*Th_ECE: 18_08*

# Protein Secondary Structure Prediction using Graph Neural Network with Ensemble Learning

**Name: Md Liad Hossain**
**Roll: 1809008**

**February, 2024**

**Department of Electronics and Communication Engineering**
*Khulna University of Engineering & Technology*
*Khulna- 9203, Bangladesh*

# Protein Secondary Structure Prediction using Graph Neural Network with Ensemble Learning

This thesis is submitted to the Department of Electronics and Communication Engineering, Khulna University of Engineering & Technology, in partial fulfillments for the requirements of the degree of

**“Bachelor of Science in Electronics and Communication Engineering”**

Supervised by

**Dr. A.B.M. Aowlad Hossain**
**Professor,**
**Department of Electronics and Communication Engineering,**
**KUET, Khulna.**

Submitted by

**Md Liad Hossain**
**Roll: 1809008**
**Department of Electronics and Communication Engineering,**
**KUET, Khulna.**

**February, 2024**

**Department of Electronics and Communication Engineering**
*Khulna University of Engineering & Technology*
*Khulna- 9203, Bangladesh*

# Protein Secondary Structure Prediction using Graph Neural Network with Ensemble Learning

## Author

**Md Liad Hossain**
Roll: 1809008
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

## Supervisor

**Dr. A.B.M Aowlad Hossain**
Professor
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

……………………………………….
Signature

## External

**Dr. Sheikh Md. Rabiul Islam**
Professor
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

………………………………………
Signature

**February, 2024**

**Department of Electronics and Communication Engineering**
*Khulna University of Engineering & Technology*
*Khulna- 9203, Bangladesh*

# Statement of Originality

This thesis has not been published anywhere in full, or in part. All items, or sub items that appear on the thesis are referenced, where needed. The thesis cannot be copied without the permission of the author. Also this thesis has not submitted earlier as an undergraduate thesis.

The above declarations are true. Understanding these, this work has been submitted for evaluation of an undergraduate thesis.

Date:  22 February, 2024 Md Liad Hossain

# Acknowledgement

All the praise to the almighty God, whose blessing and mercy succeeded me to complete this thesis work fairly. After that, I humbly acknowledge the valuable suggestions, advice, guidance and sincere co-operation of **Dr. A.B.M Aowlad Hossain**, Professor, Department of Electronics and Communication Engineering, Khulna University of Engineering & Technology, under whose supervision this work was carried out. His intellectual advice, encouragement and guidance make me feel confident and scientific research needs much effort in learning and applying and need to have a broad view at problems from different perspective. I also want to convey my heartiest gratitude to my external **Dr. Sheikh Md. Rabiul Islam**, Professor, Department of Electronics and Communication Engineering, Khulna University of Engineering & Technology, for giving his valuable time to evaluate my thesis. I would like to convey my heartiest gratitude to all the faculty members, official and staffs of the Department of Electroincs and Communication Engineering as they have always extended their cooperation to complete this work. Last but not least, I wish to thank my friends and family members for their constant supports.

# ABSTRACT

Predicting the secondary structure of proteins using amino acid sequences is crucial in learning about the properties and functions of proteins, including DNA replication, biological processes, and enzyme catalysis.Using a graph neural network, we offer a method for predicting a protein's secondary structure. Because all amino acids in the primary sequence are connected to two neighbor amino acids, a graph can be drawn to depict their interconnectedness. The graph Neural Network concept takes advantage of this link to get some remarkable results. First, use the primary sequence (amino acid) to create a graph from the dataset. The model's feature is orthogonal encoding, which assigns a unique embedding to each node in the network. Then, using the GNN algorithm, iterate the entire graph to aggregate the information of the neighbor nodes. Then we have applied window padding method and extracted the feature vector to get optimized output. Then also added a feature called conformation parameter for all pair of distinct amino acid and secondary structure. Conformation parameters are the proportions that amino acids tend to form secondary structures. Following this, two well-known neural network model called support vector machine (SVM) and Random Forest are inserted into the graph for different length of primary sequence, which identifies the 8 states of the protein secondary structure. SVM was used for smaller primary sequences and Random Forest was used for larger primary sequences. The simulation results show that the accuracy on the RSCB PDB dataset is 92.62%, which is better than other models. Then we have analyzed various performance measurement parameters and compared with the conventional methods. The comparison analysis demonstrates how much superior the suggested method is over traditional approaches.

**Contents**

**Acknowledgement** **iv**

**ABSTRACT** **v**

**List of Figures** **viii**

**List of Tables** **ix**

**List of Abbreviations** **x**

**CHAPTER I : Introduction** **1**

1.1 Objectives 1

1.2 Motivation 2

1.3 Unfamiliarity of the Solution 2

1.4 Work Plan 3

1.5 Project Budget 3

1.6 Organization of The Thesis 3

**CHAPTER II : Literature Review and Theoretical Background** **5**

2.1 Introduction 5

2.2 Protein 5

2.2.1 Protein Primary Structure 6

2.2.2 Protein Secondary Structure 7

2.2.3 Protein Tertiary Structure 7

2.2.4 Protein Quaternary Structure 8

2.3 Graph 8

2.4 Support Vector Machine 11

2.5 Random Forest 16

2.5.1 Decision Tree 16

2.5.2 Ensemble Methods 17

2.5.3 Working Procedure of Random Forest 17

2.6 Related Work 18

2.7 Summary 21

**CHAPTER III : Proposed Method: Graph Neural Network with Ensemble Learning** **22**

3.1 Introduction 22

3.2 Design And Fabrication of Proposed Model 22

3.2.1 Algorithm And WorkFlow 23
3.2.2 Data Prepossecing 24
3.2.3 Encoding 24
3.2.4 Graph Generation 24
3.2.5 Window Padding 29
3.2.6 Conformation Parameter 30
3.2.7 Neural Network 30
3.3 Summary 31
**CHAPTER IV : Experimental Results and Analysis 32**
4.1 Introduction 32
4.2 Dataset 32
4.3 Performance Measure Indices 33
4.4 Results 36
4.5 Comparative Analysis 39
4.6 Summary 39
**CHAPTER V : Socio-Economic Impact and Sustainability 40**
5.1 Impact on Societal, Health, Legal and Cultural Issues 40
5.1.1 Societal Impact 40
5.1.2 Health Impact 40
5.1.3 Legal and Ethical Considerations 40
priority, especially when handling genetic or health-related data. It is ethically required to obtain informed consent from people who contribute to datasets. 41
5.1.4 Cultural Impact 41
5.2 Impact on the environment and sustainability 41
**CHAPTER VI : Addressing Complex Engineering Problems and Activities 42**
6.1 Addressing Complex Engineering Problems 42
6.2 Addressing Complex Engineering Activities 43
**CHAPTER VII : Conclusions, Limitations and Future Works 44**
7.1 Conclusions 44
7.2 Limitations 44
7.3 Future Works 44
**References 45**

# List of Figures

| | |
|---|---|
| Figure 1.1: Work Plan of the thesis. | 3 |
| Figure 2.1: An Illustrative Display of Graph | 9 |
| Figure 2.2: Graphical Representation of Linear SVM | 12 |
| Figure 2.3: Graphical Representation of Non-linear Dataset | 13 |
| Figure 2.4: Graphical Representation of  Non-linear Dataset in 3-D | 13 |
| Figure 2.5: Graphical Representation of Non-linear SVM in 3-D | 14 |
| Figure 2.6: Graphical Representation of Non-linear SVM | 14 |
| Figure 2.7: Visualization of Bagging and Boosting Methods | 17 |
| Figure 2.8: Visualization of Random Forest Algorithm | 18 |
| Figure 3.1: An outline of our Proposed Model | 23 |
| Figure 3.2: Proposed GNN Model Flowchart | 25 |
| Figure 3.3: A Random Graph G | 26 |
| Figure 3.4: Embedding (One-hot-encoded) Vectors for Each Vertex | 26 |
| Figure 3.5: Updating Every Vertex of The Graph | 28 |
| Figure 3.6: Obtaining the Complete Graph’s Vector H | 28 |
| Figure 3.7: An Amino Acid Graph | 28 |
| Figure 3.8: Updating Vertex Of Amino Acid | 29 |
| Figure 3.9: Padding in the Basic Sequence that is Encoded | 30 |
| Figure 3.10: Window Sliding Scheme with Window’s Length = 5 | 30 |
| Figure 4.1: Splitting the Dataset into training and test set | 36 |
| Figure 4.2: Visualization of different types of performance parameters | 37 |
| Figure 4.3: Confusion Matrix of The Model | 37 |
| Figure 4.4: Prediction of primary sequence ‘SLLKKLLLA’ | 38 |
| Figure 4.5: Prediction of primary sequence ‘AAFRAA’ | 38 |

# List of Tables

Table 1.1: Overall budget of the project. 3

Table 2.1: Amino Acids' Name and Symbol 6

Table 2.2: Description of Secondary Structure 7

Table 3.1: Orthogonal Primary Encoding 27

Table 3.2: Orthogonal Secondary Encoding 29

Table 4.1: Hyperparameter Tuning of GNN 36

Table 4.2: Variations of Accuracy with Various Window Size 36

Table 4.3: Performance Analysis 37

Table 4.4: Comparison with Various Model 39

Table 6.1: Complex Engineering Problems of the Project 42

Table 6.2: Complex Engineering Activities of the Project 43

# List of Abbreviations

| | |
|---|---|
| **GNN** | **G**raph **N**eural **N**etwork |
| **SVM** | **S**upport **V**ector **M**achine |
| **PSSP** | **P**rotein **S**econdary **S**tructure **P**rediction |
| **DNN** | **D**eep **N**eural **N**etwork |
| **DSSP** | **D**ictionary of **P**rotein **S**econdary **S**tructure |
| **PDB** | **P**rotein **D**ata **B**ank |
| **PSS** | **P**rotein **S**econdary **S**tructure |

# CHAPTER I

# INTRODUCTION

Proteins consist of amino acid chains, each featuring a central alpha carbon linked to a hydrogen atom, an amino group (NH2), and a carboxyl group (COOH). Predicting protein secondary structures is vital for understanding their properties, functions, and medical implications, including treatment efficacy and identifying abnormal functions like hereditary metabolic diseases. Despite its importance, our knowledge of protein structures, particularly in humans, remains limited. Continuous research in this area is essential for advancing our understanding of biology and improving medical interventions. Nonetheless, human protein structures account for just around 0.2% percent of the total [1-2]. It is still unclear how a protein's structure was first deducted from its sequence more than 50 years ago[6]. To discriminate between ephemeral and stable cell complexity, manual experimental methods like electron microscopy[7], X-ray crystallography, and nuclear magnetic resonance (NMR) spectroscopy can be costly, time-consuming, and often unreliable. Experimental approaches have flaws that computational methods can overcome. In recent decades, computational methods have become increasingly popular in the field of bioinformatics, largely because they offer simplicity, user-friendliness, affordability, and rapid processing speeds.[1, 4, 8]. Computational approaches predict the functions of unidentified proteins by analyzing the structural and sequence data of known proteins[9]. And there has been a slew of initiatives to investigate this area.

## 1.1 Objectives

In this thesis, the graph neural network (GNN) is employed to forecast the secondary structure of proteins. Throughout the implementation process, input data is encoded into a direct or indirect graph format, which then generates a target value by considering information from graph nodes and edges. Specifically, static undirected homogeneous graphs were utilized using a supervised setup approach among the diverse range of graph types.

The study will be carried out with the following precise aims to achieve the goal:

• To research and implement the graph neural network model.

• To study the support vector machine (SVM) and Random Forest in the GNN’s neural network.

• In this work, we suggest using SVM for smaller length and Random Forest for larger length with hyper-parameter adjustment to improve classification accuracy performance.

• To evaluate the performance of the proposed strategies to that of other current methods to determine which one performs better.

## 1.2 Motivation

Huge and intricate molecules, proteins are involved in many vital functions within the human body. The distinct activity of proteins is determined by the interplay between protein tertiary and secondary structures. Over the years, several investigations and analyses of protein structure have been carried out. Less than 20,000 protein structures (Protein Data Bank) have been experimentally established, despite the fact that there are over a million known protein sequences. Because of this, it's getting more and more crucial to infer the structure of proteins from their amino acid sequence using pre-existing structures. On the other hand, we can produce a huge number of secondary structures from a particular main structure in a shorter amount of time and effort by employing computer power. Because of new technologies and a variety of techniques, prediction accuracy increases quickly as technology develops.

The utilization and exploration of GNN in bio-informatics are relatively limited due to its novelty as a model. Therefore, we are motivated to delve into this state-of-the-art theory in our thesis, aiming to harness its potential for the betterment of humanity and the world.

## 1.3 Unfamiliarity of the Solution

The Graph Neural Network is a relative new model that is very less explored compared to other methods which are used to predict protein structures. In this thesis, we intend to further explore this emerging field with new implementation and prediction methods to achieve highers accuracy and efficiency.

## 1.4 Work Plan

The following guntt chart depicts the work plan of the thesis:

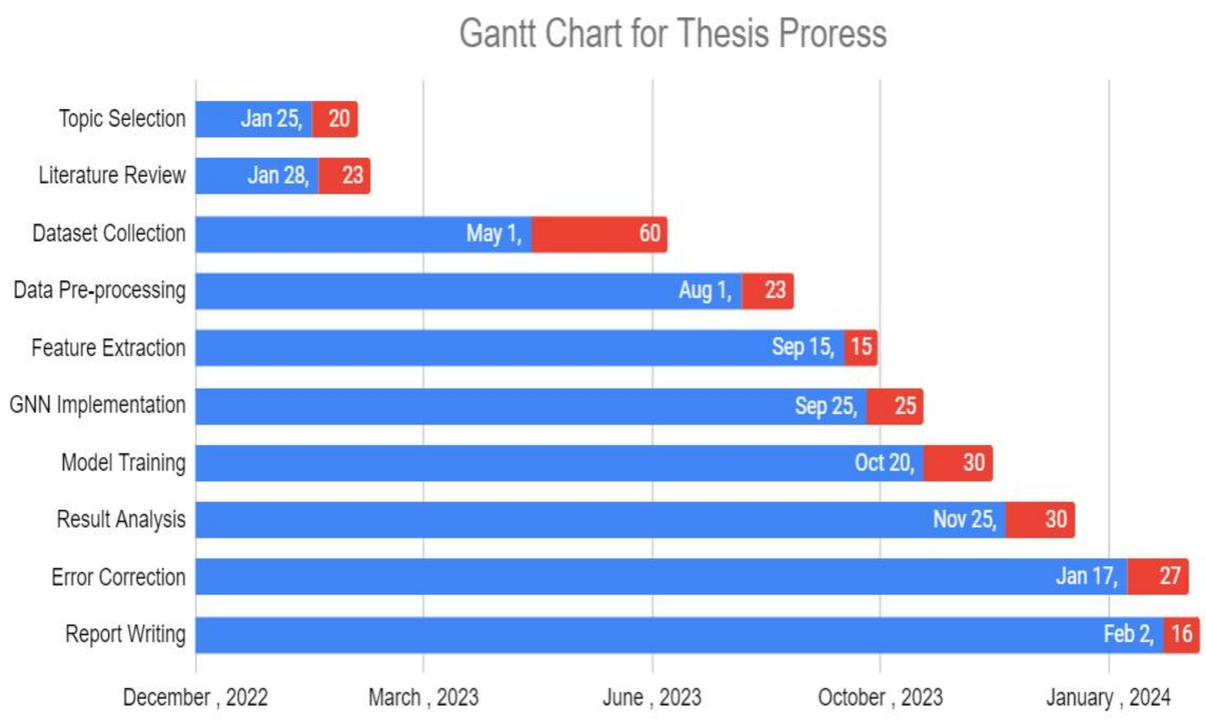


**Figure 1.1:** Work Plan of the thesis.

## 1.5 Project Budget

**Table 1.1:** Overall budget of the project.

| Item | Justification | Price (BDT) |
|---|---|---|
| Project evaluation and report cost | Requires for printing the thesis final report and binding | 1000 |

## 1.6 Organization of The Thesis

The subsequent sections of the thesis are structured in the following manner.

**Chapter-II** provides the necessary background information. This also includes a detailed description of the classification technique, protein, and its classification, as well as about graph.

This chapter also covers a variety of SVM topics and also Random Forest topics. It also offers enticing advantages of the proposed strategy for data classification.

**Chapter-III** illustrates the suggested system, entitled the Graph Neural Network, and explains how it works in detail. The general functional schematic of the proposed system is also included in this chapter. It then goes into great detail about the classification algorithm. It also explains how the graph for our proposed solution was created. It also explains how messages are passed from one node to another.

**Chapter-IV** discusses the datasets and demonstrate the suggested method's experimental outcomes on ccPDB 2.0 dataset. It shows how GNN outperforms other classification techniques in this case.

**Chapter-V** discusses about the socio-economic impact and sustainability of the project.

**Chapter-VI** addresses the complex engineering problems and activities.

**Chapter-VII** lists the observations made at the end of the experiments. It also outlines what areas of future study should be investigated to get a more ideal data classification.

# CHAPTER II

## Literature Review and Theoretical Background

### 2.1 Introduction

The basic notion of protein and its classification are introduced in this chapter. The graph is then briefly discussed. After that, look at the support vector machine. Some similar classification-based works are also examined. This chapter further explains how the suggested solution overcomes the limitations of existing methods.

### 2.2 Protein

Proteins, essential to all living organisms, consist of organic materials composed predominantly of carbon, hydrogen, nitrogen, oxygen, and sulfur. These complex molecules are constructed from amino acid polymers, also known as peptides, with each polymer comprising 20 distinct amino acid residues. These amino acids form long chains, numbering from hundreds to thousands, with their specific sequence dictating the three-dimensional structure and function of the protein. Genetic information encoded in DNA dictates the amino acid sequence, which is crucial for the protein's biological role. Proteins fold into unique spatial arrangements to fulfill their diverse functions, with their three-dimensional structures often holding key insights into their molecular actions. The hierarchical organization of protein structure includes primary, secondary, tertiary, and quaternary levels, with the linear sequence of amino acids defining the primary structure. Additionally, the termini of proteins, known as the C and N terminals, play a significant role, representing the free amino and carboxyl groups at each end of the protein molecule.

### 2.2.1 Protein Primary Structure

Simply said, primary structure is the arrangement of amino acids within a polypeptide chain. It is the most fundamental level of protein sequence. There are up to 20 distinct amino acids in one protein. As an illustration in the Table 2.1.

**Table 2.1:** Amino Acids' Name and Symbol

| Index | Symbol | Name |
|---|---|---|
| 1 | A | Alanine |
| 2 | R | Arginine |
| 3 | N | Asparagine |
| 4 | D | Aspartate |
| 5 | C | Cysteine |
| 6 | Q | Glutamine |
| 7 | E | Glutamate |
| 8 | G | Glycine |
| 9 | H | Histidine |
| 10 | I | Isoleucine |
| 11 | L | Leucine |
| 12 | K | Lysine |
| 13 | M | Methionine |
| 14 | F | Phenylalanine |
| 15 | P | Proline |
| 16 | S | Serine |
| 17 | T | Threonine |
| 18 | W | Tryptophan |
| 19 | Y | Tyrosine |
| 20 | V | Valine |

The 20 amino acids can be grouped according to their characteristics. Important factors to take into account are charge, size, hydrophilicity or hydrophobicity, and functional groups. These characteristics have an impact on protein structure and interactions between proteins. Protein primary structures are defined as the linear combination of amino acids that will be used as input to the suggested model.

### 2.2.2 Protein Secondary Structure

Secondary structure pertains to the spatial arrangements within protein molecules that emerge after the sequential assembly of compounds but precede the folding into the tertiary structure. Hydrogen bonding plays a pivotal role in shaping the skeletal framework of the elongated chain and establishing secondary structures such as coils, strands, and helices at localized regions[10]. During the formation of secondary and tertiary structures, protein molecules typically adopt three distinct configurations, often referred to as 3-state amino acids. These configurations include turns, beta sheets, and alpha helices, with each amino acid molecule capable of assuming one of these states. To represent and visualize protein secondary structure, the Dictionary of Protein Secondary Structure (DSSP) codes are commonly employed due to their extensive repertoire of single-letter symbols[11]. Furthermore, these three states extend into eight states, as seen in the Table 2.2.

**Table 2.2:** Description of Secondary Structure

| Sl. No. | Code | Description |
|---|---|---|
| 1 | H | alpha helix |
| 2 | G | $3_{10}$ helix |
| 3 | I | pi helix |
| 4 | B | beta bridge |
| 5 | E | extended beta sheet |
| 6 | C | coil |
| 7 | S | bend |
| 8 | T | hydrogen bonded turn |

### 2.2.3 Protein Tertiary Structure

The entire three-dimensional structure of the polypeptide is referred to as the tertiary structure of a protein. One or more protein secondary structures, also referred to as protein domains, plus a single polypeptide chain "backbone" will make up the tertiary structure. Amino acid sidechains can form bonds and engage in a range of interactions. A protein's side chain interactions and bonding shape its tertiary structure. A protein's tertiary structure, which can relate to either the

entire tertiary structure or just one protein domain, is determined by its atomic coordinates[12, 13]. A quaternary structure can be formed by folding several tertiary structures together[14].
The tertiary structure of proteins is often divided into domains, which are discrete, compact folding units with an average of 100–200 residues. While big proteins often have multiple domains, small proteins may just have one. A fold, which denotes the spatial assembly of secondary structural elements into a form resembling a domain, is a characteristic found in many proteins. Folds are often linked to a certain purpose, however this is not always the case.
Although it is typically smaller and acts as the basis for folds, a structural motif is similar to a fold. While some structural motifs, especially those linked to a particular biochemical function, are structurally similar and frequently present in protein domains with similar properties, others are found in a broad variety of irrelevant proteins with different modifications.

### 2.2.4 Protein Quaternary Structure

In protein structures, the quaternary structure arises from interactions between multiple polypeptide chains. When numerous polypeptide chains combine, they form a complex known as an oligomer. Quaternary structure proteins comprise multiple subunits of the same protein or may include diverse components. For instance, hemoglobin exemplifies a protein with quaternary structure, where it binds oxygen molecules due to its iron content. Hemoglobin is composed of two alpha and two beta subunits, demonstrating its quaternary arrangement.

## 2.3 Graph

A graph is a non-linear data structure composed of edges and nodes, sometimes referred to as vertices. A graph is a picture of a group of objects where some of the elements are connected by edges. Vertices are the points that join the things that are connected, and edges are the ties that join the vertices.
Formally speaking, a graph is defined as two sets (V, E), where V is the set of vertices and E is the set of edges connecting the vertice pairs. Take a look at the graph in the Figure 3.3.
where,
nodes or vertices, V = A,B,C,D,E
and Edge, E = AC, BC, DC, CE

A graph can be represented as an array of nodes and a two-dimensional array of edges. When a network is utilized to represent the problem area, graphs have been employed to solve real-world issues. Examples of networks include circuit networks, phone networks, social networks, and protein chains.

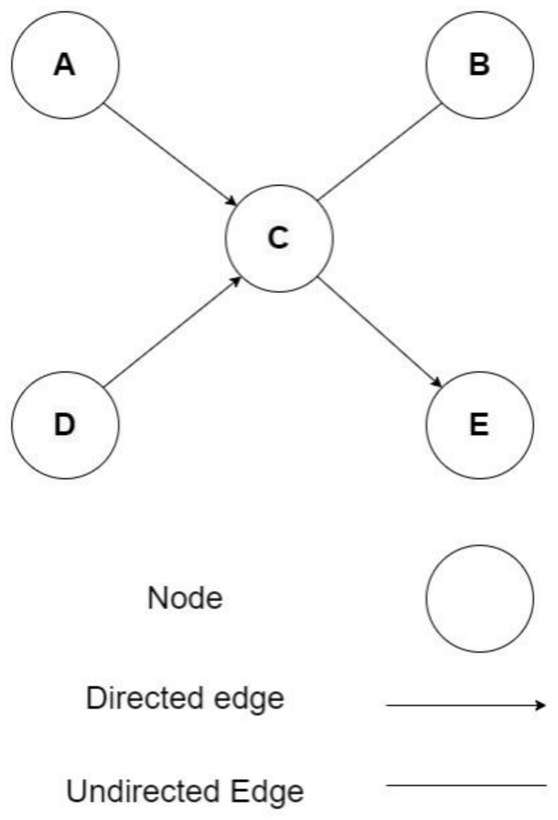


**Figure 2.1:** An Illustrative Display of Graph

**Vertex**

A vertex represents each node in the graph. The labeled circle in the Figure 3.3 denotes vertices. As a result, A to E are vertices. An array can be used to represent them.

**Edge**

A path or a line connecting two vertices is represented by an edge. The lines from A to C, B to C, and so on indicate edges in the diagram 3.3. A two-dimensional array can be used to represent an edge.

**Adjacency or Neighbour node**

If two nodes or vertices are joined by an edge, they are considered a neighboring node or adjacent node. B is neighbor to C in the Figure 3.3, while C is neighbor to {A, B, D, E}, and so on.

The most common graph structure scenarios are as follows:

• **Structural Scenarios:** When graphs are explicit, they are referred to as structural situations.

• **Non-structural Scenarios:** Situations that don't follow a structural pattern involve implicit graphs and the need to construct a graph from a task.

Graphs can be seen from many angles and exist in a range of sizes and forms. Based on weight, the graph can be altered.

• **Weighted Graph** - Graphs with values on their edges or routes. Weights refer to all of the values connected with the edges. The value of the edges can reflect weight, cost, or length.

• **Unweighted Graph** - When the edge has no value or weight attached to it. Unless there is a value connected with the graph, it is unweighted by default.

The following types of graphs can be classified based on their direction.

• **Directed Graph** - A digraph is a network of objects (node, edge) in which all of the vertices are routed from one vertex to the next.

• **Undirected Graph** - When a group of objects is linked together and all of their edges are bidirectional. In a connected two-node system, you can go in any direction from one to the other. Every edge in undirected graphs is considered as bi-directional edges.

Graphs can be categorized into the following categories based on their nodes and edges similarity.

• **Homogeneous Graph:** Homogeneous graph's edges and nodes are similar types. For example, a social network is a homogeneous graph made up of people and their connections that all reflect the same entity type.

• **Heterogeneous Graph:** Heterogeneous graph's edges and nodes are not similar types. For example, buyer, product, and seller nodes are connected via wants-to-buy, iscustomer-of, has-bought, and is-selling edges in the graph encoding a marketplace.

Based on the runtime, graphs can be classified into the following categories.

• **Static Graph:** Static graph is effective for offline optimization/scheduling of graphs.

• **Dynamic Graph:** The input properties of a dynamic graph are more versatile since they change over time. It is necessary to realize graph learning tasks in order to build the loss function. Tasks related to graph learning fall into three groups.

• **Node-level Tasks:** Node level problems involve semi-supervised problem solving techniques such as node classification, which involves classifying nodes into many classes, node regression, which predicts a continuous value for each node, node clustering, which splits nodes into distinct disjoint groups depending on their kinds, etc.

• **Edge-level Tasks:** Edge-level activities such as link prediction and edge classification need the prediction of current edges between two nodes or the classification of different types of edges.

• **Graph-level Tasks:** Graph representation identification is necessary for all kinds of graph-level applications, including graph matching, graph regression, and graph classification. When predicting a protein's secondary structure, the basic sequence of the protein must be used as the basis. Once the primary sequence (amino acid) graph has been created, each node is classified into one of the eight secondary structure states using the node level job.

From the viewpoint of supervision, three types of graph learning tasks based on training settings are given below.

• **Supervised Setting:** It supplies labeled data during training.

• **Semi-supervised Setting:** For training, it provides both unlabeled and labeled data where the ratio of unlabeled data is larger than labeled data. At the testing phase, the transductive situation supplies the prediction of labels of unlabeled vertices but the inductive situation produces new unlabeled vertices from a similar distribution. Lately, mixed setting such as mixed transductive inductive scheme [15, 16] are implemented.

• **Unsupervised Setting:** To predict patterns of the model, it only provides unlabeled data.

## 2.4 Support Vector Machine

Support vector machines, or SVMs, are a classification technique that make use of optimal margins. Object recognition,[17] speaker identification,[18] and text categorization[19] are just a few of the disciplines where SVM has been successfully utilized. SVM offers two primary advantages over newer algorithms: faster processing and better performance with fewer samples (in the thousands). When there are more dimensions than samples, SVM is effective.

At its core, SVM operates by transforming samples into a high-dimensional Hilbert space and seeking a separating hyperplane within this space[20]. By utilizing labeled training samples representing each category, Support Vector Machine (SVM) endeavors to identify the optimal linear decision boundary, or hyperplane, with the widest margin between classes[21].

There are two types of SVM:

• **Linear SVM:** Linear SVM is used for linearly separable data, or that can be divided into two groups with just one straight line. The effectiveness of the Linear SVM method can be understood by an example. Let’s consider the following Figure 2.2, which contains a

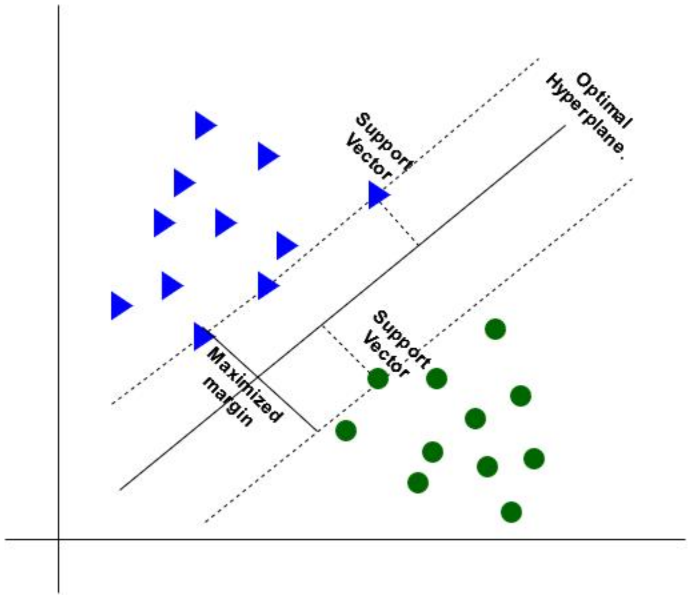


**Figure 2.2:** Graphical Representation of Linear SVM

dataset with two tags (green and blue) and two features (x1 and x2).

We're in pursuit of a classifier capable of discerning between green and blue within the coordinate pair (x1, x2). In a two-dimensional space, drawing a straight line between these classes enables straightforward differentiation. SVM facilitates the identification of the optimal line, often termed a hyperplane, among various possibilities for distinguishing these classes. Utilizing the SVM method, we pinpoint the closest points (support vectors) from both classes to delineate the margin, defined as the distance between these vectors and the hyperplane. The optimal hyperplane is determined by maximizing this margin.

• **Non-linear SVM:** Non-linear SVM is used for data that is non-linearly separable, or that cannot be divided into two groups with a single straight line. Here, we are unable to draw even a single straight line (Figure 2.3).

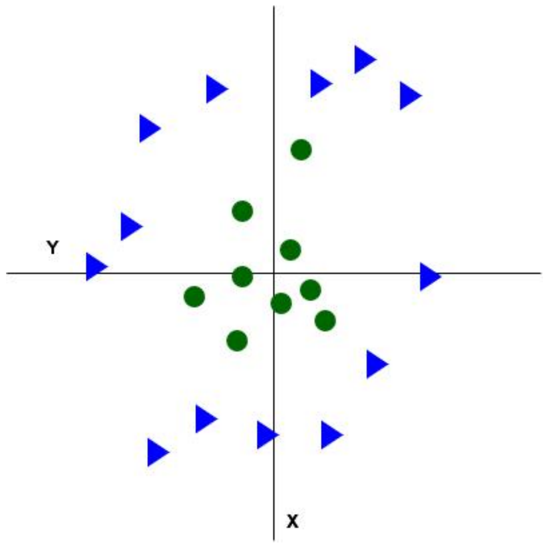


**Figure 2.3:** Graphical Representation of Non-linear Dataset

So to separate these data points we'll need to add another dimension. We'll add a third dimension (z), for non-linear data (Figure 2.4). It can be computed as follows:

$z = x^2 + y^2$

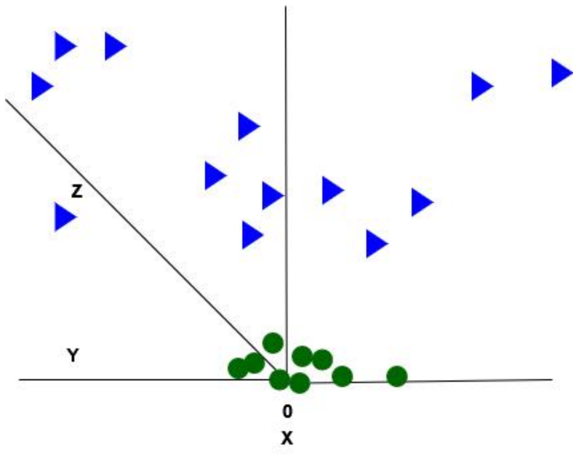


**Figure 2.4:** Graphical Representation of Non-linear Dataset in 3-D

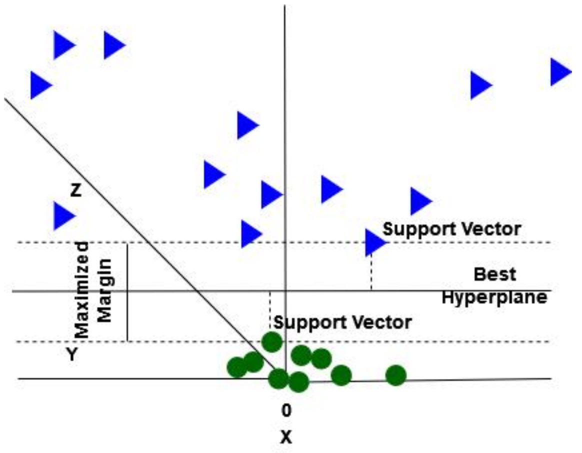


**Figure 2.5:** Graphical Representation of Non-linear SVM in 3-D

As seen in the following Figure 2.5, SVM will now partition the datasets into classes. It appears to be a plane parallel to the x-axis as we’re in 3-d space (Figure 2.5). After converting it to 2d space with z=1, it looks like this (Figure 2.6), with radius=1.

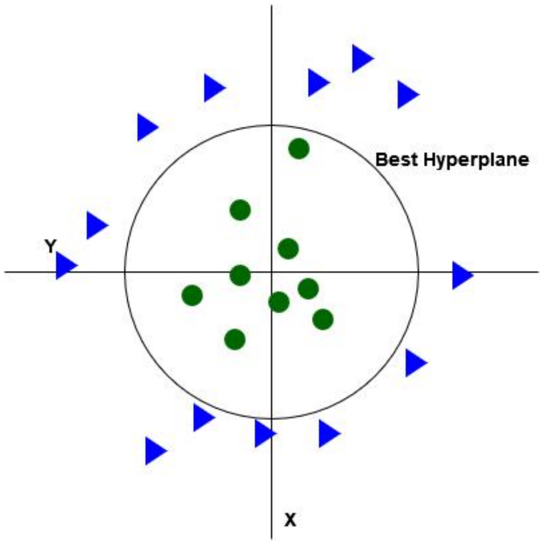


**Figure 2.6:** Graphical Representation of Non-linear SVM

SVM classifies a class by K (X, $X_i$) is the subsequent kernel function:

$$f(x) = \sum_{n=1}^{N} \alpha_i t_i K(X, X_i) + b \tag{2.1}$$

Here,

$\alpha_i$ = Individual weight

N = Total count of vectors

$t_i$ = Target

b = Bias term

$X_i$ = Support vectors

$\alpha_i t_i K(X, X_i)$ = Total weight of each node

SVM kernels include radial basis function (RBF), linear, sigmoid, and polynomial, among others [22].

• **Radial Basis Function (RBF) Kernel:** The radial basis function, which creates a "bump" around each data point, is one of the most widely used kernel functions.

$$K(X, X_i) \ = \ e^{-\gamma \|X - X_i\|^2}, \gamma > 0 \tag{2.2}$$

• **Sigmoid Kernel Function:** The Sigmoid kernel function is often represented as:

$$K(X_i, X_j) \ = \ tanh(\gamma X_i^T X_j \ + \ r) \tag{2.3}$$

• **Polynomial Kernel Function:** The dot product is computed using the Polynomial kernel, which tends to produce directionally dependent results, by optimizing the kernel's power. The magnitude of the output depends on the magnitude ($x_i$) of the vector.

$$K(X, X_i) \ = \ (1 \ + \ X. X_i^T)^d, \ d = degree \ of \ kernel \ function \tag{2.4}$$

Kernel hyperparameters C and $\gamma$ are two of them. The penalty for misclassifying a data item is determined by C, and the decision region is represented by $\gamma$.

## 2.5 Random Forest

Leo Breiman and Adele Cutler are the trademark holders of the widely used machine learning technique known as "random forest," which aggregates the output of several decision trees to produce a single conclusion. Its versatility and ease of use, combined with its ability to handle both regression and classification problems, have driven its popularity.

### 2.5.1 Decision Tree

Given the composition of multiple decision trees, it's advantageous to first grasp the basics of the decision tree algorithm. Decision trees start with a straightforward query, such as "Should I surf?" This question leads to subsequent inquiries, like "Is the wind blowing offshore?" or "Is there a long-period swell?" These inquiries serve as the basis for data division, forming decision nodes within the tree structure. Each query guides toward a final conclusion, represented by a leaf node. The "Yes" branch contains observations meeting the criteria, while the opposite path holds those that do not. Typically trained with the Classification and Regression Tree (CART) algorithm, decision trees aim to identify optimal splits to partition the data effectively. Split quality is evaluated using metrics like mean square error (MSE), information gain, or Gini impurity. Despite the drawbacks of decision trees, such as bias and overfitting, the random forest algorithm overcomes these issues by aggregating multiple trees into an ensemble. This ensemble approach yields more accurate predictions, especially when individual trees exhibit low correlation with one another.

The Random Forest algorithm generates judgments based on data in a variety of ways, therefore it's necessary to understand a few key words linked to this process. Some of these terms include:

- **Entropy**

  It is a metric for the data set's unpredictability or randomness.

- **Information Gain**

  The information gain is a measurement of the drop in entropy following the splitting of the data set.

- **Leaf Node**

  A node that conveys the judgment or the categorization is called a leaf node.

- **Decision Node**

A node that has two or more branches.

- **Root Node**

  All of your data is stored in the root node, which is the highest decision node.

### 2.5.2 Ensemble Methods

A collection of classifiers, like as decision trees, are used in ensemble learning techniques, and their predictions are combined to determine the most frequently occurring outcome. The two most popular ensemble techniques are boosting and bagging, sometimes referred to as bootstrap aggregation.

- **Bagging:** Bagging is the process of generating a distinct training subset via replacement from sample training data. The result is determined by a majority vote.
- **Boosting:** Boosting is the process of turning weak learners into strong learners by building successive models until the greatest accuracy model is reached. For instance, ADA and XG boosts.

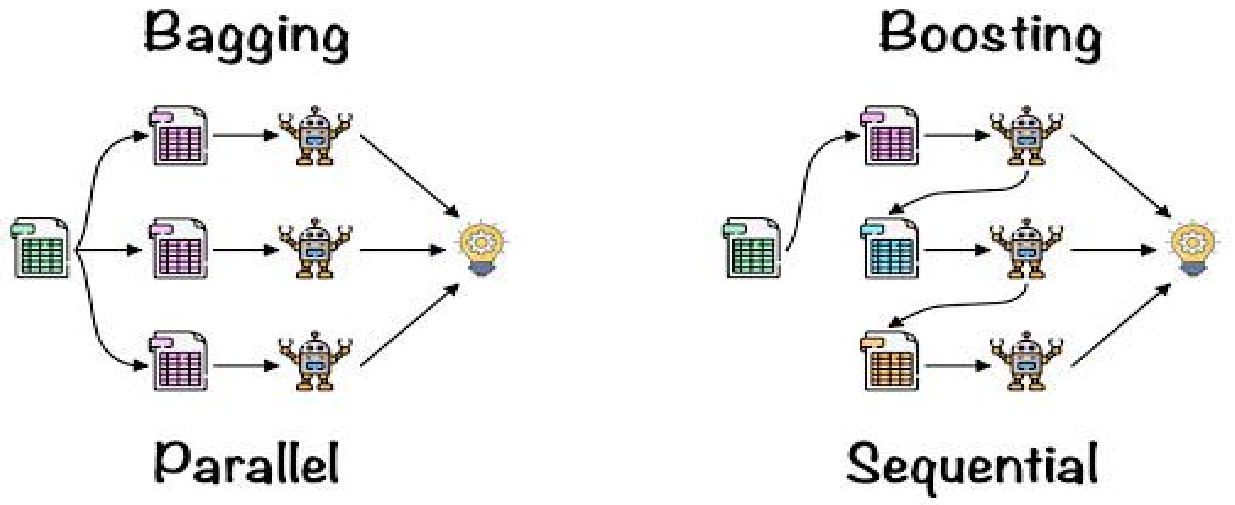


**Figure 2.7:** Visualization of Bagging and Boosting Methods

### 2.5.3 Working Procedure of Random Forest

The random forest algorithm relies on three key hyperparameters that must be configured prior to training: node size, the quantity of trees, and the number of features sampled. With these parameters set, the random forest classifier becomes applicable for tackling both regression and classification tasks.

A bootstrap sample, or a sample of data taken from a training set with replacement, makes up each decision tree in the ensemble of decision trees used in the random forest technique. We shall talk more about the out-of-bag (oob) sample, which is one-third of the training sample that is classified as test data. Then, another randomization is introduced via feature bagging, which decreases decision tree correlation and increases dataset variety. The type of difficulty will determine how the prediction is determined. In a classification task, the predicted class will be decided by a majority vote or the most prevalent categorical variable, whereas in a regression job, each decision tree will be averaged. Finally, utilizing the oob sample and cross-validation, that prediction is verified.

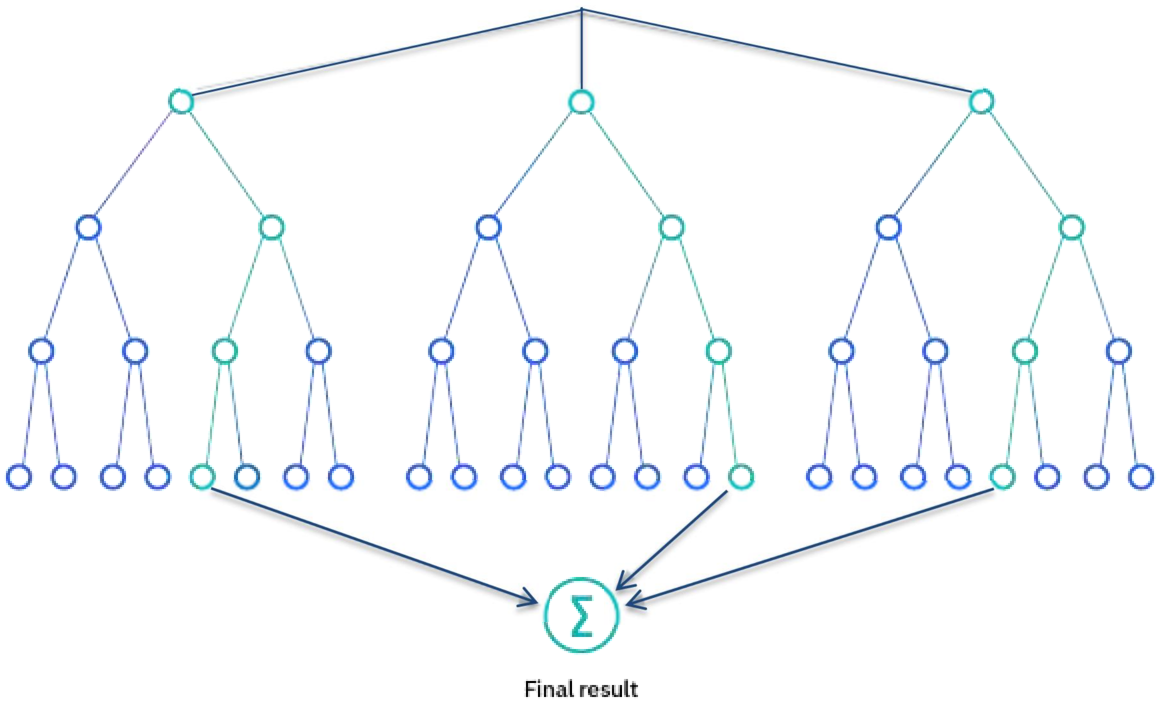


**Figure 2.8:** Visualization of Random Forest Algorithm

## 2.6 Related Work

Predicting the secondary structure of proteins from sequences serves as a pivotal intermediary step, linking primary and tertiary structure prediction. Numerous studies have employed a range of supervised and unsupervised methodologies to forecast protein secondary structure. Various machine and deep learning algorithms, including support vector machines, have been utilized to predict the protein's secondary structure [23, 20], probability graph models [24, 25], artificial neural network [26-29], hidden Markov models [30, 31], bidirectional recurrent neural network(BRNN) [32-34], and so on.

Traditionally, protein secondary structures have been categorized into three states, but advancements in the field led to the introduction of eight aspects for secondary assignment in 1983 [21] and are still in use today. In an effort to enhance the efficiency of protein secondary structure prediction, a recent study introduced an innovative deep learning architecture [35]. This architecture combines prediction by a residual network, convolutional neural network, and bidirectional recurrent neural network to achieve improved results. On the CB513 dataset benchmark, the proposed deep network achieved significant accuracy rates of 71.4% for 8-state prediction and 74% for ensemble learning.

Even in cases where the sequence of the target protein deviates significantly from known references, nearest-neighbor-based methods offer a strategy to estimate its secondary structure by leveraging local sequence similarities. This approach often involves employing a sliding window to compare segments of the target protein with those of known proteins. Among the widely adopted nearest neighbor prediction servers are NNSSP [36] and Preator [37]. NNSSP server, harnessing multiple sequence alignments to integrate evolutionary insights, achieves an accuracy rate of 72.2%. In contrast, Preator utilizes local pair-wise alignment of the target sequence, setting itself apart from conventional multiple sequence alignment techniques.

Hidden Markov Models (HMMs) have emerged as another tool for secondary structure prediction, akin to the nearest neighbor method. In this approach, HMMs are constructed within a structured context to predict the structure of unknown proteins. First, a multiple sequence alignment profile is generated using short fragments from similar sequences with known structures [38]. This methodology is exemplified by Bystroff et al. in their program HMMSTR, boasting an accuracy rate of 74.3% [39].

The neural network approach draws inspiration from the synaptic connections of brain neurons, processing inputs through multiple stages before mapping them to the desired outcome. Neural networks are trained using a dataset of sequences with known structures, adjusting weight values to influence signals. A notable example of this technique is Rost and Sander's [40] PHD program, boasting 72.2% accuracy when trained on a test set of profiles from various sequences. Moreover, leveraging larger databases and PSI-BLAST[27] to construct the training set further improved accuracy to 75%.

In their study, Kathuria et al. [8] employ a Random Forest classification approach to predict the secondary structure of proteins. Originally developed by Breiman in 2001, Random Forest is a versatile technique that utilizes classification and regression trees (CART) to generate numerous decision trees from a single training dataset [41]. In their research, the authors focus exclusively on classifying alpha (α) structure and non-alpha structure. During the preprocessing step, they assign the structure-property of all alpha structure polypeptides to "alpha," while setting the structure of all other polypeptide sequences to "non-alpha." The effectiveness of their model is validated using a ROC curve, demonstrating its ability to achieve the desired outcomes. Notably, the study is confined to binary classification within their scope.

In the referenced article, Support Vector Machines (SVM) and Artificial Neural Networks (ANN) are employed for predicting protein secondary structure [42]. An effort is made to enhance accuracy by condensing the 8 secondary protein states (G, E, I, H, T, S, B, C) into three states (H, C, E). Both primary sequences and targets undergo conversion into binary forms in these methods, with prediction executed using a sliding window [43, 44]. Various binary classifiers exhibit differing performances when employing an ad hoc sliding window size in NN and SVM techniques[45, 20]. Rost et al. utilized three-layer feed-forward NNs with protein evolutionary data on 126 non-homologous [46] data sets (RS126), achieving an accuracy of Q3 = 70.8%. To address overfitting, Riis et al. [47] proposed utilizing NNs with minor neural networks for predicting three protein secondary structure states. Additionally, they attained 71.3% accuracy on the RS126 set by employing another NN and integrating multiple alignment information as a feature. By leveraging a larger training dataset of 681 non-homologous proteins, Chandonia et al. [48] developed a novel technique for PSSP, achieving 74.8% accuracy.

Vapnik et al. introduced the support vector machine (SVM) [49] as an alternative solution for classification problems, utilizing frequent profiles and evolutionary information as encoded inputs [20]. Building on this, other researchers have utilized PSI-BLAST PSSM profiles [50] as input vectors and integrated the sliding window approach with SVM architecture, resulting in a 70 percent accuracy rate [51]. SVM has emerged as a reliable tool for addressing PSSP challenges, often outperforming other learning systems, including neural networks [52].

Ahmed et al. utilized a genetic algorithm alongside a multiple window technique, achieving a Q3 accuracy of 68 percent. Spencer and colleagues [53] developed a protein secondary structure

predictor based on position-specific scoring matrices, employing BLAST and optimized deep learning network topologies trained on a dataset of 198 proteins.
Additionally, Zhang et al. elucidated the architecture of a Graph Neural Network (GNN) [54], showcasing its potential for predicting protein secondary structure. Inspired by this, GNN is leveraged in protein secondary structure prediction, marking a significant advancement in the field. With these insights, the application of graph neural networks for predicting protein secondary structure has garnered considerable attention.

## 2.7 Summary

This chapter introduces the fundamental ideas of protein, graph, and support vector machines, as well as their categorization. Then contrast this novel approach to some existing work and methodologies. The limits of those approaches were identified, and various parts of the proposed solution to address those flaws were described.

# CHAPTER III

# Proposed Method: Graph Neural Network with Ensemble Learning

## 3.1 Introduction

This chapter delves into the concept and operation of the proposed model graph neural network with ensemble learning. The classification algorithm in this study uses graph and vertice relationships, which is a novel approach in the bio-informatics field. The accuracy of categorization algorithms improves as a result of this. GNN is a classification method where a set of objects is connected and every edge is bidirectional. It may be applied in many different contexts, such as data mining and machine learning.

## 3.2 Design And Fabrication of Proposed Model

Graph neural networks (GNNs) have emerged as a powerful tool for handling data structured in graph formats. Traditional machine learning and deep learning methods primarily focus on tabular or sequential data, making it challenging to effectively analyze graph data due to its diverse nature. However, the increasing popularity of GNNs in various domains such as recommendation systems[55, 56], knowledge graphs[57], protein-protein interaction networks[58], social networks[59] and bio-informatics[49, 50]has shown promising results. GNNs excel in capturing latent patterns within diverse types of graph data, including images, texts, and videos. These networks encode information considering the scale, type, and structure of the graph, along with the appropriate loss functions, enabling them to generate valuable insights based on node and edge information.

Graphs offer insights into the relationships between nodes and their connections, encompassing various categories such as dynamic directed heterogeneous graphs, dynamic undirected heterogeneous graphs, and static directed heterogeneous graphs. These categories possess distinct properties that can be independently combined.

In many instances, protein prediction relies on text data representing the amino acid sequences, forming the basis of the graph structure. However, certain proteins exhibit unchanging amino acid sequences, resulting in static, undirected homogeneous graphs where nodes and edges remain constant.

This approach uses labeled data for training and prediction. Thus, a supervised environment is employed. Every node in a graph neural network (GNN) updates its feature using an aggregate function after sending its feature message to its neighbor.

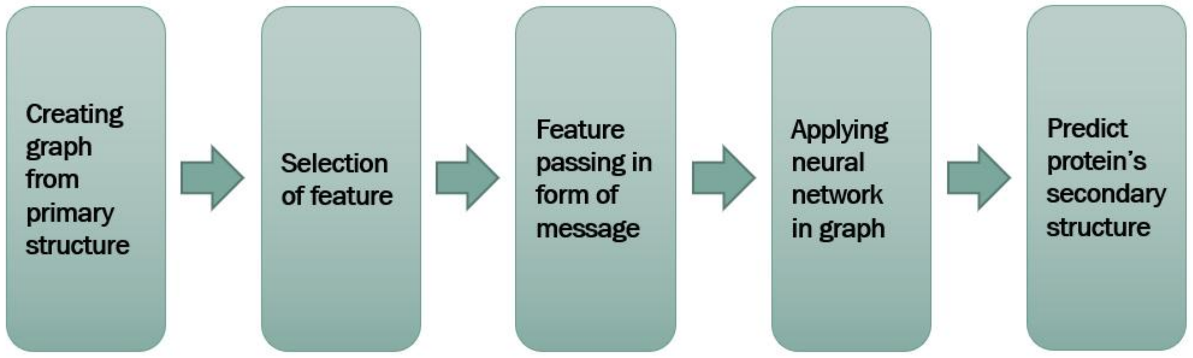

Creating graph from primary structure
Selection of feature
Feature passing in form of message
Applying neural network in graph
Predict protein's secondary structure

**Figure 3.1:** An outline of our Proposed Model

### 3.2.1 Algorithm And WorkFlow

The overall protein secondary structure prediction process is divided into multiple phases utilizing the GNN technique. Such as:

• **Step 1:** We gather data from a variety of sources and preprocess it to fill in missing data and remove irrelevant information.

• **Step 2:** For every amino acid, apply orthogonal encoding to extract the model feature.

• **Step 3:** Using this feature vector in each node, a linear graph will be constructed based on that node's amino acid.

• **Step 4:** Every vertex in the graph is updated using Neighbourhood Aggregation by appending embeddings of the current vertex and neighboring vertices.

• **Step 5:** Using zero paddings helps prevent the projected missing sequence of the initial and terminal amino acid sequences in the training dataset that is created by applying a sliding window approach.

• **Step 6:** An additional feature called conformation parameter is added for every amino acid in each dataset.

• **Step 7:** This modified graph is subjected to a neural network (SVM) model that delivers the best solution for noisy data.

• **Step 8:** Examine the differences between the real and predicted secondary structures of proteins.
• **Step 9:** Finally generate accuracy.

### 3.2.2 Data Prepossecing

To optimize time complexity and conserve CPU and RAM resources, we preprocess protein primary and secondary structures by selecting a subset of approximately 10,000 datasets. This preprocessing step aims to mitigate potential sequence labeling problems caused by mismatched lengths between primary and secondary structures. Additionally, non-standard amino acids such as B, Q, U, X, and Z are excluded during this preprocessing phase.

### 3.2.3 Encoding

We convert the eight secondary protein structures into integer representations spanning from 0 to 7. Additionally, each of the twenty amino acids is encoded using a distinct binary vector, commonly known as one-hot encoding or orthogonal encoding. This encoding scheme facilitates subsequent computations, including those involving sliding windows[51].Given the visibility of 20 amino acids, the feature dimensionality is set to 20. Consequently, each character within the amino acid sequence possesses an embedding length of precisely 20 characters.

### 3.2.4 Graph Generation

Suppose we have graph G, as depicted in Figure 3.3, where each vertex holds one-hot encodings or embeddings representing the index of the current vertex. Now, envision Figure 3.4, where information from neighboring nodes is aggregated into each node, similar to the previous graph. At each edge, a basic feed-forward (FF) neural network is applied, and subsequently, each vertex is transformed into a recurrent unit or another type of neural network architecture.
"Neighbourhood Aggregation" is a message passing procedure that each vertex in the graph carries out by sending information across the directed edges from nearby vertex around a designated reference vertex (Figure 3.5). The next step is to execute n iterations, renovating each vertex by adding the node that is the result of the neighboring vertex embeddings and applying a repeating function to the present embedding.

Ultimately, the entire graph is condensed into a vector, denoted as H (as shown in Figure 3.6), achieved by aggregating all embeddings. This vector H captures the unique characteristics of the graph. Additionally, a minor refinement has been introduced to the model. Specifically, in our proposed model, the one-hot encoding of each amino acid character is embedded within each vertex of the graph (refer to Figure 3.7), representing a distinguishing feature of our approach.

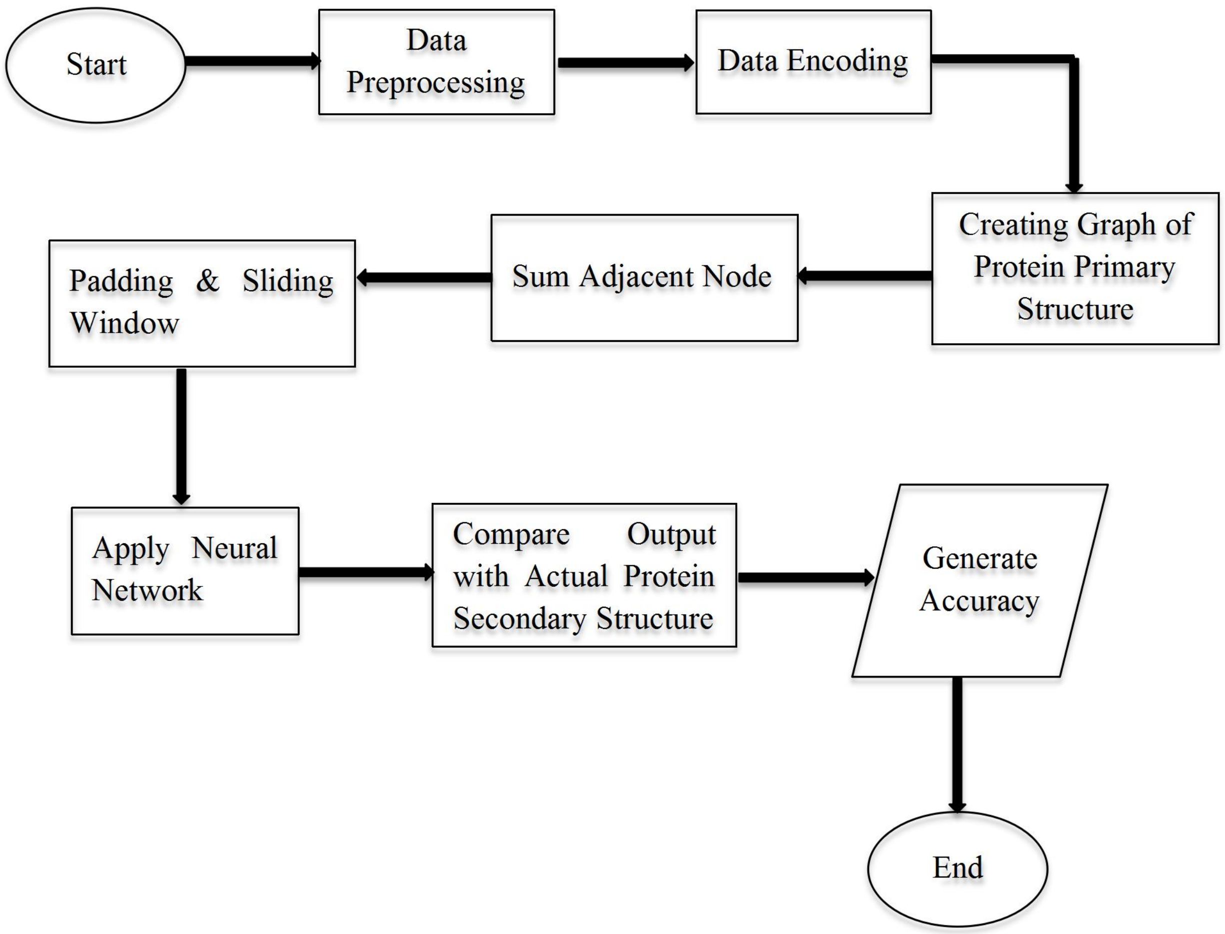


**Figure 3.2:** Proposed GNN Model Flowchart

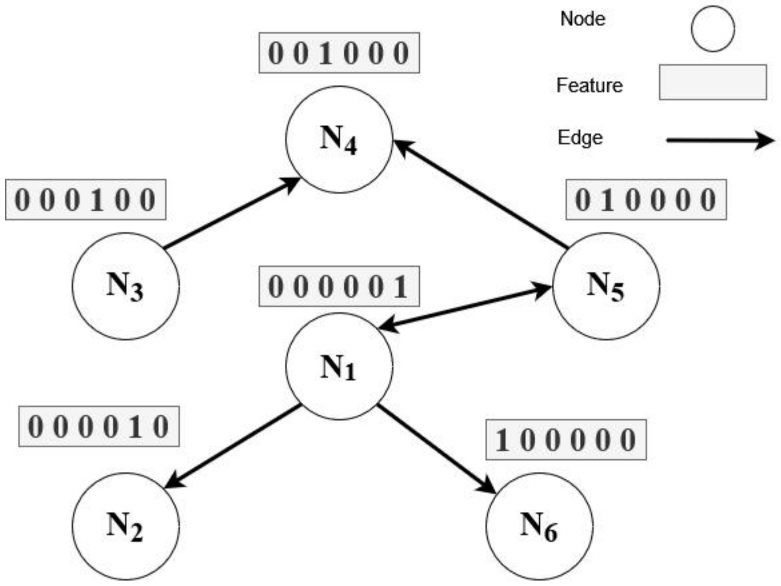


**Figure 3.3:** A Random Graph G

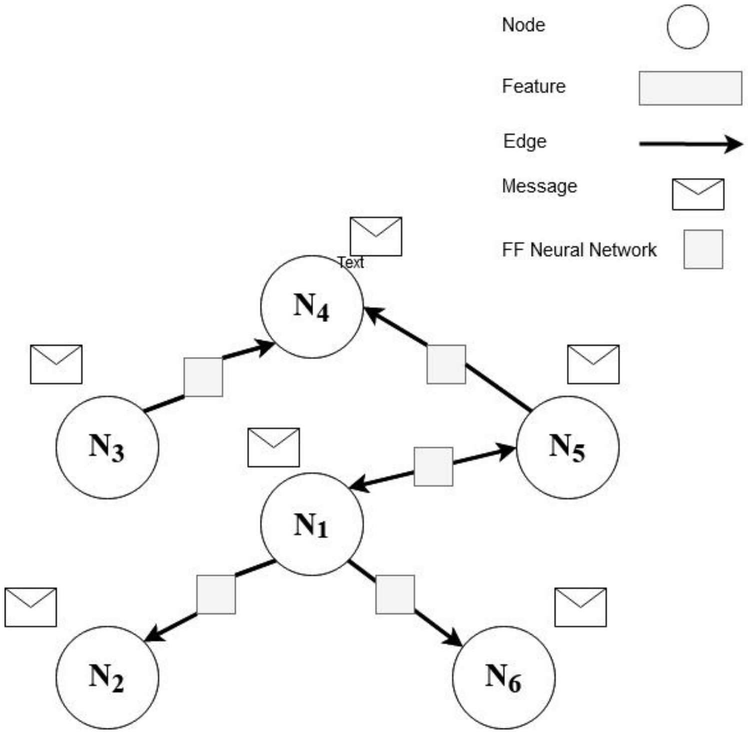


**Figure 3.4:** Embedding (One-hot-encoded) Vectors for Each Vertex

**Table 3.1:** Orthogonal Primary Encoding

| Amino Acid | Encoding |
|---|---|
| A | 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| C | 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| E | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| D | 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| G | 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| F | 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| I | 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| H | 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 |
| K | 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 |
| M | 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 |
| L | 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 |
| N | 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 |
| Q | 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 |
| P | 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 |
| S | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 |
| R | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 |
| T | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 |
| W | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 |
| V | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 |
| Y | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| X | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |

We use Neighbourhood Aggregation to add embedding of the current vertex and surrounding vertices to each vertex of the model graph, as it is a linear graph.

The graph mentioned above will now be iterated using the following guidelines:

1. For i = 0;

$$n_i = n_{i+1}$$

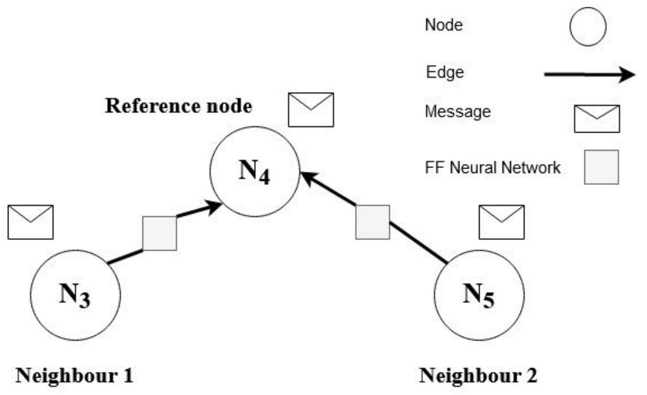


**Figure 3.5:** Updating Every Vertex of The Graph

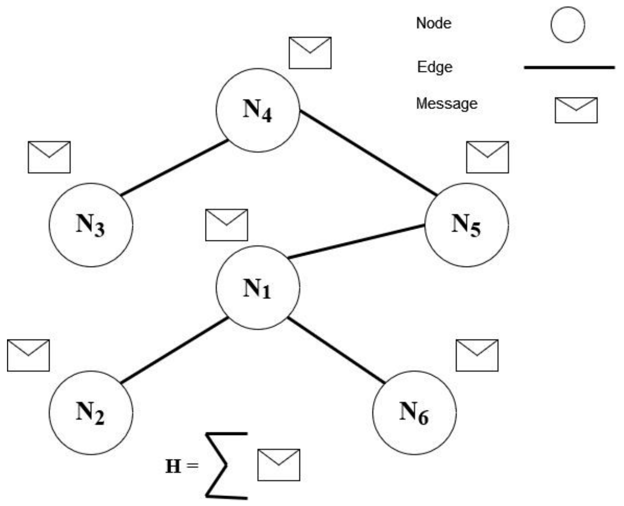


**Figure 3.6:** Obtaining the Complete Graph's Vector H

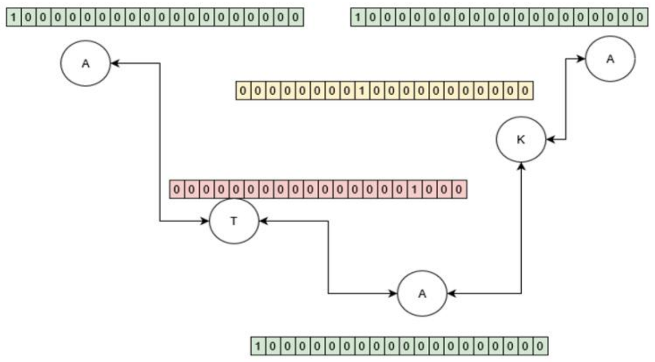


**Figure 3.7:** An Amino Acid Graph

**Table 3.2:** Orthogonal Secondary Encoding

| Secondary Structure of Protein | Encode |
|---|---|
| H ($\alpha$-helix) | 0 |
| C (Loops and irregular elements) | 1 |
| E (beta-strand) | 2 |
| B (beta-bridge) | 3 |
| G (three-helix) | 4 |
| I (pi-helix) | 5 |
| T (Turn) | 6 |
| S (Bend) | 7 |

2. For $0 < i < p$;

$$n_i = n_{i-1} + n_i + n_{i+1}$$

3. For $i = p$;

$$n_i = n_{i-1} + n_i$$

Here, i denotes the graph's vertex location, $n_i$ the one-hot encoding of the amino acid character for the i-th vertex, and p the terminal node.

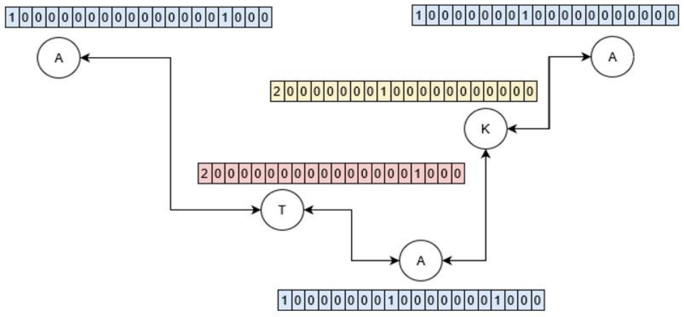


**Figure 3.8:** Updating Vertex Of Amino Acid

### 3.2.5 Window Padding

In order to avoid the expected loss of a protein main sequence or the beginning and terminal amino acid sequences, the sliding window technique employs zero paddings (Figure 3.9). The goal value is the window's middle amino acid that anticipates the amino acid's matching secondary structure.

Consider the following window scheme (Figure 3.10), where the input sequence 'AKNLK' is used and the window length is set to 5, resulting in a 5x20 size vector because the feature is 20.

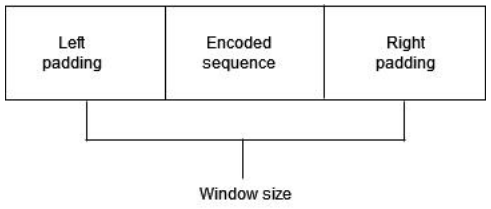


**Figure 3.9:** Padding in the Basic Sequence that is Encoded

To feed the SVM and Random Forest, this kind of minuscule amount will be formed as the iteration proceeds.

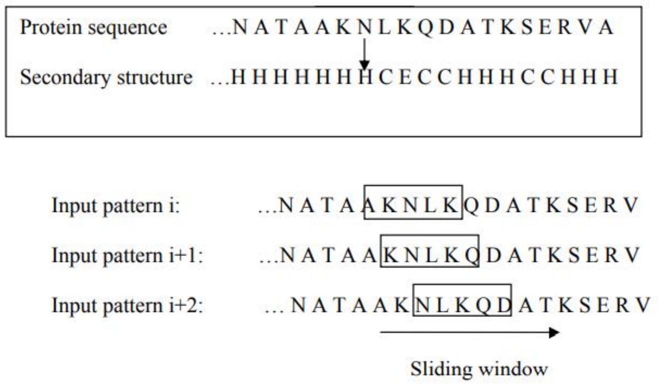

Protein sequence ...N A T A A K N L K Q D A T K S E R V A
Secondary structure ...H H H H H H H C E C C H H H C C H H H
Input pattern i: ...N A T A A K N L K Q D A T K S E R V
Input pattern i+1: ...N A T A A K N L K Q D A T K S E R V
Input pattern i+2: ... N A T A A K N L K Q D A T K S E R V
Sliding window

**Figure 3.10:** Window Sliding Scheme with Window's Length = 5

### 3.2.6 Conformation Parameter

Another feature vector that has been included is the conformation parameter. The quantities at which amino acids often form secondary structures are known as conformation parameters. The following is a definition of the conformation parameters for each amino acid $S_{ij}$:

$$S_{ij} = \frac{a_{ij}}{a_i}$$

Where i=1,2,….,20 and j=1,2,3,4,5,6,7,8.

Let's denote one of the eight categories of secondary structures as "j," representing H, C, E, B, G, I, T, and S. Each of the twenty amino acids is represented by "i." Within this context, "$a_{ij}$"

signifies the count of the i-th amino acid with the j-th secondary structure, while "$a_i$" represents the count of the i-th amino acid within a dataset.

### 3.2.7 Neural Network

In conclusion, we assess the predictive accuracy of protein secondary structures using a hybrid approach. For shorter primary structures, we rely on a neural network (SVM) model, while for longer primary structures, we turn to a Random Forest model. SVM kernel hyperparameters are:

• **C:** The penalty for misclassifying a data point is determined by C.

• **Gamma:** The decision region is represented by gamma ($\gamma$). larger gamma ($\gamma$) value overfits the training data.

Enhanced accuracy was attained by fine-tuning the parameters, setting C=0.1, $\gamma$=1.5, and utilizing the radial basis function kernel for the Kaggle (RSCB PDB) dataset.Random Forest kernel hyperparameters are:

• **n_estimators:** How many decision trees are present in the forest. While raising this hyperparameter usually enhances the model's performance, it also raises the computational expense of training and prediction.

• **random_state:** It governs the feature sampling strategy during the search for the optimal split at each node (when max_features < n_features), as well as the stochasticity introduced by bootstrapping the samples for tree construction (when bootstrap=True).

Better accuracy was achieved with n_estimators=100, random_state=42 for the kaggle(rscb pdb).

## 3.3 Summary

This chapter begins with a well-organized flowchart that displays the suggested method's working mechanism, followed by a representation of the graph creation and message passing, aggregation, and finally feeding into a neural network for classification. The support vector machine that is involved is indeed highlighted.

# CHAPTER IV

## Experimental Results and Analysis

### 4.1 Introduction

This chapter delves into evaluating the efficacy and efficiency of the proposed methodologies. Several tests are carried out to evaluate the scalability of these approaches. Additionally, a simulation study is conducted to offer deeper insights into the effectiveness of the proposed approach. The chapter scrutinizes the simulation results alongside experimental datasets to gauge the effectiveness of the proposed strategies.

### 4.2 Dataset

Using the RSCB PDB dataset from Kaggle, multiple experiments with various features are performed to determine the accuracy of the suggested technique.

The RSCB PDB dataset, obtained via Kaggle (Dataset Source), is one of the benchmark dataset for predicting protein secondary structure. The dataset has 100000 instances. Peptide sequences and secondary structures are listed in the primary dataset. It's a tabular representation of (https://cdn.rcsb.org/etl/kabschSander/ss.txt.gz), which was downloaded from the RSCB PDB on 2018-06-06.

Description of columns of dataset:

• pdb_id: the id that was used to find it on https://www.rcsb.org/

• chain_code: When a protein contains many peptides (chains), the chain code is required to identify a specific one.

• seq: the peptide's sequence of amino acids

• sst8: the secondary structure with eight states (Q8)

• sst3: secondary structure with three states (Q3)

• len: the number of amino acids in the peptide

• hasnonstdaa: whether there are any non-standard amino acids in the peptide (B, O, U, X, or Z).

The nonstandard proteins B, O, U, X, and Z are all masked with the "*" character. It is usual to reduce the aforementioned eight states (Q8) to three (Q3) for secondary structure prediction by merging (E, B) into E, (H, G, I) into E, and (C, S, T) into C.

## 4.3 Performance Measure Indices

We carried out numerous experiments using different combinations of datasets sourced from various repositories to evaluate the classification accuracy and computational efficiency of graph neural networks. Our evaluation involved employing a range of error metrics, such as MAE, MSE, and RMSE. Within this assessment framework, True Positives (TP) indicate correctly identified positive instances, while False Negatives (FN) represent incorrectly classified positives. True Negatives (TN) correspond to accurately identified negative instances, while False Positives (FP) denote misclassified negative instances.

**Accuracy**

The accuracy of a measurement system refers to the degree of agreement between the measured quantity and its true value.

$$Accuracy = \frac{True\ Positive(TP) + True\ Negative(TN)}{True\ Positive(TP) + True\ Negative(TN) + False\ Positive(FP) + False\ Negative(FN)} \quad (4.1)$$

**Precision**

Precision, in the context of classification, is defined as the ratio of True Positives (TP) to the sum of True Positives and False Positives (FP). It quantifies the proportion of correctly predicted positive instances out of all instances that were predicted as positive.

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP) + FalsePositive(FP)} \quad (4.2)$$

**Recall**

Recall measures how well the model detects true positives, representing the proportion of correctly classified positive instances out of all actual positives.

$$Recall = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Negative(FN)} \quad (4.3)$$

**F-score**

The F-score, or F1-score, assesses a model's accuracy by merging its precision and recall metrics. It's computed as the harmonic mean of precision and recall, offering a balanced evaluation of the model's performance. Depending on the situation, adjustments can be made to emphasize either precision or recall in the F-score calculation.

$$F1 = 2.\frac{Precision.Recall}{Precision + Recall} \tag{4.4}$$

**Mean Absolute Error (MAE)**

Mean Absolute Error (MAE) provides an estimate of errors in the discrepancy between pairs describing the same occurrence.

$$MAE = \frac{\sum_{i=1}^{n} y_i - x_i}{n} \tag{4.5}$$

Here,

$y_i$ = Predicted value

$x_i$ = Actual value

n = Total number of object

**Mean Squared Error (MSE)**

The mean squared error (MSE) or mean squared deviation (MSD) calculates the average of the squared discrepancies between predicted and actual values, providing a measure of the overall error in the estimator's predictions.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - x_i)^2}{n} \tag{4.6}$$

Here,

$y_i$ = Predicted value

$x_i$ = Actual value

n = Total number of object

**Root Mean Square Error (RMSE)**

The root mean square error (RMSE) represents the standard deviation of prediction errors or residuals. These residuals indicate the distances of the data points from the regression line.

RMSE quantifies the dispersion of these residuals, essentially showing how closely the data points are clustered around the line of best fit.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - x_i)^2}{n}} \tag{4.7}$$

Here,

$y_i$ = Predicted value

$x_i$ = Actual value

n = Total number of object

**Relative Absolute Error (RAE)**

Relative Absolute Error (RAE) compares the mean error (residual) of a model to the errors generated by a simple or naive model, expressed as a ratio. A model is considered reasonable if its RAE is less than one, indicating superior performance compared to the simplistic model.

$$RAE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - x_i)^2}{\sum_{i=1}^{n} x_i^2}} \tag{4.8}$$

Here,

$y_i$ = Predicted value

$x_i$ = Actual value

n = Total number of object

**Root Relative Squared Error (RRSE)**

The Root Relative Squared Error (RRSE) is calculated as the square root of the sum of squared errors of a forecasting model, normalized by the sum of squared errors of a simple model. In other words, it is the square root of the Relative Squared Error (RSE).

$$RRSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - x_i)^2}{\sum_{i=1}^{n}(x_i - x_m)^2}} \tag{4.9}$$

Here,

$y_i$ = Predicted value

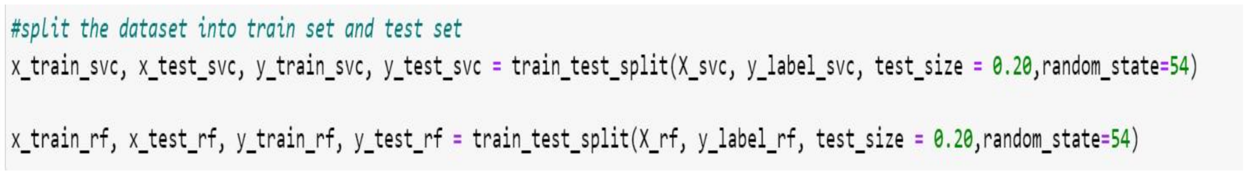
$x_i$ = Actual value

$x_m$ = Mean of actual value

n = Total number of object

## 4.4 Results

The experiments were performed using the RSCB PDB dataset. We split the dataset into training and testing sets, with 80% designated for training and 20% for testing purposes. This division was maintained consistently for both SVM and Random Forest models, as depicted in Figure 4.1.



```
#split the dataset into train set and test set
x_train_svc, x_test_svc, y_train_svc, y_test_svc = train_test_split(X_svc, y_label_svc, test_size = 0.20,random_state=54)

x_train_rf, x_test_rf, y_train_rf, y_test_rf = train_test_split(X_rf, y_label_rf, test_size = 0.20,random_state=54)
```

**Figure 4.1:** Splitting the Dataset into training and test set

The Table 4.1 shows the hyperparameter tweaking for the dataset.

**Table 4.1:** Hyperparameter Tuning of GNN

| Support Vector Machine | | Random Forest | |
|---|---|---|---|
| C | $\gamma$ | n_estimators | random_state |
| 1.5 | 0.1 | 100 | 42 |

In our proposed model, we utilize a sliding window approach, where varying window sizes result in different accuracies. Table 4.2 illustrates how accuracy varies with different sliding window sizes.

**Table 4.2:** Variations of Accuracy with Various Window Size

| Window Size | Accuracy |
|---|---|
| 5 | 90.93% |
| 7 | 91.42% |
| 9 | 91.60% |
| 11 | 91.81% |
| 13 | 92.13% |
| 15 | 92.23% |
| 17 | 92.30% |
| 19 | 92.43% |
| 21 | **92.62%** |
| 23 | 92.59% |

According to the Table 4.2, the dataset with window size 21 has a maximum accuracy of 92.62%. Different types of performance analysis are measured and shown in the Table 4.3, visualization in Figure 4.2 and the confusion matrix in Figure 4.3 to evaluate the proposed model.

**Table 4.3:** Performance Analysis

| Precision | Recall | F-score | MAE | MSE | RMSE | RAE | RRSE |
|---|---|---|---|---|---|---|---|
| 0.93 | 0.93 | 0.92 | 0.24 | 1.17 | 1.08 | 0.24 | 0.64 |

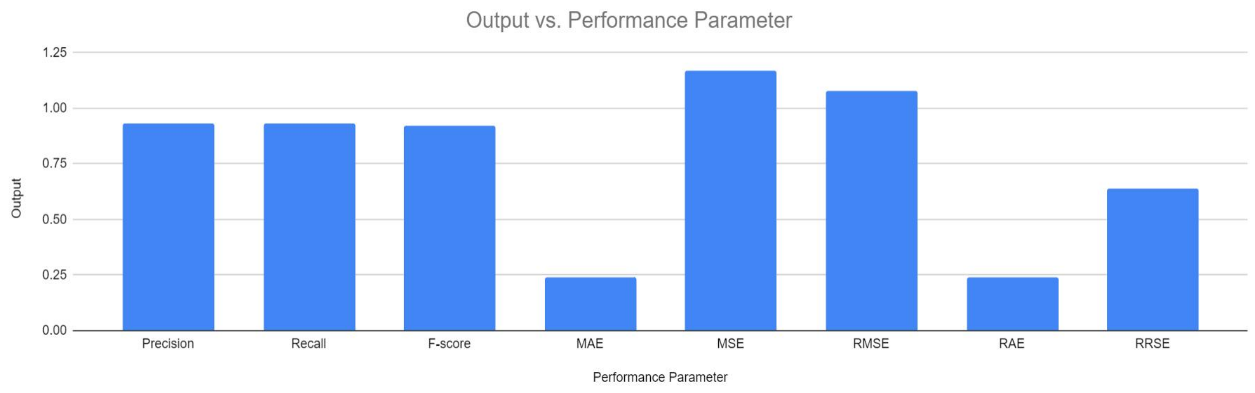


**Figure 4.2:** Visualization of different types of performance parameters

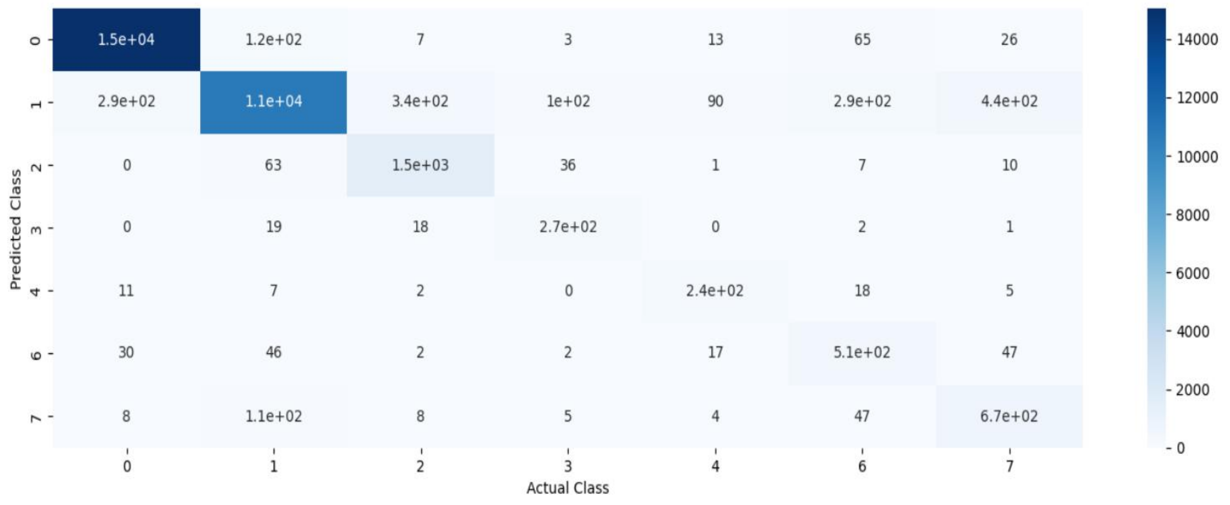
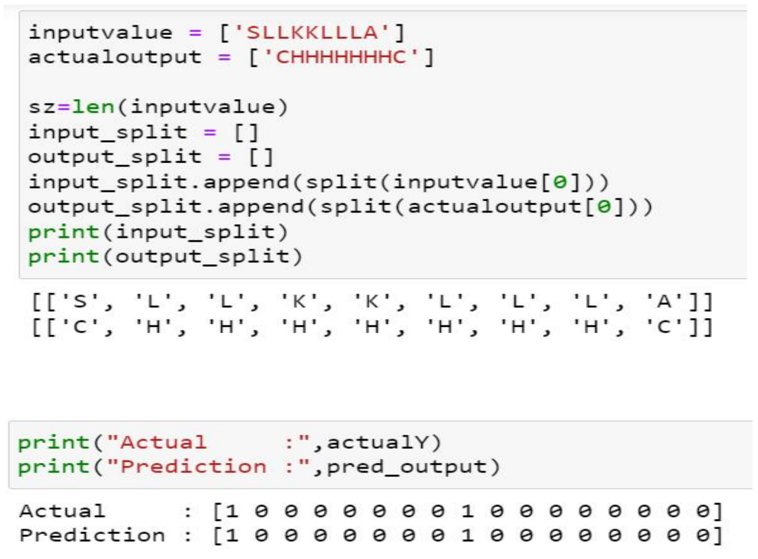


**Figure 4.3:** Confusion Matrix of The Model

Here are two examples of secondary structure prediction from the primary sequences predicted by the model in Figure 4.4, 4.5:



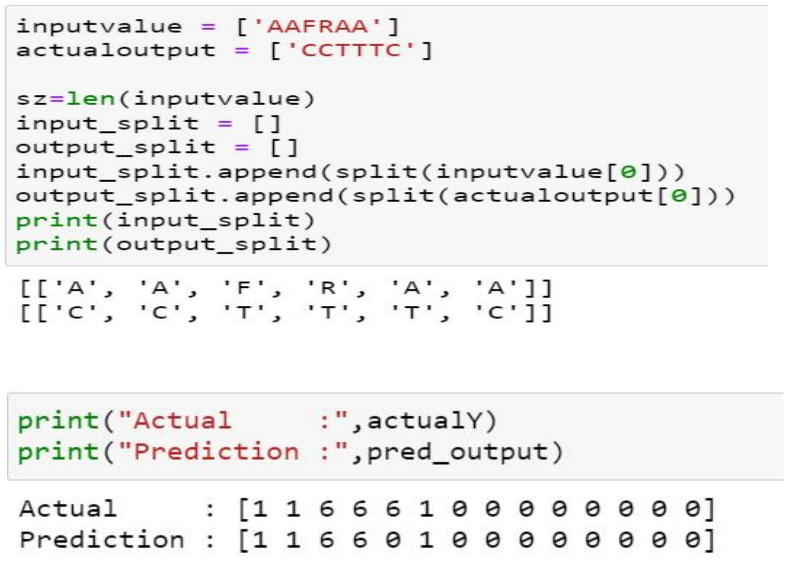
```
inputvalue = ['SLLKKLLLA']
actualoutput = ['CHHHHHHHC']

sz=len(inputvalue)
input_split = []
output_split = []
input_split.append(split(inputvalue[0]))
output_split.append(split(actualoutput[0]))
print(input_split)
print(output_split)
```

```
[['S', 'L', 'L', 'K', 'K', 'L', 'L', 'L', 'A']]
[['C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'C']]
```

```
print("Actual     :",actualY)
print("Prediction :",pred_output)
```

```
Actual     : [1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
Prediction : [1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
```

**Figure 4.4:** Prediction of primary sequence 'SLLKKLLLA'



```
inputvalue = ['AAFRAA']
actualoutput = ['CCTTTC']

sz=len(inputvalue)
input_split = []
output_split = []
input_split.append(split(inputvalue[0]))
output_split.append(split(actualoutput[0]))
print(input_split)
print(output_split)
```

```
[['A', 'A', 'F', 'R', 'A', 'A']]
[['C', 'C', 'T', 'T', 'T', 'C']]
```

```
print("Actual     :",actualY)
print("Prediction :",pred_output)
```

```
Actual     : [1 1 6 6 6 1 0 0 0 0 0 0 0 0]
Prediction : [1 1 6 6 0 1 0 0 0 0 0 0 0 0]
```

**Figure 4.5:** Prediction of primary sequence 'AAFRAA'

## 4.5 Comparative Analysis

Table 4.4 uses a comparison with other different approaches in protein secondary structure prediction to show that graph neural network (GNN) has reached a standard level of development.

**Table 4.4:** Comparison with Various Model

| Type | Year | Accuracy | Ref. |
|---|---|---|---|
| CNN | 2014 | 66.7 | [52] |
| LSTM | 2014 | 67.4 | [53] |
| Convolutional neural fields | 2016 | 68.3 | [54] |
| BRNN | 2017 | 68.5 | [55] |
| Multi-scale CNN | 2018 | 70.3 | [56] |
| CNN-BiLSTM | 2019 | 70.4 | [57] |
| SAINT | 2019 | 73.3 | [58] |
| GNN(Only for smaller length) | 2021 | 89.73 | [59] |
| **GNN with Ensemble** | CurrentMethod | **92.62** | |

From the Table 4.4, We can get to this conclusion after examining how well GNN performs. Most studies retain an accuracy of 60-70 percent, and one previous GNN model retain an accuracy of 89.73% but it was only for smaller length and performs poor for larger length data. However our proposed model achieves an accuracy of 92.62% and it was for both larger and smaller length data. Because we have used SVM for smaller length data and Random Forest for larger length data.

## 4.6 Summary

This section concentrates on the suggested method's performance analysis. The simulation results indicate that the suggested method outperforms other related methods in the majority of scenarios.

# CHAPTER V

# Socio-Economic Impact and Sustainability

## 5.1 Impact on Societal, Health, Legal and Cultural Issues

### 5.1.1 Societal Impact

- **Scientific advancement:** Accurate protein secondary structure prediction contributes to advancements in molecular biology, biochemistry, and biophysics. This knowledge helps scientists understand the structure and function of proteins, leading to breakthroughs in drug design, disease understanding, and biotechnology.
- **Education and awareness:** Improved predictions can enhance educational programs in the life sciences, promoting a better understanding of the molecular basis of life. This knowledge can be disseminated to the public, fostering scientific literacy and interest in related fields.

### 5.1.2 Health Impact

- **Drug discovery and design:** Accurate predictions aid in designing drugs that target specific proteins involved in diseases. This can lead to the development of more effective and targeted therapeutic interventions, potentially reducing side effects and improving patient outcomes.
- **Disease understanding:** Protein structure prediction helps in understanding the molecular mechanisms underlying various diseases, providing insights into potential diagnostic markers and treatment strategies.

### 5.1.3 Legal and Ethical Considerations

In the legal and ethical aspects of the protein prediction project, there are many important considerations. On the legal side, it's crucial to ensure that the software adheres to data protection laws, especially if it involves handling personal or sensitive information. Intellectual property rights should be respected to avoid legal conflicts.

On the ethical side, transparency in the software's predictions is crucial. Users should be informed about the limitations and uncertainties associated with the tool. Privacy is a crucial priority, especially when handling genetic or health-related data. It is ethically required to obtain informed consent from people who contribute to datasets.

### 5.1.4 Cultural Impact

It is culturally significant that the development of a software program that can predict protein secondary structure from primary sequence has accelerated scientific study, biotechnology, and healthcare breakthroughs. This invention promotes a collaborative culture among scientists worldwide, facilitating knowledge exchange and integrative approaches to challenging biological problems. Moreover, the software's impact on drug discovery, personalized medicine, and industry standards underscores its potential to shape the cultural landscape of molecular biology, influencing how we approach medical research, education, and ethical considerations in the life sciences.

## 5.2 Impact on the environment and sustainability

While the impact of a protein secondary structure prediction software on the environment is indirect, it's contributions to biotechnological and pharmaceutical advancements can influence sustainability in various ways. By facilitating more targeted drug discovery, the software may contribute to the development of environmentally friendly pharmaceuticals. Additionally, improved understanding of protein structures can enhance the design of proteins for industrial and biotechnological applications, potentially leading to more sustainable production processes. Such software can complement the larger cultural shift in the biological sciences toward computational and data-driven methods, which can promote an efficient and resource-optimized research environment. Even though the software itself doesn't directly impact the environment, it's role in improving processes aligns with the growing interest in sustainable practices in scientific research and industry.

# CHAPTER VI

# Addressing Complex Engineering Problems and Activities

## 6.1 Addressing Complex Engineering Problems

**Table 6.1:** Complex Engineering Problems of the Project

| **Complex Engineering Problems** | | **Addressing the Complex Engineering Problems in the Project** |
|---|---|---|
| WP1 | Depth of knowledge required (WK3-WK5, WK8) | Literature review is required to study existing methods, requires the knowledge of Graph, SVM, Random Forest and ensemble learning. |
| WP2 | Range of conflicting requirements | Prediction accuracy vs. computational efficiency, data quantity vs. data quality, length of the primary sequence, number of state of the secondary structure etc. |
| WP3 | Depth of analysis required | Requires the analysis of amino acids, primary and secondary structure of proteins and different machine learning algorithms. |
| WP4 | Familiarity of issues | Computer science and Engineering graduates are not typically exposed to the issues related to protein structure and thus the work involves infrequently encountered issues. |
| WP5 | Extent of applicable codes | Data preprocessing, feature extraction, applying machine learning models (SVM and Random Forest) and training the datasets. |
| WP6 | Extent of stakeholder involvement | Involves varies stakeholders like bioinformaticians, computational biologists, molecular biologists, software developers, engineers etc. |
| WP7 | Interdependence | Interdependency between programming and protein primary sequence in a software to predict the secondary structure. |

## 6.2 Addressing Complex Engineering Activities

**Table 6.2:** Complex Engineering Activities of the Project

| **Complex Engineering Activities** | | **Addressing the Complex Engineering Activities in the Project** |
|---|---|---|
| EA1 | Range of resources | The project needs diverse resources including computational hardware, data processing and visualization tools, information and technologies. |
| EA2 | Level of interactions | A good level of interaction is needed among the students and the drug specialist to know about the structure of various proteins used in drug. |
| EA3 | Innovation | A degree of innovation is required to design a less complexity and more accurate prediction algorithm. |
| EA4 | Consequences to society / environment | Understanding the protein's secondary structure will help in drug design and development, disease understanding and treatment in the modern society, sharing and accessing protein data, prediction models and results contribute to the environmental impact of digital technologies. |
| EA5 | Familiarity | Can play a great role in DNA matching such as identification of dead or burnt bodies, identification of parents of a child etc. |

# CHAPTER VII

# Conclusion, Limitations and Future Works

## 7.1 Conclusions

This study explores the prediction of protein secondary structures using Graph Neural Networks (GNN), showing promising results. Unlike traditional GNNs, this model comprises multiple interconnected components that collectively form the final predictive model.

The process involves creating a graph from the data, processing it with neural network techniques, and aggregating information from neighboring nodes. Techniques like conformation parameters, window padding, and orthogonal encoding are applied for optimization. Experimentation with different window padding sizes is conducted to find the best configurations. Additionally, hyperparameters for Random Forest and SVM models are fine-tuned using a trial-and-error approach to ensure optimal performance.

## 7.2 Limitations

Training larger datasets can be time-consuming due to the model's complexity. Our understanding of the relationship between amino acid sequences and secondary structures is incomplete, posing challenges for accurate predictions. Furthermore, proteins' dynamic nature and conformational changes are essential for their functionality, which Graph Neural Networks (GNN) may struggle to capture effectively. GNNs may also face difficulties in incorporating long-range dependencies crucial for predicting secondary structures accurately.

## 7.3 Future Works

The future direction of this thesis can be outlined as follows:

- Enhanced accuracy can be attained by integrating neural networks with complementary models.
- Exploring alternative optimization techniques to further improve accuracy.
- Introducing 3D graphing of secondary structures into the proposed model to enhance its capabilities.
- Intend to improve this suggested model by including protein tertiary structure prediction.

# References

[1] D. Li, T. Li, P. Cong, W. Xiong, and J. Sun. “A novel structural position-specific scoring matrix for the prediction of protein secondary structures”. In: *Bioinformatics* 28.1 (2012), pp. 32–39.

[2] J. Bła˙zewicz, P. Łukasiak, and S. Wilk. “New machine learning methods for prediction of protein secondary structures”. In: *Control and Cybernetics* 36 (2007), pp. 183–201.

[3] C. Bystroff, V. Thorsson, and D. Baker. “HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins”. In: *Journal of molecular biology* 301.1 (2000), pp. 173–190.

[4] C. Kathuria, D. Mehrotra, and N. K. Misra. “Predicting the protein structure using random forest approach”. In: *Procedia computer science* 132 (2018), pp. 1654–1662.

[5] W. Wardah, M. Khan, A. Sharma, and M. A. Rashid. “Protein secondary structure prediction using neural networks and deep learning: A review”. In: *Computational Biology and Chemistry* 81 (2019), pp. 1–8. ISSN: 1476-9271. DOI: https://doi. org/10.1016/j.compbiolchem.2019.107093. URL: https://ieeexplore.ieee.org/document/8118041

[6] S. Rout, S. Mishra, and S. Mishra. “A review on application of artificial neural network(ANN) on protein secondary structure prediction”. In: Feb. 2017, pp. 1–6. DOI: 10.1109/ICECCT.2017.8118041.

[7] A. McNaught and A Wilkinson. *IUPAC Compendium of Chemical Terminology, 2nd edn.(the “Gold Book”) Blackwell Scientific Publications*. 1997.

[8] C. Branden and J. Tooze. “Introduction to protein structure, Garland Pub”. In: *Inc. NewYork* (1991).

[9] J Kyte. “Structure in Protein Chemistry Garland Publishing”. In: *Inc., New York, and London* 201 (1995).

[10] H. Wang and J. Leskovec. “Unifying graph convolutional neural networks and label propagation”. In: *arXiv preprint arXiv:2002.06755* (2020).

[11] A. Rossi, M. Tiezzi, G. M. Dimitri, M. Bianchini, M. Maggini, and F. Scarselli. “Inductive–transductive learning with graph neural networks”. In: *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer. 2018, pp. 201–212.

[12] D. Roobaert and M. M. Van Hulle. “View-based 3D object recognition with support vector machines”. In: *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No. 98TH8468)*. IEEE. 1999, pp. 77–84.

[13] M. Schmidt and H. Gish. “Speaker identification via support vector classifiers”. In: *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*. Vol. 1. IEEE. 1996, pp. 105–108.

[14] H. Drucker, D. Wu, and V. N. Vapnik. “Support vector machines for spam categorization”. In: *IEEE Transactions on Neural networks* 10.5 (1999), pp. 1048–1054.

[15] J. Guo, H. Chen, Z. Sun, and Y. Lin. “A novel method for protein secondary structure prediction using dual-layer SVM and profiles”. In: *PROTEINS: Structure, Function, and Bioinformatics* 54.4 (2004), pp. 738–743.

[16] W. Kabsch and C. Sander. “Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features”. In: *Biopolymers: Original Research on Biomolecules* 22.12 (1983), pp. 2577–2637.

[17] A. Patle and D. S. Chouhan. “SVM kernel functions for classification”. In: *2013 International Conference on Advances in Technology and Engineering (ICATE)*. IEEE. 2013, pp. 1–9.

[18] S. Hua and Z. Sun. “A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach”. In: *Journal of molecular biology* 308.2 (2001), pp. 397–407.

[19] S. C. Schmidler, J. S. Liu, and D. L. Brutlag. “Bayesian segmentation of protein secondary structure”. In: *Journal of computational biology* 7.1-2 (2000), pp. 233–248.

[20] W. Chu, Z. Ghahramani, and D. L. Wild. “A graphical model for protein secondary structure prediction”. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 21.

[21] N. Qian and T. J. Sejnowski. “Predicting the secondary structure of globular proteins using neural network models”. In: *Journal of molecular biology* 202.4 (1988), pp. 865–884.

[22] D. T. Jones. “Protein secondary structure prediction based on position-specific scoring matrices”. In: *Journal of molecular biology* 292.2 (1999), pp. 195–202.

[23] D. W. Buchan, F. Minneci, T. C. Nugent, K. Bryson, and D. T. Jones. “Scalable web services for the PSIPRED Protein Analysis Workbench”. In: *Nucleic acids research* 41.W1 (2013), W349–W357.

[24] E. Faraggi, T. Zhang, Y. Yang, L. Kurgan, and Y. Zhou. “SPINE X: improving protein secondary structure prediction by multistep learning coupled with prediction of solvent accessible surface area and backbone torsion angles”. In: *Journal of computational chemistry* 33.3 (2012), pp. 259–267.

[25] K. Asai, S. Hayamizu, and K. Handa. “Prediction of protein secondary structure by the hidden Markov model”. In: *Bioinformatics* 9.2 (1993), pp. 141–146.

[26] Z. Aydin, Y. Altunbasak, and M. Borodovsky. “Protein secondary structure prediction for a single-sequence using hidden semi-Markov models”. In: *BMC bioinformatics* 7.1 (2006), pp. 1–15.

[27] P. Baldi, S. Brunak, P. Frasconi, G. Soda, and G. Pollastri. “Exploiting the past and the future in protein secondary structure prediction”. In: *Bioinformatics* 15.11 (1999), pp. 937–946.

[28] J. Chen and N. Chaudhari. “Cascaded bidirectional recurrent neural networks for protein secondary structure prediction”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4.4 (2007), pp. 572–582.

[29] C. Mirabello and G. Pollastri. “Porter, PaleAle 4.0: high-accuracy prediction of protein secondary structure and relative solvent accessibility”. In: *Bioinformatics* 29.16 (2013), pp. 2056–2058.

[30] B. Zhang, J. Li, and Q. Lü. “Prediction of 8-state protein secondary structures by a novel deep learning architecture”. In: *BMC bioinformatics* 19.1 (2018), pp. 1–13.

[31] A. A. Salamov and V. V. Solovyev. “Protein secondary structure prediction using local alignments”. In: *Journal of molecular biology* 268.1 (1997), pp. 31–36.

[32] D. Frishman and P. Argos. “Seventy-five percent accuracy in protein secondary structure prediction”. In: *Proteins: Structure, Function, and Bioinformatics* 27.3 (1997), pp. 329–335.

[33] D Mount. “Bioinformatics: sequence andgenome analysis”. In: *COMSDrig Hiror LiOrcio)’PaSS, New York* (2001), pp. 3–5.

[34] C. Bystroff, V. Thorsson, and D. Baker. “HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins”. In: *Journal of molecular biology* 301.1 (2000), pp. 173–190.

[35] B. Rost and C. Sander. “Improved prediction of protein secondary structure by use of sequence profiles and neural networks”. In: *Proceedings of the National Academy of Sciences* 90.16 (1993), pp. 7558–7562.

[36] L. Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.

[37] J. D. Otero-Cruz, D. A. Torres-Núñez, C. A. Báez-Pagán, and J. A. Lasalde-Dominicci. “Fourier transform coupled to tryptophan-scanning mutagenesis: Lessons from its application to the prediction of secondary structure in the acetylcholine receptor lipidexposed transmembrane domains”. In: *Biochimica et Biophysica Acta (BBA)Proteins and Proteomics* 1784.9 (2008), pp. 1200–1207.

[38] G.-Z. Zhang, D.-S. Huang, and H.-Q. Wang. “Protein secondary structure prediction based on the amino acids conformational classification and neural network technique”. In: *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 5. IEEE. 2004, pp. V–573.

[39] H. Kim and H. Park. “Protein secondary structure prediction based on an improved support vector machines approach”. In: *Protein engineering* 16.8 (2003), pp. 553– 560.

[40] B. Rost and C. Sander. “Combining evolutionary information and neural networks to predict protein secondary structure”. In: *Proteins: Structure, Function, and Bioinformatics* 19.1 (1994), pp. 55–72.

[41] S. K. Riis and A. Krogh. “Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments”. In: *Journal of Computational Biology* 3.1 (1996), pp. 163–183.

[42] J.-M. Chandonia and M. Karplus. “The importance of larger data sets for protein secondary structure prediction with neural networks”. In: *Protein Science* 5.4 (1996), pp. 768–774.

[43] C. Cortes and V. Vapnik. “Support-vector networks”. In: *Machine learning* 20.3 (1995), pp. 273–297.

[44] R. Soheilifard and C. A. Toussi. “On the contribution of normal modes of elastic network models in prediction of conformational changes”. In: *2016 23rd Iranian Conference on Biomedical Engineering and 2016 1st International Iranian Conference on Biomedical Engineering (ICBME)*. IEEE. 2016, pp. 263–266.

[45] H.-J. Hu, Y. Pan, R. Harrison, and P. C. Tai. “Improved protein secondary structure prediction using support vector machine with a new encoding scheme and an advanced tertiary classifier”. In: *IEEE Transactions on NanoBioscience* 3.4 (2004), pp. 265– 271.

[46] A. Hossain, F. Zaman, M Nasser, and M. M. Islam. “Comparison of GARCH, neural network and support vector machine in financial time series prediction”. In: *International Conference on Pattern Recognition and Machine Intelligence*. Springer. 2009, pp. 597–602.

[47] M. Spencer, J. Eickholt, and J. Cheng. “A deep learning network approach to ab initio protein secondary structure prediction”. In: *IEEE/ACM transactions on computational biology and bioinformatics* 12.1 (2014), pp. 103–112.

[48] Z. Zhang, J. Leng, L. Ma, Y. Miao, C. Li, and M. Guo. “Architectural Implications of Graph Neural Networks”. In: *IEEE Computer Architecture Letters* 19.1 (2020), pp. 59–62. DOI: 10.1109/LCA.2020.2988991.

[49] M. Zitnik and J. Leskovec. “Predicting multicellular function through multi-layer tissue networks”. In: *Bioinformatics* 33.14 (2017), pp. i190–i198.

[50] M. Zitnik, M. Agrawal, and J. Leskovec. “Modeling polypharmacy side effects with graph convolutional networks”. In: *Bioinformatics* 34.13 (2018), pp. i457–i466.

[51] K. Lin, A. C. May, and W. R. Taylor. “Amino acid encoding schemes from protein structure alignments: Multi-dimensional vectors to describe residue types”. In: *Journal of theoretical biology* 216.3 (2002), pp. 361–365.

[52] R. Fai, M. Hassan, Y. F. Chin, and M. Mohamad. “Optimized Local Protein Structure with Support Vector Machine to Predict Protein Secondary Structure”. In: vol. 295.Jan. 2012, pp. 333–342. ISBN: 978-3-642-32825-1. DOI: 10.1007/978-3-642- 32826-8_34.

[52] J. Zhou and O. Troyanskaya. “Deep Supervised and Convolutional Generative Stochastic Network for Protein Secondary Structure Prediction”. In: *31st International Conference on Machine Learning, ICML 2014* 2 (Mar. 2014).

[53] S. Sønderby and O. Winther. “Protein Secondary Structure Prediction with Long Short Term Memory Networks”. In: (Dec. 2014).

[54] S. Wang, J. Peng, J. Ma, and J. Xu. “Protein Secondary Structure Prediction Using Deep Convolutional Neural Fields”. In: *Scientific reports* 6 (Jan. 2016). DOI: 10 . 1038/srep18962.

[55] A. Johansen, C. Sønderby, S. Sønderby, and O. Winther. “Deep Recurrent Conditional Random Field Network for Protein Secondary Prediction”. In: Aug. 2017, pp. 73–78. DOI: 10.1145/3107411.3107489.

[56] J. Zhou, H. Wang, Z. Zhao, R. Xu, and Q. Lu. “CNNHPSS: Protein 8-class secondary structure prediction by convolutional neural network with highway”. In: *BMC Bioinformatics* 19 (May 2018). DOI: 10.1186/s12859-018-2067-8.

[57] E. Asgari, N. Poerner, A. McHardy, and M. Mofrad. “DeepPrime2Sec: Deep Learning for Protein Secondary Structure Prediction from the Primary Sequences”. In: (July 2019). DOI: 10.1101/705426.

[58] M. Uddin, S. Mahbub, M. Rahman, and M. Bayzid. “SAINT: Self-Attention Augmented Inception-Inside-Inception Network Improves Protein Secondary Structure Prediction”. In: (Sept. 2019). DOI: 10.1101/786921.

[59] T. H. Nahid, F. A. Jui and P. C. Shill, "Protein Secondary Structure Prediction using Graph Neural Network," 2021 5th International Conference on Electrical Information and Communication Technology (EICT), 2021, pp. 1-6, doi: 10.1109/EICT54103.2021.9733590.