = number of negative sequences*
Step 4	*generate k-mers dictionary with specified (k) length*
Step 5	*for each k-mer in dictionary*
Step 6	*compute matches_p that refers to number of occurrences of k-mer in positive sequences*
Step 7	*compute mean_p, depending on three variables (probability of k-mer, matches_p, n_p)*
Step 8	*compute standard deviation (std_p) by equation:* $std\_p = \sqrt{mean\_p}$
Step 9	*compute matches_n that refers to number of occurrences of k-mer in negative sequences*
Step 10	*compute mean_n, depending on three variables (probability of k-mer, matches_n, n_g)*
Step 11	*compute standard deviation (std_n) by equation:* $std\_n = \sqrt{mean\_n}$
Step 12	*compute z_p (z-score for positive sequence)* $= \frac{matches\_p - mean\_p}{std\_p}$
Step 13	*compute z_n (z-score for negative sequence)* $= \frac{matches\_n - mean\_n}{std\_n}$
Step 14	*compute ranking z-score = z_p – z_n*
End	

3.2.4.3. Connecting each Consensus Motif with the Dataset

In this step, the k-mers for which the z-score value is calculated will be connected with the database. The k-mer that matches some or all of the consensus motifs that exist in the motif attribute of the Bindingsite_data.csv file will be returned as an outcome of this step. In addition, the results will be saved in a specific table that contains two features: the consensus motif and its z-score value.

3.2.5. Encoding Features

In the feature encoding phase, each feature will be encoded into a real number. The input for this phase consists of four features that are resulted from the previous steps. From one-hot encoding's path, two attributes will be obtained: DNA sequences (encoded by one-hot encoding, so this feature is not needed for the encoding step) and their TFs. For the consensus motif's path, the result shows two features, which are the consensus motif and its z-score.

3.2.6. CNN Classification

The features dataset will be divided into two parts before implementing the prediction model, known as the training set and the testing set. The percentage of training data is seventy percent, while thirty percent is the ratio of testing data from the overall dataset. In general, establishing the prediction model is the most critical stage in the suggested system. The Convolutional Neural Network (CNN) model is employed in this thesis because it is simple to interpret while still being particularly capable of dealing with the datasets. It is also capable of achieving the best prediction performance. The CNN model is used with carefully chosen input data because of its proven usefulness in providing satisfying results and improving prediction performance. As a result, the suggested system is designed around the CNN model for class label prediction. Table 3.1 depicts the CNN structure as follows:

TABLE 3.1: THE PROPOSED STRUCTURE OF THE CNN SYSTEM

Layer	Layer information	Output shape	Params number
conv2d_1 (Conv2D)	No. of filters = 32, filter size (2,2)	(None, 2, 4679, 32)	160
max-pooling2d_1 (MaxPooling 2D)	Pool size (2,2), strides (2,2), padding =’same’	(None, 1, 2340, 32)	0
conv2d_2 (Conv2D)	No. of filters = 64, filter size (3,3), activation=’relu’	(None, 1, 2340, 64)	18496
max-pooling2d_2 (MaxPooling 2D)	Pool size (3,3), strides (3,3), padding =’same’	(None, 1, 780, 64)	0
Conv2d_3 (Conv2D)	No. of filters = 64, filter size (3,3), activation=’relu’	(None, 1, 780, 64)	36928
max-pooling2d_3 (MaxPooling 2D)	Pool size (3,3), strides (3,3), padding =’same’	(None, 1, 260, 64)	0
Conv2d_4 (Conv2D)	No. of filters = 128, filter size (5,5)	(None, 1, 260, 128)	204800
max-pooling2d_4 (MaxPooling 2D)	Pool size (5,5), strides (5,5), padding =’same’	(None, 1, 52, 128)	0
Flatten	/	(None, 6656)	0
dropout_1 (Dropout)	0.8	(None, 6656)	0
dense_1 (Dense)	1024, activation=’relu’	(None, 1024)	6816768
dropout_2 (Dropout)	0.6	(None, 1024)	0
dense_2 (Dense)	512, activation=’relu’	(None, 512)	524800
dropout_3 (Dropout)	0.6	(None, 512)	0
dense_3 (Dense)	1, activation=’sigmoid’	(None, 1)	513

The convolution process has been conducted between the input and filter from left to right, as indicated in Figure (2.11) of the input and output feature maps provided in the previous chapter. Assume the input dataset I of i, j indices received k weight filters W of size $L \times L$ of m and n indices, and its receptive field is also $L \times L$ of indices $i + m, j + n$, and it travels in S stride. The convolution process is illustrated in the algorithm (2.2). As a result, the obtained feature map is the result of the convolution

layer being sent through the pooling layer to be down-sampled. The Algorithm (2.3) displays the pooling algorithm.

CNN is regarded as a black box that is designed to extract features based on pre-specified parameters and is proposed to predict DNA binding sites in order to provide the desirable outcome. The construction of the CNN model utilized is depicted in Figure 3.4.

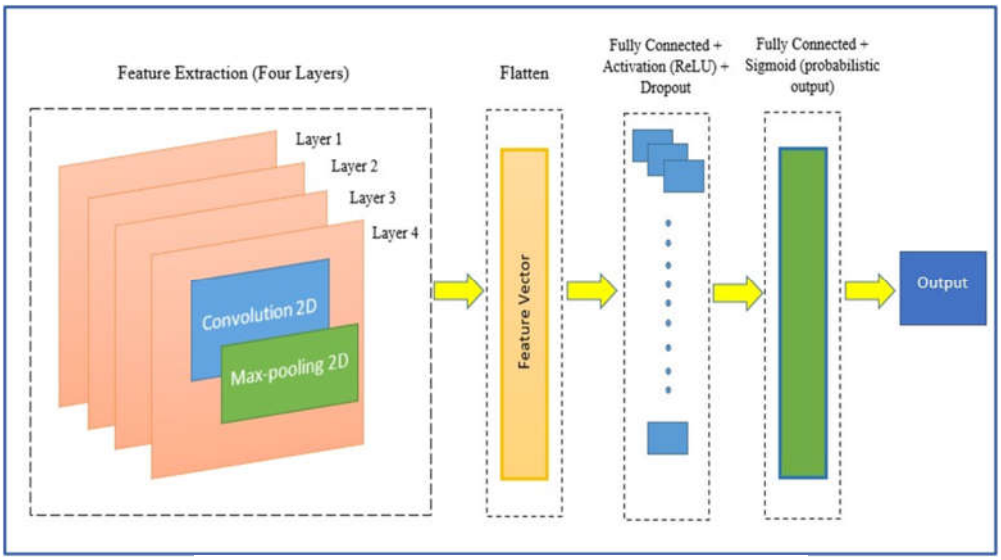


FIGURE 3.4: ARCHITECTURE OF THE USED CNN MODEL

3.2.7. Model Evaluation

In this phase, the accuracy of the CNN model is evaluated depending on the testing set. To provide a more reliable assessment, the model's performance is evaluated utilizing the training and testing datasets. The model is trained for a period of time while the loss is monitored in the training and test datasets as well as the accuracy measure. Every epoch, the model produces the training loss and accuracy as well as the testing loss and accuracy. Actually, there are several other metrics, such as precision, recall, F1-score, and AUPRC (all the measures equations in Chapter 2). These measurements will support the model's performance and ensure the result's validation.

Chapter Four

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction

This chapter offers the outcomes of experiments of the suggested system for prediction of DNA binding sites. The proposed method depends on three sets of databases, which are (AtTFDB, AtcisDB, and AtRegNet). The goal of this method is to predict protein DNA binding affinity to DNA probes by utilizing an effective deep learning technique, which is a convolutional neural network. Following that, the results will be analyzed and compared with the performance of the different models. The capability to discover transcription factor binding locations or motifs in the genome is used to assess performance. To evaluate the results, some metrics, such as accuracy and loss, are commonly employed in machine learning and motif discovering evaluation. In addition, other standard measurements (precision, recall, F-measure, and AUPRC) are also used.

4.2 Experimental Settings

The proposed model tests have been carried out on a laptop Lenovo core i5 RAM 4G SDD Hard Python language 3.8 versions using the Jupyter Notebook program on Windows 10 operating system. The steps and algorithms have been executed more than once in order to achieve the ideal result for each algorithm.

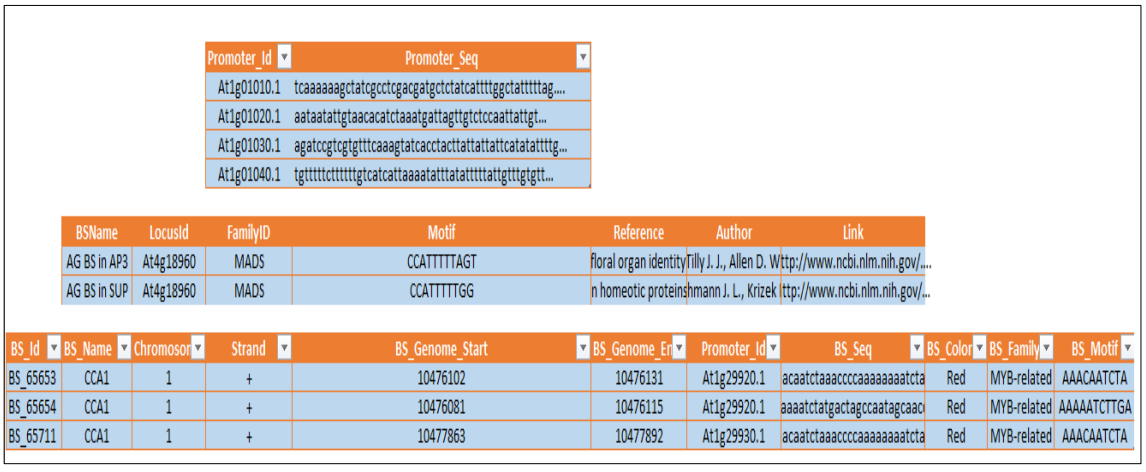
4.3 Dataset Description

The database included in this study has been obtained from the Arabidopsis Gene Regulatory Information Server (AGRIS; http://arabidopsis.med.ohio-state.edu/). It was released on March 11, 2019 by a number of researchers, including Dr. Erich Grotewold, Wilberforce Ouma, Eric Maina, and Fabio Gomez. This database contains 3 accessible datasets (AtTFDB, AtcisDB, and AtRegNet). The Arabidopsis Transcription Factor Database (AtTFDB) gives ideas about

1773 transcription factors (TFs) that have been classified into 50 types depending on the existence of similar domains, which usually participate in DNA-binding operations. The AtcisDB contains 3000 bp promoter areas of genes and was retrieved from TAIR9 as in the latest version of AGRIS, yielding a total of 33,239 sequences. The gene regulatory network (AtRegNet) Database, on the other hand, has 8100 nodes, 814 TF and 7286 non-TF encoding genes, and is linked via 11,123 edges [27].

4.4 The Results of Data Preprocessing

Preprocessing is an important stage that has been conducted on the AGRIS dataset to avoid large value variances that dominate the results. As already mentioned in Chapter 3, data cleaning, integration, and reduction are significant steps to minimize dimensionality and therefore avoid the main reason for overfitting. Figure 4.1 presents three data tables before the data preprocessing phase. While Figure 4.2 shows the dataset that is used after the data preprocessing operations.

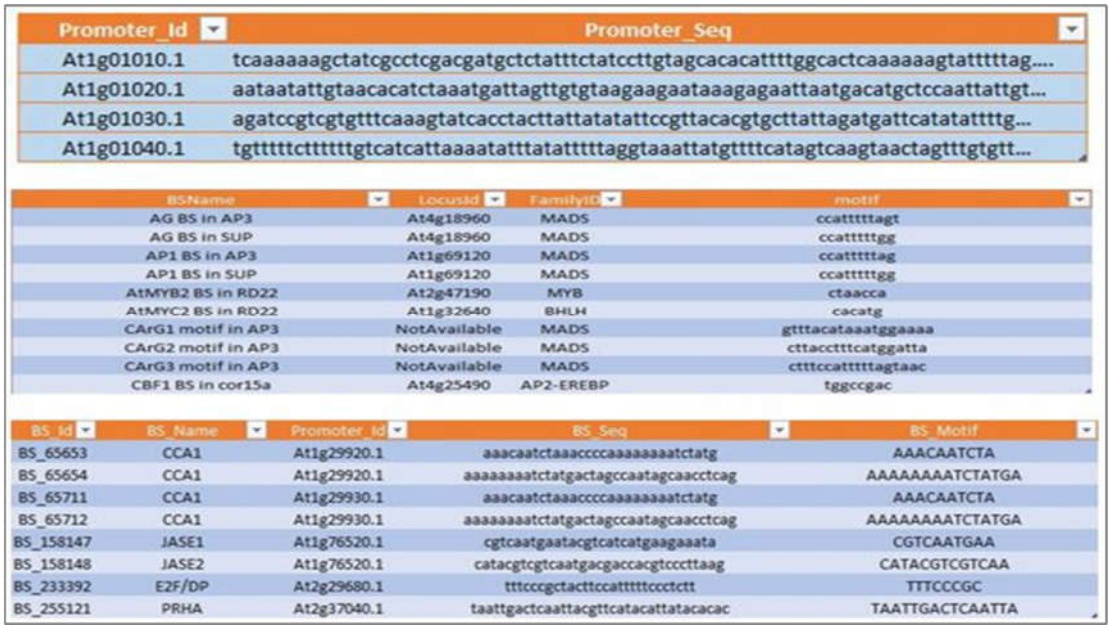


Promoter_Id	Promoter_Seq
At1g01010.1	tcaaaaaagctatcgcctcgacgatgctctatcattttggctatttttag....
At1g01020.1	aataatattgtaacacatctaaatgattagttgtctccaattattgt...
At1g01030.1	agatccgtcgtgtttcaaagtatcacctacttattattattcatatattttg...
At1g01040.1	tgttttttcttttttgtcatcattaaaatatttatatttttattgtttgtgtt...

BSName	LocusId	FamilyID	Motif	Reference	Author	Link
AG BS in AP3	At4g18960	MADS	CCATTTTTAGT	floral organ identity	Tilly J. J., Allen D. W	ttp://www.ncbi.nlm.nih.gov/....
AG BS in SUP	At4g18960	MADS	CCATTTTTGG	n homeotic proteins	hmann J. L., Krizek	ttp://www.ncbi.nlm.nih.gov/...

BS_Id	BS_Name	Chromosor	Strand	BS_Genome_Start	BS_Genome_En	Promoter_Id	BS_Seq	BS_Color	BS_Family	BS_Motif
BS_65653	CCA1	1	+	10476102	10476131	At1g29920.1	acaatctaaaccccaaaaaaaatcta	Red	MYB-related	AAACAATCTA
BS_65654	CCA1	1	+	10476081	10476115	At1g29920.1	aaaatctatgactagccaatagcaac	Red	MYB-related	AAAAATCTTGA
BS_65711	CCA1	1	+	10477863	10477892	At1g29930.1	acaatctaaaccccaaaaaaaatcta	Red	MYB-related	AAACAATCTA

FIGURE 4.1: THE DATASET BEFORE THE PREPROCESSING STAGE



Promoter_Id	Promoter_Seq
At1g01010.1	tcaaaaaagctatcgcctcgacgatgctctatttctatccttgtagcacacatttttggcactcaaaaaagtatttttag....
At1g01020.1	aataatattgtaacacatctaaatgattagttgtgtaagaagaataaagagaattaatgacatgctccaattattgt...
At1g01030.1	agatccgtcgtgtttcaaagtatcacctacttattatatattccgttacacgtgcttattagatgattcatatattttg...
At1g01040.1	tgtttttctttttttgtcatcattaaaatatttatatttttaggtaaattatgttttcatagtcaagtaactagtttgtgtt...

BSName	LocusId	FamilyID	motif
AG BS in AP3	At4g18960	MADS	ccatttttagt
AG BS in SUP	At4g18960	MADS	ccatttttgg
AP1 BS in AP3	At1g69120	MADS	ccatttttag
AP1 BS in SUP	At1g69120	MADS	ccatttttgg
AtMYB2 BS in RD22	At2g47190	MYB	ctaacca
AtMYC2 BS in RD22	At1g32640	BHLH	cacatg
CArG1 motif in AP3	NotAvailable	MADS	gtttacataaatggaaaa
CArG2 motif in AP3	NotAvailable	MADS	cttacctttcatggatta
CArG3 motif in AP3	NotAvailable	MADS	ctttccatttttagtaac
CBF1 BS in cor15a	At4g25490	AP2-EREBP	tggccgac

BS_Id	BS_Name	Promoter_Id	BS_Seq	BS_Motif
BS_65653	CCA1	At1g29920.1	aaacaatctaaaccccaaaaaaaatctatg	AAACAATCTA
BS_65654	CCA1	At1g29920.1	aaaaaaaatctatgactagccaatagcaacctcag	AAAAAAAATCTATGA
BS_65711	CCA1	At1g29930.1	aaacaatctaaaccccaaaaaaaatctatg	AAACAATCTA
BS_65712	CCA1	At1g29930.1	aaaaaaaatctatgactagccaatagcaacctcag	AAAAAAAATCTATGA
BS_158147	JASE1	At1g76520.1	cgtcaatgaatacgtcatcatgaagaaata	CGTCAATGAA
BS_158148	JASE2	At1g76520.1	catacgtcgtcaatgacgaccacgtcccttaag	CATACGTCGTCAA
BS_233392	E2F/DP	At2g29680.1	tttcccgctacttccatttttccctctt	TTTCCCGC
BS_255121	PRHA	At2g37040.1	taattgactcaattacgttcatacattatacacac	TAATTGACTCAATTA

FIGURE 4.2: DATASET AS A RESULT OF THE DATA PREPROCESSING PHASE

4.5 The Results of Computing One-hot method

4.5.1 K-mer Embedding

Here, the result of the *k*-mer embedding process will be displayed. The input size is limited to the first 4000 DNA sequences from the PromoterSeq.csv file with a length of 200. The size of the words depends on the size of the value of k, with length = 6. Therefore, each sequence will be split into 195 words, as obtained by the rule $[(200 - 6\,/1] + 1 = 195$. Table 4.1 shows the result of the 6-mer method.

TABLE 4.1: SPLITTING A DNA SEQUENCE RESULTS IN A 6-MER

Words (k = 6)							
No. of record	**No. of Word**						
	0	1	2	…	192	193	194
0	tcaaaa	caaaaa	aaaaaa	…	tatcca	atccat	tccatt
1	agatcc	gatccg	atccgt	…	ttttct	tttctt	ttcttg
2	tgtttt	gttttt	ttttc	…	ggaata	gaataa	aataaa
3	gcatat	catatt	atattt	…	ttttgc	tttgca	ttgcag
4	cgcagt	gcagtc	cagtca	…	aagtct	agtctt	gtcttc
…	…	…	…	…	…	…	…
3995	ttatgt	tatgtt	atgtta	…	ttacac	tacaca	acacat
3996	agtgct	gtgctg	tgctgt	…	caaaat	aaaatt	aaatta
3997	tttagt	ttagtt	tagttg	…	gcatgt	catgtg	atgtga
3998	aaaaag	aaaagc	aaagcg	…	atatga	tatgaa	atgaat
3999	tgacac	gacact	acactt	…	taattc	aattca	attcaa

4.5.2 One-hot Encoding

In this phase, each 6-mer word is turned into a $4 \times L$ one-hot binary array (L is the sequence length). One-hot vectors are used to continuously modify the parameters of kernel matrices by learning those vectors. Figure 4.3 presents an example of a one-hot matrix from this work.

```
[0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0,
0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0,
1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0,
0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0,
0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0,
0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0,
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0,
0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0,
0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0,
0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0,
```

FIGURE 4.3: ONE-HOT ENCODING MATRIX

4.5.3 Binding each DNA sequence with its TF/s

In this step, the DNA sequence that entered the k-mer embedding step will be connected with its transcription factor/s. This occurred by matching the Promoter_Id attribute in the PromoterSeq.csv file with the Promoter_Id attribute in the BindingSite.csv file and returning BS_Name attribute values that represent the name of the TF/s of each Promoter_Id. The result is a table consisting of two attributes, which are Promoter_Id and TFs, as shown in Table (4.2).

TABLE 4.2: THE RESULT OF CONNECTING DNA SEQUENCE WITH ITS

Promoter_Id	TFs
At1g01010.1	SORLIP1
At1g01010.1	G-box
At1g01010.1	ATHB2
At1g01010.1	BoxII
At1g01010.1	CCA1
At1g01010.1	MYB4
At1g01010.1	GATA
...	...
At2g21610.1	MYB
At2g21610.1	GATA
At2g21610.1	ATHB2
At2g21610.1	DPBF1&2
At2g21610.1	AtMYC2
At2g21610.1	LFY
At2g21610.1	L1-box

4.6 The Results of Consensus Motif Finding

4.6.1 K-mer Generation

The aim of this step is to generate a set of k-mers (also called "bag of words") with a k length of 6 nucleotide characters. It is important to remember that k-mers for larger values of k are also impacted by the factors that affect lower values of k. The Figure below represents the result of this step.

['aaaaac', 'aaaaag', 'aaaaat', 'aaaacc', 'aaaacg', 'aaaaga', 'aaaagt', 'aaacac',
'aaacag', 'aaacat', 'aaacca', 'aaaccc', 'aaaccg', 'aaacga', 'aaacgc', 'aaacgt',
'aaactc', 'aaactt', 'aaagaa', 'aaagac', 'aaagag', 'aaagcc', 'aaagct', 'aaaggg',
'aaagtc', 'aaataa', 'aaatac', 'aaatcc', 'aaatcg', 'aaatga', 'aaatgc', 'aaatta',
'aagagt', 'aagatg', 'aagatt', 'aagcca', 'aagccc', 'aagcct', 'aagcga', 'aagcgc',
'aagcta', 'aaggaa', 'aaggag', 'aaggcc', 'aaggga', 'aagggc', 'aaggggg', 'aagggt',
'aagtca', 'aagtct', 'aagtga', 'aagttt', 'aataaa', 'aataac', 'aatacg', 'aatact',
'aataga', 'aatcag', 'aatcat', 'aatcca', 'aatccg', 'aatcga', 'aatcgc', 'aatcgt',
.....
.....
'tttcct', 'tttcga', 'tttcgc', 'tttcgg', 'tttcgt', 'tttctg', 'tttgaa', 'tttgag', 'tttgat',
'tttgca', 'tttgcg', 'tttgct', 'tttggt', 'tttgta', 'tttgtg', 'tttgtt', 'ttttaa', 'ttttag', 'ttttat',
'ttttca', 'ttttcc', 'ttttcg', 'ttttga', 'ttttgg', 'ttttgt', 'tttta', 'ttttg', 'ttttt']

FIGURE 4.4: SAMPLE FROM K-MER GENERATION

4.6.2 Z-score-based Motif Finding

To calculate the z-score motif method, the DNA sequence will be read from the PromoterSeq file. The sequences should be split into positive and negative sequence sets. And then computes z-score variables (mean, standard deviation, and matches) for each set. The matches are the number of matches between the k-mer resulting from the previous step and the DNA sequences. Finally, a z-score from a positive set will be associated with a z-score from a negative group to yield a ranking z-score. The z-score method's results are shown in Figure 4.5.

[-1.3241500844943914, 1.0755531137161825, 0.40832317673813634, 0.35648348432167865,
-0.4654189125943944, .23499730802500807, 0.9930265469236446, -0.40911751709950295,
1.1285813931098048, -0.07479327280852033, -0.11691502778379004, 0.01850259768607998,
-0.8088905093220404, -0.3933134168208383, -0.21265545988308787, -0.09479037448855365,
-0.5934258818245155, 0.816099241581437, 1.0135915002145843, 1.0426540014090548,
.....
.....
2.6382165382866845, 0.19970929297990025, -0.11114862773964518, 0.9640070101956546,
1.907355811795938, -0.19105933553587562, -1.1713640596131398, -1.4311426719569695,
2.532856292901215, 0.14412192173922733, -0.13675088978176575, 0.8236764164963475,
1.954350763771906, 2.6442814272641684, -0.6125662545322328, 1.5448541240153162,
2.1248777234238077, 0.6261354476973509, 0.826444409509115, 1.2195326809486815,
-1.2684100699685743, 2.2379020553536506, -0.2883712268407095, 0.6716039006642447,
1.0709192263048166, 2.771719524415147, 0.0, 2.4090592707490472, 3.722439082302685]

FIGURE 4.5: THE RESULT OF Z-SCORE CONSENSUS MOTIF METHOD

4.6.3 Connecting each Consensus Motif with the Dataset

Through this phase, each consensus motif generated by the z-Score method will be linked to the current dataset. Firstly, the consensus motif will be matched to the motif attribute in the Bindingsite_data.csv file (as shown in Figure 4.6. Then, the matching consensus motifs and their z-score will be connected with the DNA sequences that contain these motifs. The result of this step is the 3 features saved in table 4.3.

Consensus motifs

['aaaaca', 'aaacag', 'aacaga', 'aatagg', 'acagaa', 'agaata', 'aggaaa', 'atagga', 'cagaat', 'gaatag', 'taggaa',
'tttagt', 'ttttag', 'ttttgg', 'ttttta', 'tttttg', 'ctaacc', 'taacca', 'cacatg', 'gccgac', 'ggccga', 'tggccg', 'aaaaaa',
.....
.....
'attaaa', 'attagt', 'ctaatt', 'taattt', 'aattta', 'atttaa', 'caattt', 'ccaatt', 'taatgg', 'ttaatg', 'tttaat', 'actcat']

Z-score for each motif

[-1.3241500844943914, 1.0755531137161825, 0.40832317673813634, 0.35648348432167865,
-0.4654189125943944, 0.23499730802500807, 0.9930265469236446, -0.40911751709950295,
1.1285813931098048, -0.07479327280852033, -0.11691502778379004, 0.01850259768607998,
-0.8088905093220404, -0.3933134168208383, -0.21265545988308787, -0.09479037448855365,
.....
.....
-0.5076999308213317, 0.4619127533019878, -1.034321600324624, -0.3598138899939016,
0.5629793323939047, -0.6793820645491877, -1.2334599922066225, 0.7989606931263218,
1.6139763580357567, -1.2743019769531259, -1.1100716634060426, -0.2665865327922958]

FIGURE 4.6: MATCHING CONSENSUS MOTIFS WITH A DATASET

TABLE 4.3: THE 3-FEAURES RESULTED FROM THE CONNECTING MOTIF STEP

DNA Sequence	Consensus motif	z-score
tcaaaaaagctatcgcctcgacga…..	ttcaag	0.688365599
tcaaaaaagctatcgcctcgacga…..	actcaa	0.277554936
tcaaaaaagctatcgcctcgacga…..	gattca	0.18826317
tcaaaaaagctatcgcctcgacga…..	attcaa	0.166972355
tcaaaaaagctatcgcctcgacga…..	aaaaaa	0.063866926
tcaaaaaagctatcgcctcgacga…..	ttttta	-0.21265546
…	**…**	**…**
tgacacttctttttcttggataaa….	ttttta	-0.363352681
tgacacttctttttcttggataaa….	ttatta	-0.636917368
tgacacttctttttcttggataaa….	taattg	-0.742839014
tgacacttctttttcttggataaa….	aaatct	-0.808890509
tgacacttctttttcttggataaa….	taacta	-1.274301977

4.7 The Results of Encoding features

In this stage, all the TFs attribute values from the connect tf.csv file will be encoded depending on the LabelEncoder library. This library converts any string into a real number. Before that, each transcription factor that belonged to the same Promoter_Id should be put into one sublist in the general list. Then, all values in each sublist will be merged into one value and converted into a float type. It is worth noting that padding will be used in each sublist by adding huge values of the number (1) for each sublist. In addition, cons (consensus motif) and z-score attributes are saved in the motifseq5.csv file and will be encoded into numbers in order to enter them into CNN by the same method of encoding TFs. The results of this step are depicted in Figures 4.7 and 4.8.

```
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2.0, 4.0, 1.0, 1.0, 2.0, 2.0, 9.0, 2.0, 1.0, 2.0, 7.0, 1.0, 4.0, 2.0, 1.0, 6.0,
3.0, 1.0, 6.0, 1.0, 0.0, 2.0, 3.0, 1.0, 5.0, 1.0, 3.0, 1.0, 8.0, 2.0, 0.0, 1.0, 0.0, 4.0, 8.0, 5.0, 1.0, 7.0, 1.0, 9.0]
```

FIGURE 4.7: THE RESULTS OF ENCODING TFS ATTRIBUTE

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 3.0, 0.0, 1.0, 4.0, 1.0, 4.0, 7.0, 1.0, 3.0, 3.0, 1.0, 4.0, 3.0, 8.0, 8.0, 1.0, 7.0, 1.0, 4.0, 2.0, 1.0]
```

FIGURE 4.8: THE RESULTS OF ENCODING CONS ATTRIBUTE

4.8 Results of CNN Classification

In this phase, the results of using a CNN neural network model to perform prediction of transcription factor-binding sites (TFBS) in the DNA will be presented and discussed. To study the performance of the suggested model, 70% of the dataset is used as the training group and 30% as the testing. Moreover, the SGD algorithm (which includes learning rate and momentum) is added to improve the overall score. In principle, the CNN model used in this research is developed using the Keras Python library. Keras is a small and simple Python deep learning toolbox that can be placed on top of TensorFlow (Theano or CNTK). It enables designers

to concentrate on principles of deep learning, like designing CNN layers, while handling the essential aspects of tensors, their forms, and their algebraic specifics. The Keras backend must be TensorFlow (or CNTK). Keras can be used for deep learning techniques without needing communication with the more advanced TensorFlow [55]. Therefore, Keras is appropriate for research where prototypes may be quickly developed and tested. The CNN model is selected specifically, and its structure consists of fifteen layers, comprising four convolution layers, four max-pooling layers for extracting features, and one flatten layer. It is followed by three fully-connected layers and three dropout layers for prediction.

The first layer is the convolution layer, with a kernel size of 2×2 and 32 different filters. The second layer is a max-pooling layer with a 4×4 pooling dimension, strides of 4×4, and a type of "same". A convolution layer with a kernel size of 3×3 and 64 different filters and a ReLU activation function are applied in layer three. The fourth layer is a max-pooling layer with a pooling size of 3×3, strides of 3×3, and the property "same". The 5th and 6th layers use the same settings as the 3rd and 4th layers. The seventh layer is the convolution layer, with a kernel size of 5×5 and 128 different filters. Following that, a max-pooling in layer eight that has a pooling dimension of 5×5, strides of 5×5, and a type of "same". The ninth layer is the flatten layer. A dropout layer is in tenth layer to overcome the overfitting problem with a ratio of 0.8. Then, the eleventh layer is a fully connected layer, which has 1024 neurons and a ReLU activation. A dropout layer is in layer twelve, and the ratio is set at 0.6. The thirteenth layer is a fully connected layer with 512 neurons, with a ReLU function. The fourteenth layer is the dropout layer. The last fully connected layer has 1 neuron to predict probability classes in layer number fifteen and uses a sigmoid layer as a prediction layer.

In addition, Mean Squared Error (MSE) is used as the loss function to compute the loss, accuracy as a metric, and this model is trained with 50 epochs. The model hyper-parameters are shown in table 4.4.

TABLE 4.4: HYPER-PARAMETERS OF THE PROPOSED MODEL

Hyper-parameters	Value
Learning rate (α)	1.0
Momentum factor	0.9, 0.99
Batch size	350
Maximum number of epochs	50
Dropout	0.6, 0.8
No. of filters	32, 64, 128

4.9 Evaluating the Proposed Model

Accuracy and loss are used in this research to evaluate the performance of the suggested approach in addition to precision, recall, F-score, and AUPRC measurements. The program is implemented with k-length ranges from 3 nucleotide strings (3-mer) to 8 characters (8-mer). Table 4.5 shows the evaluation of CNN model results by multiple metrics (accuracy, precision, recall, F1-score and others). The results are very close to each other, and this refers to performance efficiency under any k-mer length.

TABLE 4.5: THE EVALUATION OF THE CNN MODEL WITH VARIOUS MEASURES ACROSS MULTIPLE K-MER LENGTHS

CNN Classifier	Evaluation Measures					
	Accuracy	Loss	Precision	Recall	F1- score	AUPRC
K = 3	99.26 %	5 %	100 %	100 %	100 %	1.0
K = 4	99.26 %	5 %	100 %	100 %	100 %	1.0
K = 5	99.2 %	6.31 %	100 %	100 %	100 %	1.0
K = 6	99.18 %	7.63 %	100 %	100 %	100 %	1.0
K = 7	99.14 %	8.44 %	100 %	100 %	100 %	1.0
K = 8	99.06 %	9 %	100 %	100 %	100 %	1.0

On the contrary, when observing the results of previous models of TFBS predictions, the proposed model outperforms all of them on accuracy (ACC), recall (Rec), and precision (Pre) metrics. Table 4.6 and Figure 4.9 offer the full details of the comparison between models.

TABLE 4.6: COMPARISON BETWEEN PROPOSED MODEL AND OTHER SYSTEMS

	ACC %	Pre %	Rec %
DeepBind	78.21	81.96	69.26
DeepSEA	81.58	94.81	64.66
Zeng	81.69	83.69	76.20
DeepSite	85.62	88.47	80.09
Proposed Model	99.26	100	100

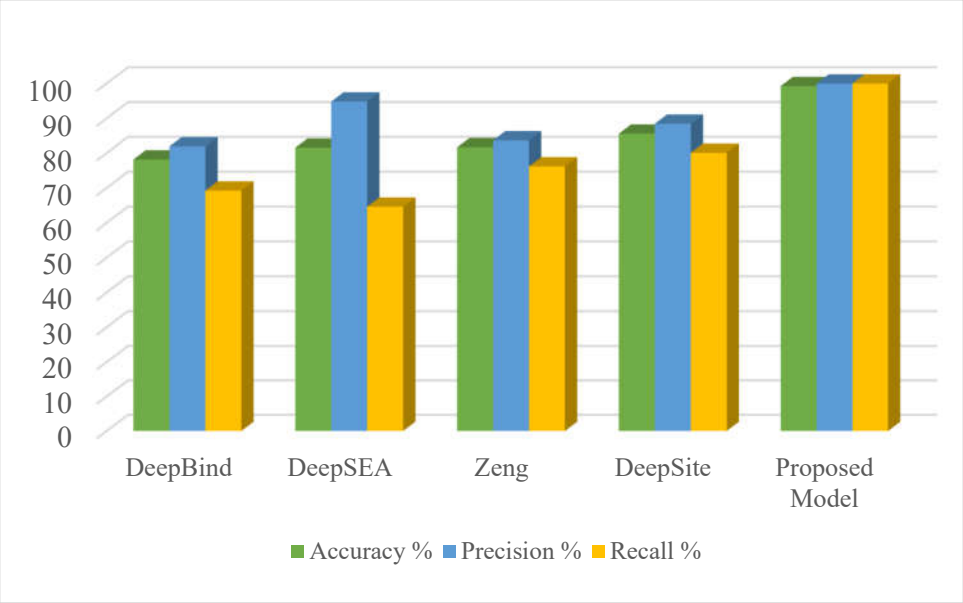


FIGURE 4.9: 3D-MODEL OF METHODS COMPARISON

Chapter Five

CONCLUSION AND FUTURE WORKS

5.1. Conclusions

The most essential characteristics revealed during the implementation of the proposed methodology and the discussion of its results are the following:

1- The proposed system has effectively proved successful in identifying DNA's binding sites via utilizing z-score motif finding method. Furthermore, according to all standard evaluation metrics, the proposed method yields promising findings in the prediction model.

2- The importance of reducing data dimensions in order to avoid overfitting, that effect on the model's performance.

3- The comparison with other methods, as presented in Table (4.6), shows that the proposed system is the best of all models depending on three measurements (accuracy, precision, and recall) by achieving an accuracy of about 99.26%, a recall value of 100%, and a precision value of 100%, too.

4- This research helps biologists more in identifying points or sites of connection between TFs and DNA sequences and speeds up their work and analysis of any other dataset.

5.2. The Future Works

Some future directions can be highlighted here:

1) Applying the suggested system to different databases such as the UniProbe database, ChIP-seq datasets, GTRD datasets and others.

2) Examining and demonstrating the influence of alternative consensus motif mining techniques on the prediction model, such as expectation maximization (EM), Gibbs sampling, and others.
3) Investigating different models that can be compared to the CNN model and attempting to reduce prediction error.
4) Employing various classification methods, such as the SVM algorithm or RNN network, to find DNA binding sites.

References

[1] W. P. Alberts B, Johnson A, Lewis J, Raff M, Roberts K, “Molecular Biology of the Cell Cycle,” 6th ed., vol. 49, no. 2, pp. 175–179, 2014.

[2] Y. Zhang, S. Qiao, S. Ji and Y. Li, "DeepSite: bidirectional LSTM and CNN models for predicting DNA–protein binding," International Journal of Machine Learning and Cybernetics, vol. 11, p. 841–851, 2019.

[3] N. M. Luscombe, D. Greenbaum, and M. Gerstein, “What is bioinformatics? A proposed definition and overview of the field,” *Methods Inf. Med.*, vol. 40, no. 4, pp. 346–358, 2001, doi: 10.1055/s-0038-1634431.

[4] F. Y. Peng, Z. Hu, and R. C. Yang, “Bioinformatic prediction of transcription factor binding sites at promoter regions of genes for photoperiod and vernalization responses in model and temperate cereal plants,” *BMC Genomics*, vol. 17, no. 573, pp. 1–16, 2016, doi: 10.1186/s12864-016-2916-7.

[5] P. Ongsulee, “Artificial Intelligence, Machine Learning and Deep Learning,” in *2017 Fifteenth International Conference on ICT and Knowledge Engineering*, pp. 1–6, 2017, doi: 10.1109/ICTKE.2017.8259629.

[6] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," Nature Methods, vol. 12, no. 10, pp. 931-934, 2015, doi: 10.1038/nmeth.3547.

[7] H. Zeng, M. D. Edwards, G. Liu and D. K. Gifford, "Convolutional neural network architectures for predicting DNA–protein binding," Bioinformatics, vol. 32, no. 12, p. i121–i127, 2016.

[8] H. W. & B. K. Jiyun Zhou, Ruifeng Xu, Yulan He, Qin Lu, “PDNAsite: Identification of DNA binding Site from Protein Sequence by Incorporating Spatial and Sequence Context,” *Sci. Rep.*, vol. 6, no. 27653, 2016, doi: 10.1038/srep27653.

[9] H. R. Hassanzadeh and M. D. Wang, “DeeperBind: Enhancing prediction of sequence specificities of DNA binding proteins,” in *Proceedings - 2016 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2016*, pp. 178–183, 2017, doi: 10.1109/BIBM.2016.7822515.

[10] Q. Zhang, L. Zhu, and D. S. Huang, “High-order convolutional neural network architecture for predicting DNA-protein binding sites,” *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, vol. 16, no. 4, pp. 1184–1192, 2019, doi: 10.1109/TCBB.2018.2819660.

[11] Z. Shen, W. Bao and D.-S. Huang, "Recurrent Neural Network for Predicting Transcription Factor Binding Sites," Scientific Reports, vol. 8, no. 15270, pp. 1-10, 2018.

[12] M. C. and A. B. H. Ameni Trabelsi, “Comprehensive Evaluation of Deep Learning Architectures for Prediction of DNA/RNA Sequence Binding

Specificities," in *Oxford University Press*, vol. 35, no. 14, pp. i269–i277, 2019.

[13] Y. Y. & J. K. Sungjoon Park, Yookyung Koh, Hwisang Jeon and Hyunjae Kim, "Enhancing the interpretability of transcription factor binding site prediction using attention mechanism," *Sci. Rep.*, vol. 10, no. 13413, 2020, [Online]. Available: https://doi.org/10.1038/s41598-020-70218-4.

[14] M. Mahmoud, N. Belal, and A. Youssif, "Prediction of Transcription Factor Binding Sites of SP1 on Human Chromosome1 ,"*appl. sci.*, vol. 11, no. 5123, 2021.

[15] S.Yuan, D.Gao, X.Xie, C.Y.Ma, W.Su, Z. Zhang, Y. Zheng and H.deng, "IBPred: A sequence-based predictor for identifying ion binding protein in phage ," Comput. Struct. Biotechnol. J., vol. 20, pp. 4942-4951, 2022.

[16] M. Fatahi, "Ultra-high Field MRI Bio-effects and Safety Assessment A multidisciplinary Approach," Magdeburg University, 2017.

[17] W.-L. Huang, C.-W. Tung and C. Liaw, "Rule-Based Knowledge Acquisition Method for Promoter Prediction in Human and Drosophila Species," The Scientific World Journal, pp. 1-14, 2014.

[18] M. Mitra, "Elements of RNA, its techniques and applications," American Journal of Current Microbiology, vol. 7, no. 1, pp. 1-6, 2019.

[19] J. M. Finkelstein, "Structures of membrane proteins," *Nature*, vol. 511, no. 21, pp. 1–1, 2014, doi: 10.1038/nature13372.

[20] D.S. Latchman, "Transcription factors: an overview Function of transcription factors," *Int. J. Exp. Path*, vol. 74, pp. 417–422, 1993, [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2002184/pdf/ijexpath00017-0003.pdf.

[21] P. D'haeseleer, "What are DNA sequence motifs?," *Nat. Biotechnol.*, vol. 24, no. 4, pp. 423–425, 2006, doi: 10.1038/nbt0406-423.

[22] M. A. H. Samee, B. G. Bruneau, and K. S. Pollard, "A De Novo Shape Motif Discovery Algorithm Reveals Preferences of Transcription Factors for DNA Shape Beyond Sequence Motifs," *Cell Syst.*, vol. 8, no. 1, pp. 27-42.e6, 2019, doi: 10.1016/j.cels.2018.12.001.

[23] Y. Zhang, J. Xuan, B. G. de los Reyes, R. Clarke, and H. W. Ressom, "Network motif-based identification of transcription factor-target gene relationships by integrating multi-source biological data," *BMC Bioinformatics*, vol. 9, no. 203, 2008, doi: 10.1186/1471-2105-9-203.

[24] A. Cichonska, "Machine Learning for Systems Pharmacology," Aalto University, 2018.

[25] C. Y. Lee and S. Myong, "Probing steps in DNA transcription using single-molecule methods," *J. Biol. Chem.*, vol. 297, no. 3, pp. 1–16, 2021, doi:

10.1016/j.jbc.2021.101086.

[26] N. An, “PLANT HIGH-THROUGHPUT PHENOTYPING USING PHOTOGRAMMETRY AND 3D MODELING TECHNIQUES,” Kansas State University, Kansas, USA, 2015.

[27] A. Yilmaz, M. K. Mejia-Guerra, K. Kurz, X. Liang, L. Welch, and E. Grotewold, “AGRIS: The arabidopsis gene regulatory information server, an update,” Nucleic Acids Res., vol. 39, no. SUPPL. 1, 2011, doi: 10.1093/nar/gkq1120.

[28] E. Rahm and H. Do, “Data cleaning: Problems and current approaches,” *IEEE Data Eng. Bull.*, vol. 23, no. 4, pp. 3–13, 2000.

[29] Najah. A. Shanan, “An Approach to Bacteria Classification based on Topic Modeling and Naive Bayes,” University of Babylon, 2021.

[30] M. Lenzerini, “Data Integration: A Theoretical Perspective,” in *The twenty-first of International Conference on Management of Data and Symposium on Principles Database and Systems*, 2002, pp. 233–246, [Online]. Available: https://doi.org/10.1145/543613.543644.

[31] D. Uhm, S.-H. Jun, and S.-J. Lee, “A Classification Method Using Data Reduction,” *Int. J. Fuzzy Log. Intell. Syst.*, vol. 12, no. 1, pp. 1–5, 2012, doi: 10.5391/ijfis.2012.12.1.1.

[32] M. Domazet-Lošo and B. Haubold, "Alignment-free detection of local similarity among viral and bacterial genomes," Bioinformatics, vol. 27, no. 11, pp. 1466-1472, 2011.

[33] M. Nabeel Asim, M. Imran Malik, A. Dengel, and S. Ahmed, “K-mer Neural Embedding Performance Analysis Using Amino Acid Codons,” in *Proceedings of the International Joint Conference on Neural Networks*, 2020, pp. 1–8, doi: 10.1109/IJCNN48605.2020.9206892.

[34] X. Min, W. Zeng, N. Chen, T. Chen, and R. Jiang, “Chromatin accessibility prediction via convolutional long short-term memory networks with k-mer embedding,” in *Bioinformatics*, 2017, vol. 33, no. 14, pp. i92–i101, doi: 10.1093/bioinformatics/btx234.

[35] S. L. Harris and D. Harris, "Sequential Logic Design," in Digital Design and Computer Architecture, RISC-V ed., Elsevier Inc, 2022, ch. 3, pp. 106-169.

[36] J. Yan and M. Zhu, “A Review about RNA-Protein-Binding Sites Prediction Based on Deep Learning,” *IEEE Access*, vol. 8, pp. 150929–150944, 2020, doi: 10.1109/ACCESS.2020.3014996.

[37] F. A. Hashim, M. S. Mabrouk, and W. Al-Atabany, “Review of Different Sequence Motif Finding Algorithms.,” *Avicenna J. Med. Biotechnol.*, vol. 11, no. 2, pp. 130–148, 2019, [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/31057715%0Ahttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6490410.

[38] N. I. Gershenzon, G. D. Stormo, and I. P. Ioshikhes, “Computational technique for improvement of the position-weight matrices for the DNA/protein binding sites,” *Nucleic Acids Res.*, vol. 33, no. 7, pp. 2290–2301, 2005, doi: 10.1093/nar/gki519.

[39] R. Siddharthan, “Dinucleotide weight matrices for predicting transcription factor binding sites: Generalizing the position weight matrix,” *PLoS One*, vol. 5, no. 3, pp. 1–10, 2010, doi: 10.1371/journal.pone.0009722.

[40] H. S. Booth, J. H. Maindonald, S. R. Wilson, and J. E. Gready, “An efficient Z-score algorithm for assessing sequence alignments,” *J. Comput. Biol.*, vol. 11, no. 4, pp. 616–625, 2004, doi: 10.1089/cmb.2004.11.616.

[41] N. Li and M. Tompa, “Analysis of computational approaches for motif discovery,” Algorithms Mol. Biol., vol. 1, no. 8, 2006, doi: 10.1186/1748-7188-1-8.

[42] Y. LeCun, “The Power and Limits of Deep Learning,” *Res. Manag.*, vol. 61, no. 6, pp. 22–27, 2018, doi: 10.1080/08956308.2018.1516928.

[43] M. A. Wani, F. A. Bhat, S. Afzal, and A. I. Khan, *Advances in Deep Learning*, 1st ed., vol. 57. springer, 2020.

[44] D. Q. Zeebaree, H. Haron, and A. M. Abdulazeez, “Gene Selection and Classification of Microarray Data Using Convolutional Neural Network,” in *ICOASE 2018 - International Conference on Advanced Science and Engineering*, 2018, pp. 145–150, doi: 10.1109/ICOASE.2018.8548836.

[45] S. Ahlawata and A. Choudharyb, “Hybrid CNN-SVM Classifier for Handwritten Digit Recognition,” in *International Conference on Computational Intelligence and Data Science (ICCIDS 2019)*, 2020, pp. 2554–2560, doi: 10.1016/j.procs.2020.03.309.

[46] L. Bragilevsky, “Deep Learning For Satellite Image Analysis,” Simon Fraser University, 2017.

[47] D. Podareanu, V. Codreanu, S. Aigner, and V. Weinberg, “Best Practice Guide - Deep Learning,” 2019. doi: 10.13140/RG.2.2.31564.05769.

[48] A. Wiranata, S. A. Wibowo, R. Patmasari, R. Rahmania, and R. Mayasari, “Investigation of Padding Schemes for Faster R-CNN on Vehicle Detection,” in *Proceedings - 2018 International Conference on Control, Electronics, Renewable Energy and Communications, ICCEREC 2018*, 2018, pp. 208–212, doi: 10.1109/ICCEREC.2018.8712086.

[49] W. Du, Z. Wang, and D. Chen, “Optimizing of Convolutional Neural Network Accelerator,” in *Green Electronics*, 2018, pp. 147–166.

[50] H. Muthanna, “Gene Expression Classification of Alzeheimar Disease Stages Using Machine Learning,”University Of Babylon, 2021.

[51] B. Matougui, M. Batouche, and A. Boukelia, “A K-mer based Multi

Convolutional Neural Network Classifier of Low-Ranking Taxonomic Bins from Metagenome A K-mer based Multi Convolutional Neural Network Classifier of Low-Ranking Taxonomic Bins from Metagenome," in *COSI 2019*, 2019, no. September, pp. 1–13, [Online]. Available: https://www.researchgate.net/ publication/335540811.

[52] A. Chemchem, F. Alin, and M. Krajecki, "Improving the Cognitive Agent Intelligence by Deep Knowledge Classification Article," *Int. J. Comput. Intell. Appl.*, vol. 18, no. 1, 2019, doi: 10.1142/S1469026819500056.

[53] Y. Ren, J. Yang, Q. Zhang, and Z. Guo, "Multi-feature fusion with convolutional neural network for ship classification in optical images," *Appl. Sci.*, vol. 9, no. 4209, 2019, doi: 10.3390/app9204209.

[54] V. M. Patro and M. Ranjan Patra, "Augmenting Weighted Average with Confusion Matrix to Enhance Classification Accuracy," *Trans. Mach. Learn. Artif. Intell.*, vol. 2, no. 4, 2014, doi: 10.14738/tmlai.24.328.

[55] N. K. Manaswi, *Deep Learning with Applications Using Python*. New York, USA: Apress, 2018.

Appendix A
The Accepted Paper

Iraqi Journal of Science
University of Baghdad – College of Science
Editorial Board
EISSN: 2312-1673
ISSN: 0067-2904

No:CS/5049 Date: 9/2/2022

Acceptance Letter

To: Faisal Abdullah Aziz , Sura Z. AL-Rashid

Department of Software, College of Information Technology, University of Babylon

Title:

Prediction of DNA Binding Sites Bound to Specific Transcription Factors by the SVM Algorithm

Dear Author:

We are pleased to inform you that your manuscript is accepted for publication in **Volume (63) Issue (11)** and will be published on **(November) 2022** in **Iraqi Journal of Science (IJS)**.

*Thank you for submitting your work to the **Iraqi Journal of Science (IJS)**.*

Your Sincerely

Prof. Ali H. Ad'hiah **(Ph.D.)**

Editor in Chief

Iraqi Journal of Science

جامعة بغداد / كلية العلوم
المجلة العراقية للعلوم
الجادرية - بغداد - العراق

- **Website:** WWW.ijs.scbaghdad.edu.iq
- **Email address:** http://ijs.scbaghdad.edu.iq/
- **Phone:** 009647903375590
- **Address:** Iraq, Baghdad, Al-Jadriya distract, University of Baghdad Campus, College of Science.

الخلاصة

عوامل النسخ (TFs) هي جزيئات كبيرة ترتبط وتبدأ النسخ في مختلف مناطق الصياغة التعبيرية التي تمثل جزء من promoter الموجود بداخل الحمض النووي. في التعبير الجيني ، تلعب عوامل النسخ (TFs) دورًا رئيسيًا، حيث تقوم بالتحكم في نسبة نقل المعلومات الجينية من الحمض النووي إلى الرنا الناقل عن طريق اتصالها بسلسلة الحمض النووي بموقع معين يسمى مواقع ربط عامل النسخ (TFBS). لطالما كان اكتشاف ما يسمى بال(motif) يمثل تحديًا لأنه يحتاج إلى هياكل بيوفيزيائية دقيقة للإشارة إلى موقع TFBS بداخل سلاسل الحمض النووي. لسوء الحظ ، استطاعت الدراسات البيولوجية الوصول فقط الى بعض مواقع الربط هذه. بالإضافة إلى ذلك ، فإن هذه الاختبارات تستغرق وقتًا طويلاً ومكلفة. من أجل القيام بذلك، يجب إنشاء نظام تنبؤ يتنبأ بدقة بعلاقات الربط بين TFs و سلاسل الحمض النووي. و قد تم اقتراح طرق التعلم العميق مؤخرًا ، و التي اظهرت نتائج تنافسية لعملية التنبؤ بموقع ربط عامل النسخ.

يبدأ النظام المقترح بمرحلة معالجة البيانات التي تتكون من ثلاث خطوات: تنظيف البيانات، وتكامل البيانات ، وتقليل البيانات. بعد ذلك ، ستستمر نتائج هذه المرحلة (وهي تسلسل الحمض النووي) على مسارين. يتكون المسار الأول من الخطوات التالية: تضمين -*k* mer ، ترميز احادي قوي ، وربط كل سلسلة DNA مع TF\s الخاص به. يتضمن المسار الثاني أيضًا ثلاث خطوات ، وهي: توليد *k*-mer ، و طريقة z-score لايجاد النمط المتكرر، وربط كل فكرة إجماع بمجموعة البيانات. بعد ذلك ، سيتم إدخال الإخراج من هذه المسارات في مرحلة التشفير. في هذه المرحلة، يتم ترميز الميزات في شكل رقمي. بعد ذلك ، يتم إدخال الميزات المشفرة في CNN لتصنيف مواقع ربط الحمض النووي.

أخيرًا ، تم تنفيذ خطوة التقييم اعتمادًا على مقاييس مختلفة مثل الدقة ، و نسبة الخطأ ، والدقة (Precision) ، و ال(Recall) ، ودرجة F1 وAUPRC . أظهرت النتائج أن أداء النظام المقترح كفوء و فعال. حقق النموذج دقة تنبؤ بلغت ٩٩,٢٦٪ ، ونسبة خطأ تتراوح بين ٥ و ٩٪ ، ودقة (Precision) ، واسترجاع ، ودرجة f1 تبلغ ١٠٠٪. بالإضافة إلى ذلك ، حقق AUPRC نسبة ١٠٠ ٪ أيضًا.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية تكنولوجيا المعلومات - قسم البرمجيات

تصميم نظام للتنبؤ بمعامل استنساخ الDNA بالاعتماد على خوارزمية الCNN

رسالة مقدمة الى

مجلس كلية تكنولوجيا المعلومات - جامعة بابل كجزء من متطلبات

نيل درجة الماجستير في تكنولوجيا المعلومات / البرمجيات

من قبل

فيصل عبد الله عزيز لفته

بإشراف

أ.م.د. سرى زكي الراشد

١٤٤٤ هـ ٢٠٢٢ م