Yasin Görmez

A Ph.D. Thesis

AGU 2022

DEVELOPING DEEP LEARNING MODELS FOR PROTEIN STRUCTURE PREDICTION

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Ph. D.

By
Yasin Görmez
October 2022

DEVELOPING DEEP LEARNING MODELS FOR PROTEIN STRUCTURE PREDICTION

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Ph. D.

By
Yasin Görmez
October 2022

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name-Surname: Yasin Görmez
Signature :

REGULATORY COMPLIANCE

Ph.D. thesis titled Developing Deep Learning Models for Protein Structure Prediction has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By	Advisor
Yasin GÖRMEZ	Assoc. Prof. Zafer AYDIN

Head of the Electrical and Computer Engineering Program

Assoc. Prof. Zafer AYDIN

Signature

ACCEPTANCE AND APPROVAL

Ph.D. thesis titled Developing Deep Learning Models for Protein Structure Prediction and prepared by Yasin Görmez has been accepted by the jury in the Electrical and Computer Engineering Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

12 /10 / 2022

(Thesis Defense Exam Date)

JURY:

Advisor : Assoc. Prof. Zafer AYDIN

Member : Assist. Prof. Burcu BAKIR GÜNGÖR

Member : Assist. Prof. Özkan Ufuk NALBANTOĞLU

Member : Assoc. Prof. Mete ÇELİK

Member : Assist. Prof. Bekir Hakan AKSEBZECİ

APPROVAL:

The acceptance of this Ph.D. thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated ….. /….. / …….. and numbered ……..…… .

…….. /…….. / ……..

(Date)

Graduate School Dean
Prof. Dr. İrfan ALAN

ABSTRACT

DEVELOPING DEEP LEARNING MODELS FOR PROTEIN STRUCTURE PREDICTION

Yasin GÖRMEZ
Ph.D. in Electrical and Computer Engineering Department
Advisor: Assoc. Prof. Zafer AYDIN

October 2022

The three-dimensional structure of a protein provides important clues about the function of that protein. Although there have been many studies on protein structure prediction, the problem has still not been solved completely. As it is very difficult to predict the three-dimensional structure of a protein directly, predictions of structural properties of proteins such as secondary structure, solvent accessibility, and torsion angles are carried out first, which are later used as inputs to more elaborate structure estimation tasks. In this thesis, novel deep learning models have been developed by using convolutional neural networks (CNN), graph convolutional networks (GCN) and long-short-term memory (LSTM) recurrent neural networks to predict secondary structure, solvent accessibility and torsion angles of proteins. A rich feature set formed by using PSI-BLAST, HHBlits, physicochemical properties, structural profile matrices, AA index values, and graphs representing the relationship between amino acids were used as inputs to the models. In the first study, a deep learning model was developed by using CNN and GCN layers for secondary structure prediction. In the second study, LSTM layers were added to the first model, which was extended to make solvent accessibility and torsion angle predictions as well using the multi-task learning approach. In both studies, graphs were generated using neighborhood relations between amino acids. In the last study, a novel U-net-based model was designed for secondary structure prediction using CNN, GCN, and LSTM layers. The graph matrices used as input to GCN layers were obtained by using protein contact map prediction. All models were trained, optimized and tested on benchmark data sets. Improvements were obtained in accuracy as compared to the state-of-the-art.

Keywords: Protein Structure Prediction, Deep Learning, Protein Secondary Structure Prediction, Solvent Accessibility Prediction, Torsion Angle Prediction

ÖZET

PROTEİN YAPI TAHMİNİ İÇİN DERİN ÖĞRENME MODELLERİNİN GELİŞTİRİLMESİ

Yasin GÖRMEZ
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı Doktora
Tez Yöneticisi: Doç. Dr. Zafer AYDIN
Ekim-2022

Bir proteinin üç boyutlu yapısı, o proteinin fonksiyonu hakkında önemli ipuçları sunmaktadır. Literatürde protein yapı tahmini yapan birçok çalışma bulunmasına rağmen bu problem henüz tam olarak çözümlenememiştir. Üç boyutlu protein yapı tahmininin direkt olarak yapılması çok zor olduğundan ilk etapta ikincil yapı, çözücü erişilirlik ve burulma açıları gibi yapısal özellikler tahmin edilir ve daha karmaşık yapı tahmin algoritmalarına girdi olarak gönderilir. Bu tezde, ikincil yapı, çözücü erişilirlik ve burulma açıları tahminleri için evrişimsel sinir ağları (ESA), çizge evrişimsel ağlar (ÇEA) ve uzun kısa vadeli hafıza (UKVH) temelli tekrarlayan yapay sinir ağları kullanılarak özgün derin öğrenme modelleri geliştirilmiştir. PSI-BLAST, HHBlits, fiziko kimyasal özellikler, yapısal profil matrisleri ve AAindex parametreleri kullanılarak oluşturulan zengin bir öznitelik seti ve amino asitler arasındaki ilişkinin temsil edildiği çizgeler modellerde girdi olarak kullanılmıştır. İlk çalışmada, ikincil yapı tahmini için ESA ve ÇEA kullanılarak özgün bir model oluşturulmuştur. İkinci çalışmada, ilk modele UKVH katmanları da eklenmiş ve model çok görevli öğrenme yaklaşımı sayesinde çözücü erişilirlik ve burulma açı tahminleri de yapacak şekilde güncellenmiştir. Her iki çalışmada da ÇEA modellerinin girdileri olan çizgeler amino asitler arası komşuluk ilişkisi kullanılarak oluşturulmuştur. Son çalışmada ESA, ÇEA ve UKVH kullanılarak U-net tabanlı özgün bir model ikincil yapı tahmini için tasarlanmıştır. Bu çalışmada girdi olarak kullanılan çizge matrisi protein temas haritası tahmini kullanılarak elde edilmiştir. Tüm modeller güncel veri kümelerinde eğitilmiş, optimize edilmiş ve test edilmiştir. Literatürdeki yöntemlerden daha başarılı sonuçlar elde edilmiştir.

Anahtar kelimeler: Protein Yapı Tahmini, Derin Öğrenme, Protein İkincil Yapı Tahmini, Çözücü Erişilirlik Tahmini, Burulma Açısı Tahmini

Acknowledgements

I would like to thank to my advisor Assoc. Prof. Zafer AYDIN, who encouraged me work in bioinformatics and always supported me during the thesis, for his contribution in this thesis. In addition to this, I would like to thank my family because they supported me during the whole life.

The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

NMR	Nuclear Magnetic Resonance
PSSM	Position Specific Scoring Matrix
PSSP	Protein Secondary Structure Prediction
SAP	Solvent Accessibility Prediction
TAP	Torsion Angle Prediction
FCL	Fully Connected Layers
CNN	Convolutional Neural Networks
GCN	Graph Convolutional Networks
biLSTM	bi-directional Long Short-Term Memory
ω	Omega
φ	Phi
Ψ	Psi
PDB	Protein Data Bank
SPM	Structural Profile Matrix

To My Father

Chapter 1

Introduction

Proteins have a very crucial importance for living organisms. Thus, having knowledge about the functions of proteins is very important for human life. Proteins are an important class of macromolecules found in every organism, formed as a result of the sequential linking of amino acids by peptide bonds [1]. There are 20 common amino acids in nature. Amino acids are organic compounds consisting of an amine group (-NH2), a carboxyl group (-COOH), and a side chain molecule (R) attached to an asymmetric alpha carbon atom (Cα) [2], [3]. Figure 1.1 shows an amino acid molecule. The alpha carbon atom, amine, and carboxyl groups in each amino acid form the backbone portion and they are the same in all amino acid types. In contrast, side chain molecules are unique to each amino acid. Amino acids interact with the condensation reaction and they are linked together by peptide bonds to form a polypeptide chain. This reaction takes place in intracellular organelles called ribosomes, through a process called translation. In this way, proteins needed by our metabolism are synthesized. After proteins are synthesized through the process of translation, they fold into a certain shape in 3-D space in order to perform their biological functions. The forces driving this folding are non-covalent interactions such as hydrogen bonding between protein atoms, ionic interactions, Van Der Waals forces, and hydrophobic stacking. Since there is a close relation between the three-dimensional structure and function of a protein, knowing the structure of a protein gives important clues about its biological function and molecular mechanism.

Large amounts of protein sequence data are generated from large-scale DNA sequencing studies called as the Human Genome Project [4]. However, the structure of many of these proteins has not been experimentally solved. At this point, the proteins whose three-dimensional structure has been solved experimentally constitutes less than 0.6% of the known protein sequences. There are some experimental methods such as X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy to solve the structure of a protein. These methods can be labor intensive, time consuming and costly. In addition, it is not possible to find the structure of some proteins (e.g. some membrane

proteins) experimentally. In cases where experimental methods are insufficient, prediction of protein structure by computational methods is an effective and efficient approach.

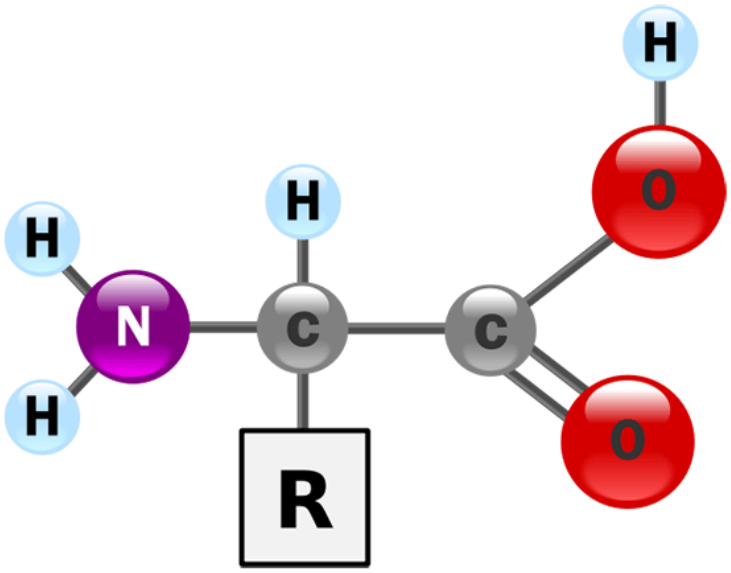


Figure 1.1 Structure of an amino acid [5].

Protein structure prediction refers to the three-dimensional structure prediction, which is a very challenging problem. Since direct prediction of the 3D structure is difficult, in the first step, various structural properties are predicted such as protein secondary structure prediction (PSSP), solvent accessibility prediction (SAP) and torsion angle prediction (TAP). To date, many machine learning techniques have been developed for PSSP [6]–[14], SAP [15]–[19] and TAP [20]–[25]. Recently, deep learning approaches have been proposed, which have gained increasing popularity and a significant improvement was achieved [26]–[33]. Because of these reasons, in this thesis, several deep learning models were developed for PSSP, SAP and TAP. Fully Connected Layers (FCL), Convolutional Neural Networks (CNN), Graph Convolutional Networks (GCN) and bi-directional Long Short-Term Memory (biLSTM) recurrent neural networks were used in different combinations and numbers. Studies in the literature show that, features in dataset also affect the accuracy of prediction as much as the machine learning model [8], [10], [34]–[37]. Because of this reason, position specific scoring matrices (PSSM) that are calculated using PSI-BLAST [38], profiles that are calculated using HHBlits [39], seven physico-chemical properties of amino acids, physico-chemical features from AAindex [40] and structural profiles for secondary structure and solvent accessibility of proteins were used to generate the feature matrix of the models. Another significant factor that affects the

accuracy of a deep learning model is the set of hyper-parameters. In this thesis, hyper-parameters of the proposed models were optimized using the Bayesian Optimization technique, which is faster and more efficient than the traditional optimization algorithms [41], [42]. The organizations of this thesis after the Introduction chapter is as follows. Chapter 2 explains the protein structure levels, protein secondary structure prediction with literature review, solvent accessibility prediction with literature review, torsion angle prediction with literature review and contact map prediction. Chapter 3 presents the benchmark datasets, feature generation, neural network layers, graph generation, proposed deep learning models and Bayesian optimization. In chapter 4, the experimental results of the proposed models are presented in detail. Finally, the thesis is concluded and future works are discussed in chapter 5.

Chapter 2

Structure of a Protein

The 20 amino acids, which are found in natural proteins and are produced by ribosomes, have different physical and chemical properties. They differ in their electrostatic charge, hydrophobicity, acid dissociation coefficient, size and functional groups. These properties play an important role in determining the structure of proteins [1]. For example, the amino acid Glycine, which has the smallest side chain molecule, has only one hydrogen atom as the side chain. The flexibility of the protein structure increases around this amino acid. The amino acid Cysteine interacts with another Cysteine to form a cross-link, thereby strengthening the protein structure. Since positive and negative charges are dissociated in an amino acid residue, amino acids show polarity. For example, the negatively charged free C=O group is the hydrogen bond acceptor, and the positively charged NH group is the hydrogen bond donor. These groups can interact within the protein structure and contribute to the formation of the protein structure [43]. The structure of proteins can be studied in levels, which are explained in the next section.

2.1 Protein Structure Levels

There are four basic levels of protein structure: primary, secondary, tertiary and quaternary structures. Figure 2.1 shows the structure levels of a protein.

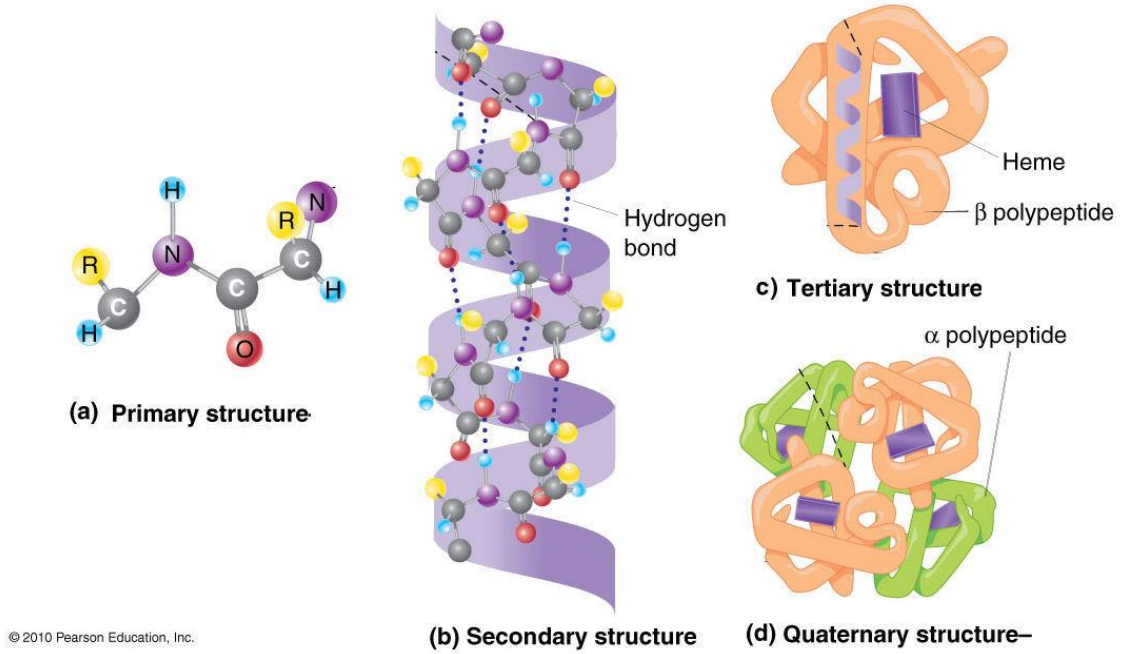


Figure 2.1 Structure levels of a protein [44].

2.1.1 Primary Structure

The straight polymer (polypeptide) chain, in which amino acids are linked by peptide bonds, refers to the primary structure of the protein. In order for the polypeptide chain to transform into biologically active proteins, it must acquire its specific 3D structure. The primary structure of a protein determines its three-dimensional structure. Modifications of amino acids in the primary structure (e.g. substitutions, insertions and deletions) can affect the three-dimensional conformation and function of the protein [45].

2.1.2 Secondary Structure

The secondary structure of a protein is defined by repetitive fold or folding that affects its overall conformation. These folds are formed by hydrogen bonds formed at regular intervals along the polypeptide backbone. The most commonly observed secondary structures are α-helix and β-sheet. Other than the two primary secondary structures, irregular folding units form random loop structures (loop).

α-helix is the main component of tissues such as hair and skin, the hardness of which depends on the number of disulfide bonds between the polypeptide chains that form their structure. In an α-helix, the backbone of the polypeptide twists clockwise to form a spiral. One turn of the helix contains ~3.6 amino acids, which has a length of 5.4 A^o. The R groups are oriented outward from the helix. This structure is fixed by the H bond between

the N atom of the peptide bond and the carbonyl O atom of the 4th amino acid. The α-helix structure is seen in globular proteins [45].

In β-sheet, H bonds are formed between different segments of a polypeptide chain or between segments of two separate polypeptide chains. This structure is commonly seen in fibrous proteins such as silk thread. The β-sheet structure can be parallel or anti-parallel [45].

Loops are structures that do not have regular repetitions such as helix and sheet. They usually exist between the helical and sheet structures and connect these structures [46]. Figure 2.2 shows the three categories of the secondary structure elements.

Figure 2.2 Three categories of the secondary structure elements [47]

2.1.3 Tertiary Structure

The tertiary structure refers to the three-dimensional coordinates of the atoms in a polypeptide chain. It is formed by the folding of the secondary structure elements at a higher level, arises especially from the interactions between the R groups. The main interactions that form the tertiary structure are hydrophobic and Van der Waals interactions. In addition to these, ionic bonds, salt bridges and hydrogen bonds also play a role in the formation of tertiary structure. The three-dimensional conformation of the protein is also supported by disulfide covalent bonds formed between cysteine amino acids [45].

2.1.4 Quaternary Structure

Some proteins contain more than one polypeptide chain. The interactions and bonds formed between these units (also called subunits) form the quaternary structure of the

protein. Note that some proteins may not contain more than one polypeptide chain and therefore may not have a quaternary structure. Hemoglobin is an example to a protein with quaternary structure that contains four chains.

2.2 Solvent Accessibility

Accessible surface area is defined as the surface area of a biomolecule accessible to a solvent such as water. The van der Waals area is proportional to the diameter of the atoms and can be defined as the surfaces of the red circles in figure 2.3. The accessible area is shown as a dashed black line in this figure. When a representative solvent molecule (shown as a blue circle in figure 2.3) travels on the van der Waals surface, the route drawn by the center of the molecule gives the accessible surface. Since some amino acids are in the interior of the protein, they are less accessible to solvent molecules, while amino acids that are closer to the surface as a result of protein folding can interact more with solvent molecules.

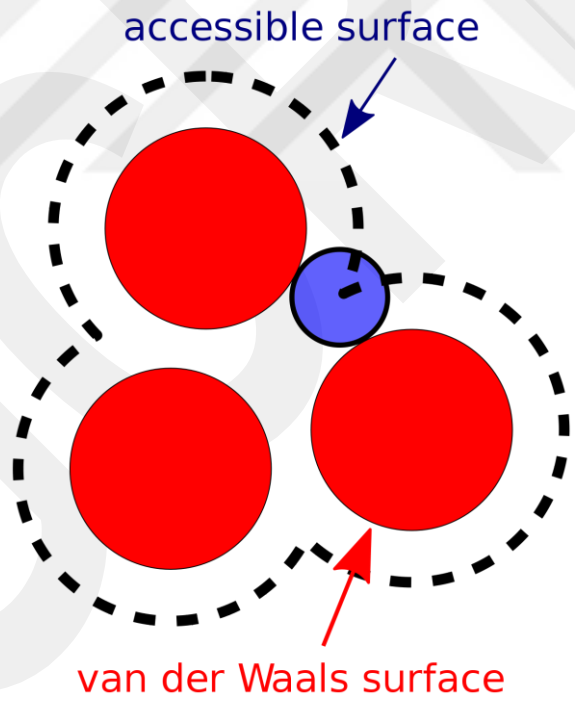


Figure 2.3 Van der Waals space and the space accessible by the solvent [48].

2.3 Torsion Angle

Torsion angles, which are also called dihedral angles, are the angles of rotation around the bonds in the backbone of the protein. There are three types of torsion angles. The

rotation angle of the protein around the peptide bond is called omega (ω), the rotation angle between N and Cα atoms is called phi (ϕ), and the angle between Cα and the carbon atom (C1) of the C=O group is called psi (ψ). Torsion angles of a protein are shown in figure 2.4. The omega (ω) angle does not show much flexibility and usually takes values close to 180 degrees. Phi (ϕ) and psi (ψ) angles can take various values in a certain range. These angles are the internal degrees of freedom of a protein and they control the conformation of the protein.

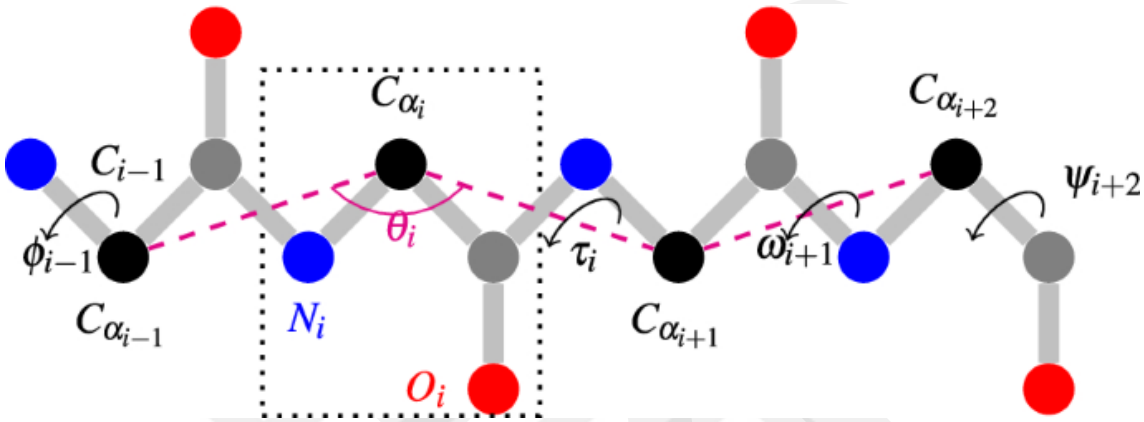


Figure 2.4 Torsion angles of a protein [49].

2.4 Protein Structure Prediction

To date, a lot of research has been done on protein structure prediction. However, the structure prediction problem has not been fully solved yet. The three-dimensional structure prediction is a very difficult problem. Since direct prediction of the structure has various difficulties, it is proceeded step by step. For example, the target protein is first compared to the proteins in the database using various alignment algorithms, and statistical profile matrices, which summarize the frequency of occurrence of amino acids at certain positions, are generated. Then, using these matrices, various features of protein structure such as secondary structure, solvent accessibility, torsion angles and contact map are predicted. In the next step, these properties, profile matrices and other physical principles are used by energy minimization algorithms to predict the three-dimensional structure of the protein.

Protein structure prediction is also used in drug design studies. Accurate identification of protein-ligand interactions is important for studies in molecular biology and pharmacology. For this purpose, the structures of ligand-bound receptor complexes have been identified by X-ray crystallography and NMR. In addition to this, structure of the

binding energies of rate constants and amino acids that are important in binding were determined as a result of mutagenesis studies. Although these experiments contribute to the adequate identification of the protein-ligand complex, they are often effortful and difficult to perform routinely. Similar information about protein-ligand complexes can be more easily obtained using molecular modeling techniques such as molecular docking and molecular dynamics simulations, which can provide detailed dynamic information even in the absence of experimental data. In these simulations, predicted coordinate values can be used for the three-dimensional structure information of the protein as well as the experimental data. It is also possible to improve the predicted structures of proteins thanks to molecular dynamics simulations. Identification of ligand-protein interactions is critical in drug design and subsequent search for new therapeutic modalities.

2.4.1 Protein Secondary Structure Prediction

PSSP aims to assign a secondary structure class label to each amino acid in the protein sequence. To compute the accuracy of PSSP, the predicted class labels are compared to true labels, which can be obtained from experimentally solved 3D structure using the DSSP [50] algorithm. Protein secondary structures were originally represented by three states: strand (E), helix (H), and coil (C). Subsequently, Kabsch and Sander extended these to eight states: α-helix (H), 3_{10}- helix (G), π-helix (I), β-strand (E), isolated β-bridge (B), turn (T), bend (S), and others (C) [50]. Although eight state PSSP is more informative, three state PSSP is still used for 3D structure prediction because it is more accurate than eight state PSSP and can still give useful constraints about the formation of the 3D structure [51]. In this thesis, secondary structure elements are predicted in 3-states where the 8-state labels are transformed to 3-states using table 2.1.

Table 2.1 Transformation of 8-state labels of protein secondary structure to 3-states

8-state	3-state
H	H
G	
I	
E	E
B	
T	L
S	
-	

2.4.1.1 Literature Review for Protein Secondary Structure Prediction

Many studies have been carried out in the field of protein secondary structure prediction. To the best of our knowledge, PSSP first was done by Pauling et al. [52]. First generation models achieved around 60% overall accuracy (i.e. Q3 accuracy) and focused on predicting secondary structure class of each amino acid using only sequence information [53], [54]. The prediction accuracy increased to 77%-80% when multiple sequence alignments (MSA) are used as input features [55], [56]. Thereafter, 80.3-81.6% Q3 accuracy was achieved using PSSMs derived from several MSA algorithms such as PSI-BLAST and HHBLITS [10]. The Q3 accuracy further reached to 84%-85% using structural profiles as additional features, which were derived by aligning the target protein to template proteins with known structures [8], [35]. Aydin et al. used different structural profiles, which were generated using different template closeness ratio, to show effect of structural profiles on accuracy of PSSP [36]. According to these studies, it can be seen that the use of informative features increases the success rates of PSSP.

PSSP studies are not limited to obtaining better feature representations. Developing better prediction models is also important for PSSP. Jones outperformed all state-of-the-art methods with 78.3% Q3 accuracy using a two-state neural network as the machine learning method and PSI-BLAST PSSMs as the input features [57]. Support vector machines were first used for PSSP by Hua and Sun who obtained a result comparable to the literature [58]. Yang et al. proposed a nearest neighbor classifier and obtained 87.51% Q3 accuracy on T99 data set using both homologous and non-homologous features for PSSP [7]. Faraggi et al. improved the Q3 accuracy of PSSP with a multistep neural network that iteratively predicts the relative solvent accessibility, torsion angles and secondary structure [59]. Huang and Chen used four physicochemical features as additional attributes to PSI-BLAST PSSM profiles for a support vector machine classifier and reached 79.52% Q3 accuracy [60]. Magnan and Baldi showed that it is possible to predict secondary structure with high accuracy using one dimensional bidirectional recurrent neural networks if sequence-based structural similarity is also added [61]. Drozdetskiy et al. obtained 82.0% Q3 accuracy for PSSP and published a web server [62]. Wang et al. optimized *c* and *gamma* parameters of support vector machine using genetic and grid search algorithms and obtained 76.11% Q3 accuracy, which was 11.3% better than the model with default parameters [63]. Wang et al. proposed DeepCNF model, which integrates Conditional Random Fields and shallow neural networks, and obtained

84.4%, 84.7%, 85.4%, 82.3% and 84.5% Q3 accuracy on CASP10, CASP11, CullPDB, CB513 and CAMEO datasets respectively [64]. Heffernan et al. showed that using long short-term memory and bidirectional recurrent neural networks improves the Q3 accuracy of PSSP, because they can capture long range interactions [65]. Wang et al. improved the Q3 and Q8 accuracy of PSSP using recurrent based deep auto encoder architecture [51]. Aydin et al. compared extreme learning machines, k-nearest neighbor, support vector machines, random forest and artificial neural networks and applied information gain ratio for feature selection on CB513 dataset. They showed that the best accuracy was obtained by support vector machines and better Q3 accuracies were obtained with lower dimensional datasets [66]. Aydin et al. compared auto encoder based dimension reduction method with other dimension reduction and feature selection techniques and showed that similar Q3 accuracies can be obtained in lower dimensions with faster models [67]. Fang et al. obtained 85.98%, 83.59% and 80.59% Q3 accuracies on CASP10, CASP11 and CASP12 datasets, respectively using a deep learning model that is based on inception-inside-inception networks [26]. Torrisi et al. reviewed deep learning models and discussed the future studies for PSSP [68]. Torrisi et al. improved Q3 and Q8 accuracies of PSSP with a deep learning model based on bidirectional recurrent neural networks and convolutional neural networks that is trained both on single sequence and evolutionary profile-based inputs [69]. Aydin et al. showed that structural profile matrices increase the accuracy of PSSP [36]. Klausen et al. improved Q3 accuracy and optimized processing time for PSSP with a deep learning model based on convolutional and long short-term memory neural networks [70]. Hanson et al. reaches similar Q3 and Q8 accuracies with robust performance model based on the ResNet architecture using long-short-term memory layers [29]. Kumar et al. improved Q3 and Q8 accuracies for PSSP on four datasets using 42 hybrid features with a deep learning model based on convolutional neural network and bidirectional recurrent neural network [71]. Zhou et al. proposed an effective deep learning architecture for PSSP by combining several layers [72]. Xu et al. proposed a deep learning model based on multi task architecture using convolutional neural networks, improved transformers and bidirectional long-short-term memory layers and improved accuracies on eight datasets [27]. Kotowski et al. showed that using U-net architecture increases the accuracy of PSSP [73]. Uddin et al. obtained better accuracy with a model based on self-attention mechanism [56]. Guo et al. achieved 81.62% Q3 accuracy with multiple advanced deep learning architectures [74]. Jumper et al. outperformed state-of-the-art methods with a deep learning architecture that uses physical

and biological features of protein and take advantage of the multi-sequence alignments [75]. Görmez et al. improved Q3 accuracies of PSSP on five datasets with a novel method that uses convolutional and graph convolutional neural network layers [28]. Zhao and Liu achieved 85.08%, 84.21%, 84.68%, 82.36% and 82.91% Q3 accuracies on 25PDB, CB513, CASP10, CASP11 and CASP12 datasets, respectively with an optimized deep learning model based on convolutional and long short-term memory neural networks [76]. Yang et al. aimed to achieve similar performance with a lightweight embedding network [77]. Xu et al. upgraded their previous model and published an open source server for PSSP [78]. Gao et al. improved Q3 accuracy on six datasets with a model based on wavelet scattering convolutional and the long-short-term memory networks [79]. Yang et al. aimed to outperform all state-of-the-art models with a model based on lightweight convolutional network and label distribution aware margin loss [80]. Urban et al. outperformed their previous model with an upgraded version [81]. Görmez and Aydin aimed to increase Q3 accuracy of PSSP by upgrading their previous model using multi-task approach and bidirectional long-short-term memory networks [82].

2.4.2 Solvent Accessibility Prediction

Solvent accessibility prediction (SAP) aims to determine which amino acids are on the outer surface of the protein and which are in the inner region of the protein. This way, various constraints are obtained for three-dimensional structure prediction. In order to determine the accuracy of the SAP methods, the prediction result is compared with the correct accessibility information calculated using the DSSP [50] algorithm starting from the three-dimensional structure. As the accessible surface area can take different values for different amino acids, it is converted into relative solvent accessibility information as a result of a standardization process. For this, the accessible surface area of each amino acid calculated by the DSSP [50] program is divided by the maximum accessibility of that amino acid. Solvent accessibility can be predicted as a real value (i.e. as absolute accessibility or relative accessibility) or it can be classified to discrete labels, which are obtained by applying thresholds to real-valued accessibility scores. In this thesis, relative solvent accessibility prediction has been performed, since, it is more useful for three-dimensional structure prediction.

2.4.2.1 Literature Review for Solvent Accessibility Prediction

SAP is also among the problems that are studied frequently in the field of protein structure prediction. Thompson and Goldstein achieved a significant improvement for SAP with the Bayesian probabilistic method [83]. Li and Pan obtained better results than the literature with their proposed method trained on multiple sequence alignment data [84]. Naderi-Manesh et al. aimed to improve the accuracy of SAP using their model based on information theory [85]. Ahmad and Gromiha published a server that contains neural network based model for absolute solvent accessibility prediction [86]. Yuan et al. proposed a support vector machine based classifier for solvent accessibility prediction and aimed to compute effect of kernel functions and sliding window sizes on the performance of the model [87]. Ahmad and Gromiha reported that proposed neural network architecture obtained better performance than the other methods in the literature [88]. Ahmad et al. obtained 18-19.5 mean absolute error with a neural network developed for real valued SAP. In their model the correlation between the experimental and predicted values ranged from 0.47 to 0.50, and 23.7 mean absolute error can be obtained if there is no neighbor information [89]. Adamczak et al. reached 15.3 mean absolute error with their proposed predictor based on recurrent neural networks and they published this model as a web server [18]. Kim and Park proposed a method based on support vector machines and PSSM that is computed by PSI-BLAST to consider handling unbalanced data and long-range interactions [90]. Nguyen and Rajapakse developed a web server using support vector machines as the classifier and RS126 and Manesh datasets [91]. Sim et al. firstly applied fuzzy based k-nearest neighbor method for SAP and they slightly improved the accuracy [92]. Faraggi et al. increased mean absolute error by 2-4 with a neural network method based on guided-learning approach [19]. Joo et al. obtained 0.148 mean absolute error with a k-nearest neighbors method that has optimum hyper-parameters [17]. Mirabello and Pollastri showed that increasing training set size affects the performance of the model positively by making analysis on SAP and published the most successful model as a web server [34]. Deng et al. obtained a significant improvement on SAP using a deep learning model based on sparse autoencoder [93]. Zhang et al. aimed to capture long range interaction using stacked deep bidirectional recurrent neural network and achieved 8.8 and 8.2 mean absolute error on CB502 and Manesh215 datasets, respectively [30]. Aydin et al. compared autoencoder with other dimension reduction and feature selection techniques and showed that similar

performance can be achieved with lower dimensional datasets [67]. Petersen and Marcatili developed a model using convolutional and long short-term memory neural networks for SAP and optimized the processing time. They showed that, better accuracy can be obtained with the optimized model in less time [70]. Kaleel et al. captured the long-range interactions by combining bidirectional recurrent neural networks and convolutional layers and they outperformed all of the up to date models [31]. Hanson et al. designed a deep learning model with ResNet approaches using long short-term memory and they improved the accuracy for SAP [29]. Xu et al. had a significant improvement on SAP using their model that is based on multi-task architecture [27]. In another article, Xu et al. upgraded their model and they published this model as a web server [78]. Görmez and Aydin increased the mean absolute error for SAP using graph convolutional neural networks [82].

2.4.3 Torsion Angle Prediction

As mentioned before, there are three types of torsion angles: omega (ω), phi (ϕ) and psi (ψ). Because ω is generally close to 180^o, TAP typically aims to predict the values of ϕ and ψ angles for each amino acid of the target protein. These angles take continuous values, but form specific clusters for various secondary structural elements in the Ramachandran graph. Although, torsion angle class information may also be used for three-dimensional protein structure prediction, in this thesis, real values of torsion angles are predicted. Similar to PSSP, torsion angle prediction is studied as a supervised learning problem. True angle values are first assigned to amino acids of proteins selected to form a data set, where the angle values are computed starting from 3D structure information. In the next step, a subset of these proteins is selected as training set and the rest as test set. A prediction method is developed using training set and predictions are made for proteins in test set. TAP may be more effective in predicting the three-dimensional structure of proteins than PSSP [22].

2.4.3.1 Literature Review for Torsion Angle Prediction

TAP is a less studied problem compared to SAP and PSSP. Various methods for TAP have been developed in the literature. Among these, there are methods that predict real (continuous) angle values as well as approaches that predict angle classes. However, predicting the real angle values is more difficult and the number of methods developed for this is relatively less. Kuang et al. improved the torsion angle state prediction using

two separate models that were based on support vector machines and feed forward neural networks [21]. Keskin et al. discovered that many torsion angle pairs in the amino acid chain of a protein are strongly related with each other [94]. Wu and Zhang used PSSM, predicted secondary structure and solvent accessibility, which are used as input for a neural network based model and obtained 28 and 48 mean absolute error for phi and psi angles, respectively [95]. Xu et al. obtained 38 and 25 mean absolute error for psi and phi angles respectively with integrated neural networks [96]. Cheung et al. tried to find all possible values of phi and psi angles using Bayesian inference [97]. Heffernan et al. obtained 19 mean absolute error for phi and 30 mean absolute error for psi using deep neural network architecture [98]. In another study, Heffernen et al. achieved 5% mean absolute percentage error for phi and 10% mean absolute percentage error for psi using bidirectional recurrent neural networks [65]. Li et al. proposed 4 different deep architectures using deep neural network, recurrent neural network, restricted Boltzmann machine and recurrent restricted Boltzmann machine and achieved 20 mean absolute error for phi and 29 mean absolute error for psi [99]. Gao et al. decreased mean absolute error for torsion angles by 2 to 6 using deep neural network architecture [100]. Fang et al. used PSI-BLAST PSSM, 30 frequencies generated using HHBlits, and eight structural profiles of secondary structure as an input for deep residual inception network architecture to predict psi and phi angles [33]. Gao et al. obtained 18.08, 20, and 20.69 mean absolute error for phi and 26.68, 30.14 and 32.63 mean absolute error for psi on TS1267, CASP11 and CASP12 datasets, respectively using combination of k-means clustering, deep convolutional neural and residual networks [101]. Klausen et al. improved accuracy and optimized processing time for TAP with a deep learning model based on convolutional and long short-term memory neural networks [70]. Hanson et al. designed a deep learning model with ResNet approaches using long short-term memory and decreased the mean absolute error for TAP [29]. Mataeimoghadam et al. decreased mean absolute error by 6 to 8 for TAP using a simple deep learning architecture [49]. Xu et al. decreased the mean absolute error for TAP on eight datasets using a multi-task architecture based deep model [27]. In another study, Xu et al. proposed a model to refine the result of other TAP models [102]. Newton et al. outperformed all up to date studies for TAP by training separate deep learning models for each category of secondary structure [103].

2.4.4 Contact Map Prediction

The protein contact map, which is also called a relic proximity map, indicates amino acid pairs that are close together in three-dimensional space. The contact map can be represented by a matrix, in which the number of rows is equal to the number of columns, which is equal to the number of amino acids. If we call this matrix *M*, *M*[*i*,*j*] (the element in row *i* and column *j* of the matrix) takes the value 1 if amino acid *i* is in a certain proximity to amino acid *j*, otherwise it is 0. Figure 2.5 shows an example for contact map. The maximum distance required for two amino acids to be considered close is called the threshold distance, which can take different values. Contact maps can be used to predict the three-dimensional structure or the folding rates of proteins. They are preferred because they are not affected by the rotation and translation of the 3D structure. Starting from an accurate contact map, it is possible to predict the three-dimensional structure using various algorithms. A contact map can also be considered as a different representation of the three-dimensional structure of the protein. Therefore, the difficulty of estimating the contact map is approximately close to the difficulty of estimating the three-dimensional structure.

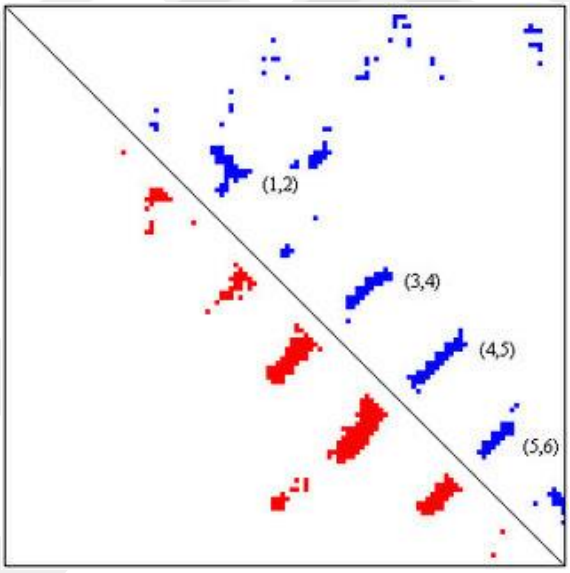


Figure 2.5 Contact map for the 1DZOA protein. The red dots show the correct map and the blue dots show the contact map estimated by the SVMcon method [104].

Although significant progress has been made in contact map prediction in the past years, this problem has not been largely solved yet. Many machine learning methods have been developed for contact map prediction. These include support vector machines [104], [105], neural networks [106]–[111], self-organizing maps [112], and rule-based classifiers [113]. Deep learning models were also used for protein contact map prediction

as well as traditional machine learning methods and the best accuracies were obtained with them [108], [110], [114]. In this thesis, contact map of proteins was predicted using SPOT-Contact [114] that is one of the most accurate models developed for this task. SPOT-Contact was downloaded from Sparks Lab [115]. The predicted contact maps are then used to generate amino acid interaction matrix for the GCN models.

Chapter 3

Materials and Methods

3.1 Benchmark Datasets

In this thesis, a total of 14 benchmark datasets were used for different purposes. The names of these datasets are CullPBD, EVAset, CASP10, CASP11, CASP12, CASP13, CASPFM, TEST2016, TEST2018, CAMEO93, CAMEO93_HARD, HARD68, validation and train. Among them, CullPDB, EVAset, CASP10 and CASP11 were only used for PSSP and the remaining were used for PSSP and TAP. Because only TEST2016, TEST2018, validation and train datasets have solvent accessibility label, SAP experiments were only performed using these datasets.

The CullPDB dataset, which have 9084 proteins, was downloaded from the protein sequence culling server [116], which includes lists of amino acid sequences from protein data bank (PDB). To obtain this dataset, the PC threshold is set to 20%, resolution threshold to 2.5, R-value and Rfree thresholds to 1.0. Secondary structure labels were assigned using DSSP [50] program. Torsion angles were assigned using a program called phipsi_linux, which was downloaded from the internet. Proteins which had different number of amino acids in the output and input files of torsion angle assignment program were eliminated. At the end of this process, 8552 proteins were obtained.

EVAset [117] is another dataset that was used for PSSP only. It originally contained 3074 proteins that were culled from PDB. Before using EVAset, proteins, which have smaller than 30 amino acid sequence, were eliminated. After this process, 2876 proteins, which had a total of 584 595 amino acids, were obtained.

The Critical Assessment of Protein Structure Prediction (CASP) [118] focuses on improving the performance of protein structure prediction problems and highlighting the current progress made in these problems. For this context, many experiments have been done and several datasets were published. In this thesis, CASP10 and CASP11 datasets were used to assess the performance of PSSP models and CASP12, CASP13 and CASPFM datasets were used to assess the performance of PSSP, SAP and TAP models. In addition to this, CASP12 had two versions where one of them was used for PSSP

models, and the other one was used for multi-task based models, which is the same as the dataset used for evaluating the performance of the OPUS-TASS [27] method.

Remaining datasets, which include TEST2016, TEST2018, CAMEO93, CAMEO93_HARD, HARD68, validation and train, are the same as the datasets used in the OPUS-TASS [27] paper. These datasets were downloaded from OPUS-TASS GitHub page [119]. Proteins, which had more than 700 amino acids, were eliminated from these datasets.

During the experiment phase, each dataset was used for a different purpose for different models. In this regard, CullPDB dataset was used to generate train, test and validation datasets for IGPRED and GraphUnet-SS. EVAset, CASP10, CASP11, CASP12 were used as test sets to compute the performance of the prediction models that were trained by CullPDB. For each of these test sets, a separate train set is derived from the CullPDB dataset. The Train dataset of the OPUS-TASS paper was used to generate train sets for IGPRED-MultiTask. The validation dataset of the OPUS-TASS paper was used as the validation set during training the multi-task learning models and to optimize the hyper-parameters of these models. CASP13, CASPFM, TEST2016, TEST2018, CAMEO93, CAMEO93_HARD, HARD68 and the other version of CASP12 were used as test sets to compute the performance of the multi-task learning models.

The presence of similar proteins in test and training sets will cause the prediction problem to be easier. For this reason, a separate train set for a test dataset or test dataset group was generated by performing pairwise BLAST alignments [120] with a stringent E-value cut-off of 0.05. Firstly, a non-redundant 22 proteins from CullPDB datasets were selected to generate CullPDB-test dataset, and the remaining 8530 proteins were used to generate CullPDB-train dataset. After that, pairwise BLAST alignments were applied between CullPDB-train and EVAset, CASP10, CASP11 and CASP12 separately to generate CullPDB-train-EVAset, CullPDB-train-CASP10, CullPDB-train-CASP11 and CullPDB-train-CASP12. In the final phase, pairwise BLAST alignments were applied between the train set of the OPUS-TASS paper and CASP13, CASPFM, TEST2016, TEST2018, CAMEO93, CAMEO93_HARD, HARD68 and the version of CASP12 used in OPUS-TASS paper to generate train-TEST2016, train-TEST2018, train-CAMEOs, train-CASPs and train-val. The number of proteins and the number of amino acids for each dataset are given in table 3.1.

Table 3.1 The number of proteins and the number of amino acids in benchmark datasets.

Benchmark Dataset	Number of Proteins	Number of Amino Acids
CullPDB-train	8530	1959390
CullPDB-test	22	5229
EVAset	2876	584595
CASP10	75	18231
CASP11	67	17179
CASP12_v1	39	11246
CullPDB-train-EVAset	6068	1230357
CullPDB-train-CASP10	7147	1471812
CullPDB-train-CASP11	7156	1474300
CullPDB-train-CASP12	7164	1475788
train	10042	2235849
validation	983	215803
TEST2016	1212	287733
TEST2018	250	50889
CAMEO93	93	22901
CAMEO93_HARD	15	4375
CASP12_v2	55	10283
CASP13	32	5354
CASPFM	56	8100
HARD68	45	6447
train-TEST2016	8850	1877909
train-TEST2018	9903	2195453
train-CAMEOs	1024	2235849
train-CASPs	9936	2208407
train-val	8999	1952387

3.2 Feature Extraction

As mentioned in the literature review, features are one of the most important elements used to improve the accuracy of protein structure prediction. Generating rich feature sets using several alignment algorithms, structural properties and other properties of amino acids can improve the performance of protein structure prediction [8], [10], [35], [36]. Because of this reason, position specific scoring matrices (PSSM) calculated using PSI-BLAST [38], profiles calculated using HHBlits [39], seven physico-chemical properties of amino acids, physico-chemical features from AAindex [40] and structural profiles were used to generate input features of the machine learning models.

3.2.1 PSI-BLAST PSSM Features

PSI-BLAST algorithm, which is fast, reliable and available online, allows an amino acid sequence to be compared to sequences in the protein database [38], [121]. PSSMs are commonly used to represent patterns (i.e., input features) in proteins. In this study, each target protein was aligned with the NR database using the PSI-BLAST algorithm. The number of iterations was set to three, E-value threshold to 10 and inclusion E-value threshold to 0.001. As a consequence, 20 scores were obtained for each amino acid (as the output of PSI-BLAST), which formed the PSSM with a dimension of 20 by N, where N is the number of amino acids. These values were normalized to the interval between 0 to 1 by applying a sigmoidal transformation as in Aydin et al [10].

3.2.2 HHMAKE Profile Features

Although PSI-BLAST is the most commonly used alignment algorithm for protein structure prediction, it's slow for iterative search due to the size of the NR database. Therefore, Remmert et al. proposed an alignment algorithm that has fast iterative sequence search [39]. In this thesis, HMM-profiles, which consist of scores for background probability, emission probability, and transition probability distributions, were computed using the HHBlits software. For this purpose, each protein was aligned to the Uniclust30 database and the number of iterations was set to two, which is the default setting for HHblits. In the first phase, 20 PSSMs scores were computed for each amino acid as log-odds ratios, which can be expressed as $\log_2(\frac{P_e(i,j)}{P_b(j)})$ where $P_e(i,j)$ is the emission probability for the j^{th} amino acid at the i^{th} match state with $1 \leq j \leq 20$, $1 \leq i \leq N$, N being the number of amino acids in the target, $P_b(j)$ is the background probability of emitting the j^{th} amino acid. Similar to PSI-BLAST PSSMs, these values are normalized to interval between 0 to 1 using a sigmoidal transformation [10]. As the second set of features, seven transition probability values of the HMM-profile are taken directly without applying any normalization. Finally, the three local diversity scores denoted as *Neff*, *Neff1*, *NeffD* parameters of the HMM-profile are taken and normalized by sigmoidal transformation. As a result, a PSSM with dimension 30 by N was obtained for each target. Special care was taken for the file format of the HMM-profiles. For instance, a value of a star character (i.e. '*') represents zero for the emission probability of match states (i.e. $P_e(i,j)$) and for transition probability scores. In the case of emission probabilities these values would be mapped first to minus infinity (when log_2 transform

is applied for computing log-odds score) and then to zero by the sigmoidal transformation. Therefore * values are directly taken as zeros in the final feature representation without computing the transformations explicitly. Furthermore, the values for *Neff*, *Neff1*, *NeffD* parameters were divided by 1000 to obtain the actual values since these are expressed in units of 0.001 according to the file format. The values for emission probabilities, background probabilities and transition probabilities in HMM-profile files were divided by -1000 to obtain the log_2 transform of the corresponding probability scores. These log_2-transformed scores obtained for background probabilities were later used directly to compute log-odds ratios as explained above and the log_2-transformed scores for transition probabilities were inverted to compute the transition probability scores.

3.2.3 Structural Profiles

A structural profile matrix (SPM) is a collection of probability distributions, in which each distribution shows the probability of a given amino acid to belong to one of the structure classes. The size of an SPM is C by N, where N is the number of amino acids in the target protein and C is the number of structure class labels used for constructing the SPM (e.g. C=3 for secondary structure and C=2 for solvent accessibility). In this thesis, two types of SPMs are used: a 3 by N SPM derived using secondary structure labels and a 2 by N SPM derived using 2-state solvent accessibility labels. Aydin et al. showed that using SPMs increased the accuracy of protein structure prediction [36], [37]. In this thesis, SPMs were computed using the HHblits method. In the first step, the target protein was aligned against the UniClust30 database. In the second step, HMM-profile of the target was aligned with the HMM-profiles of the proteins in the PDB99 database, which is an in-house (i.e. customized) database derived by Aydin lab using the scripts and commands available in the user guide of HHblits starting from the PDB99 dataset obtained from the PISCES server [36]. Finally, the SPM was computed as the weighted average of label frequencies resulting from the alignment between the target and templates in PDB99. Only distant templates were used to construct SPMs. For this purpose, templates having a percentage of sequence identity score above 20% were eliminated to minimize the impact of template similarity on accuracy rate. Details on computing the structural profiles can be found in the paper by Aydin et al. [36]. Once the SPMs are computed, each amino acid is represented by 3 scores if the SPM is derived using secondary structure labels and 2 scores if the SPM is derived using solvent accessibility labels. These scores are taken as the structural profile features. In our IGPRED and GraphUnet-SS models

(i.e. our first and third models), which were developed for PSSP, SPMs derived using secondary structure labels (i.e. those having dimension 3 by N) were used only. In our IGPRED-MultiTask model, which was developed for PSSP, SAP, and TAP, the SPMs derived using secondary structure and solvent accessibility labels (i.e. those having dimensions 3 by N and 2 by N) were used to obtain structural profile features.

3.2.4 Physico-chemical properties

Amino acids in a protein may have different types of physico-chemical properties. Fang et al. showed that summarizing these properties may improve the accuracy of prediction methods [26]. In this thesis, seven physico-chemical properties, including volume of side chain, polarity, polarizability, hydrophilicity, hydrophobicity, net charge index of side chain and solvent accessible surface area, are used as the first set of physico-chemical input features for each amino acid.

In addition to using the standard set of 7 physico-chemical properties, 35 features from the AAindex database (https://www.genome.jp/aaindex/) are also added forming the second set of 42 physico-chemical features. AAindex is a database of amino acid indices proposed by Kawashima et al. [40]. It contains numerical values for more than 500 indices (representing various properties of amino acids), amino acid substitution matrices and amino acid contact potential matrices. In this thesis, a set of 35 indices listed in table 3.2 are selected and added to the feature set. However, AAindex values were only used for one of the proposed models. It is seen from the experimental results that, AAindex values did not improve the performance of the model considerably. Therefore, for the other studies they have not been used.

Table 3.2 Selected amino acid indices from AAindex

AAindex accession number	Description of index
BHAR880101	Average flexibility indices [122]
BIGC670101	Residue volume [123]
PONJ960101	Average volumes of residues [124]
CHAM820102	Free energy of solution in water, kcal/mole [125]
CIDH920105	Normalized average hydrophobicity scales [126]
BASU050101	Interactivity scale obtained from the contact matrix [127]
BASU050102	Interactivity scale obtained by maximizing the mean of correlation coefficient over single-domain globular proteins [127]
ZHOH040101	The stability scale from the knowledge-based atom-atom potential [128]

ZHOH040102	The relative stability scale extracted from mutation experiments [128]
ZHOH040103	Buriability [128]
WOLR790101	Hydrophobicity index [129]
KIDA850101	Hydrophobicity-related index [130]
FASG890101	Hydrophobicity index [131]
KRIW710101	Side chain interaction parameter [132]
SIMZ760101	Transfer free energy [133]
ROBB790101	Hydration free energy [134]
RADA880108	Mean polarity [135]
ROSM880102	Side chain hydropathy, corrected for solvation [136]
VELV850101	Electronion interaction potential [137]
WARP780101	Average interactions per side chain atom [138]
WOLR810101	Hydration potential [139]
HOPA770101	Hydration number [140]
ZIMJ680101	Hydrophobicity [141]
ZIMJ680102	Bulkiness [141]
GRAR740102	Polarity [142]
ZIMJ680104	Isoelectric point [141]
ZIMJ680105	RF rank [141]
TAKK010101	Side-chain contribution to protein stability (kJ/mol) [143]
MEIH800103	Average side chain orientation angle [144]
MCMT640101	Refractivity [145]
HUTJ700102	Absolute entropy [146]
HUTJ700103	Entropy of formation [146]
FAUJ880103	Normalized van der Waals volume [147]
FASG760103	Optical rotation [148]
FASG760101	Molecular weight [148]

3.3 Feature Generation

Deriving a representative feature set is one of the most important steps for accurate prediction of structural elements of a protein. Based on the studies in the literature, combining different types of features has been shown to improve the accuracy of prediction. In this thesis, a rich feature set is derived for each amino acid, which contains 20 scores from PSIBLAST alignments, 30 scores from HHblits alignments, 7 scores reflecting physico-chemical properties, 35 AAindex values (only used for IGPRED, which is our first proposed model), S scores from structural profiles (S=3 for IGPRED

and GraphUnet-SS models and S=5 for IGPRED-MultiTask), and a NoSeq label as shown in figure 3.1.

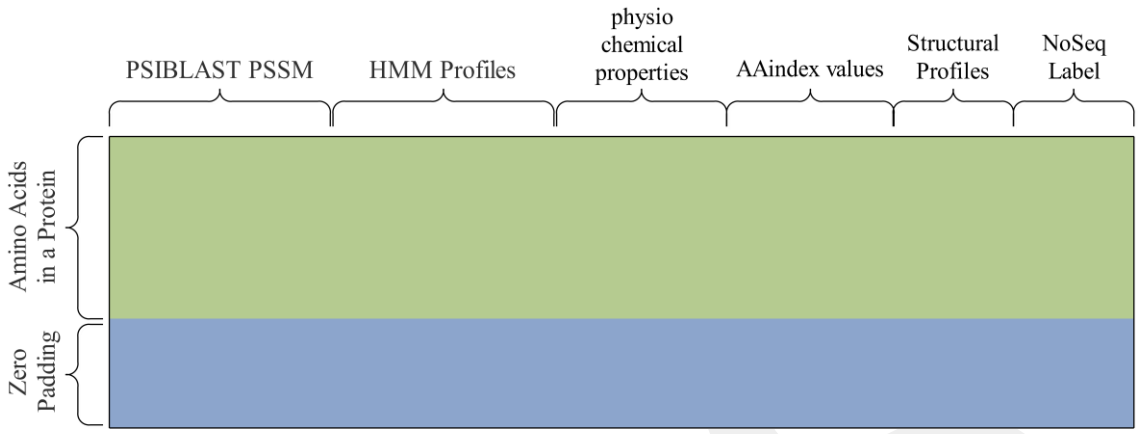


Figure 3.1 Feature matrix representation for a protein.

This type of feature set was originally proposed by Fang et al. [26]. In addition to features, which were explained in feature extraction section, a NoSeq label feature is included to denote whether a given row of feature matrix contains feature data (a NoSeq label of zero) or not (i.e. it contains zeros from zero-padding) (a NoSeq label of one). This feature is included because our deep learning models are designed to take a fixed sized input in time dimension (i.e. they expect the number of amino acids to be the same). If the number of amino acids in a target protein is less than the sequence length parameter (which is set to 700), zero-padding is applied for the remaining time steps and the NoSeq label will be set to 1 for the zero-padded section. Otherwise, it will be equal to 0. For example, if the length of the target protein is 500, then the first 500 rows of the data matrix contain features computed as explained above (i.e. including PSI-BLAST PSSM, HHMAKE PSSM, structural profiles, AAindex and physico-chemical properties) and the NoSeq label feature is set to zero. The last 200 rows have zeros as the values of all features and the NoSeq label feature is set to one. If the number of amino acids in a protein is more than 700, the amino acid sequence will be divided into multiple parts where each part contains 700 amino acids (with the last part completed into 700 amino acids by zero-padding if necessary). Then each part is treated as a separate protein. Figure 3.1 shows the feature data matrix for a protein with less than 700 amino acids (with zero padding applied). In this matrix, columns represent features and rows represent amino acids. Each protein corresponds to a data sample in each mini-batch of neural network model training.

3.4 Neural Network Layers

In this thesis, several deep learning architectures were proposed for SAP, TAP and PSSP. Deep learning models are types of neural networks that combine some special layers in different number and at different depth. Our proposed models were generated using fully connected layers (FCL), convolutional neural networks (CNN), graph convolutional networks (GCN) and bi-directional long short-term memories (biLSTM).

3.4.1 Fully Connected Layers

Artificial neural networks are formed by the combination of neurons, each of which contains a nonlinear function. Each neuron used as an input in a FCL is connected to all subsequent neurons, hence the name fully connected layer. In the FCL, neurons first perform a linear transformation to the input data, thanks to their specific weights. Result for of the neuron in the subsequent layer is dot product of input values and weight. After that a bias value is added to this result. Finally, a non-linear transformation is made with an activation function. Examples of these functions are Relu, tanh and sigmoid. The mentioned stages are shown in equation 3.1 for each neuron in a given layer (excluding the first layer).

$$\tanh(bias + \sum_{n=1}^{N} i_n w_n \tag{3.1}$$

In this equation, $bias$ represents the constant applied to each neuron, N represents the number of neurons input to the current neuron, n is the index of the input neurons, i_n represents the output of the n^{th} neuron in the previous layer that is input to current neuron, and w_n represents the corresponding weight for each neuron pair (i.e. the n^{th} neuron in the previous layer and the current neuron). Thanks to FCLs, many problems such as classification, feature extraction and clustering can be solved. In fully connected models, deep networks can be created by increasing the number of layers, thus better results can be obtained in the prediction problem. However, this fully connected structure does not yield sufficient results for some types of problems. Especially in models where artificial neural network models are used for feature extraction, other layers are frequently used as an alternative to fully connected models. For example, layers such as GCN, LSTM, CNN and transformers are used as an alternative to the fully connected layer for problems such as image processing, time series and text processing [149]–[153]. An example architecture for the fully connected layer is presented in figure 3.2.

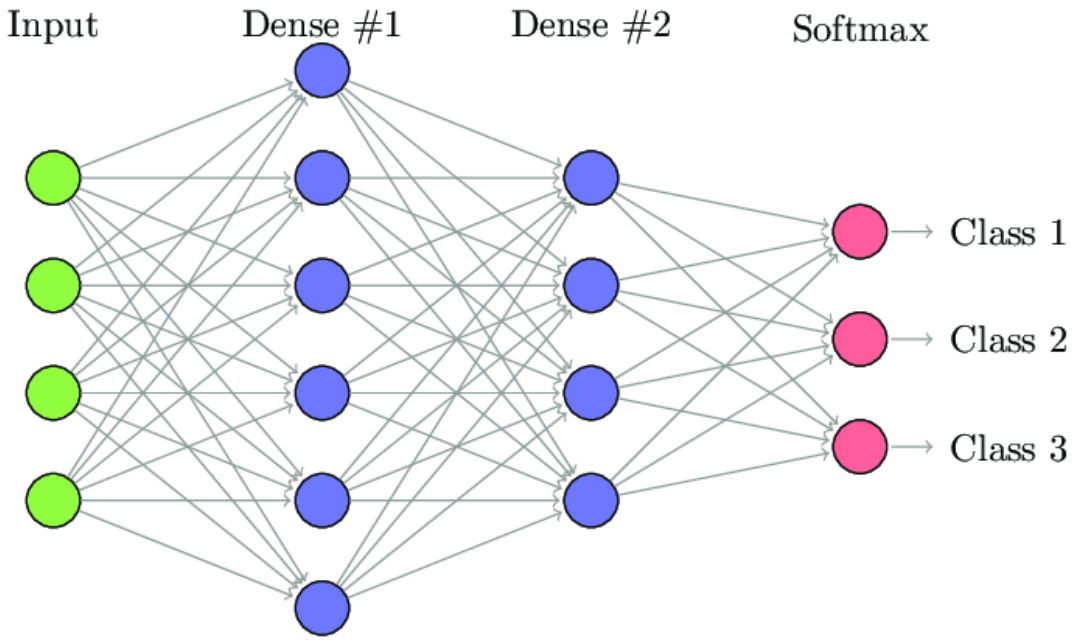


Figure 3.2 Example architecture for deep fully connected layer [154].

3.4.2 Convolutional Neural Network

CNN, which is designed with a shared weight structure, is an artificial neural network layer and is generally used for image processing [155]. In a CNN layer, unlike the FCL, an input is not connected to all subsequent neurons. In this structure, a convolution operation is performed using a given matrix or vector and several number of kernels of certain sizes. As a result of this process, a different number of new matrices or vectors are obtained for the next layer. This process can also be thought of as obtaining new pictures by applying different filters on the picture. In this example, each kernel represents a filter. For a CNN layer, similar to a FCL, a nonlinear function is applied to each value obtained by the convolution operation. In this way, the CNN layer becomes able to produce solutions for nonlinear problems. In a model created using a CNN, it is possible to use operations such as pooling, padding, batch normalization and dropout as well as convolution and activation function operations. These operations, the use of which varies according to the nature of the problem, significantly affect the performance of the model. An example for a CNN model is shown in figure 3.3. Unlike the FCL, the fact that it uses shared weights and contains different numbers of kernels allows both computations to be made in parallel and to reveal the relationships between the inputs at different positions. While the order of the input values in a FCL does not matter, this order is important for the CNN layer. With this structure, it achieves better performance than the FCL in

problems where input ordering is important such as image processing and time series. It is considered that, models that use CNN will positively affect protein structure prediction problems because amino acid sequence is important in protein structure prediction and there is a relationship between nearby amino acids.

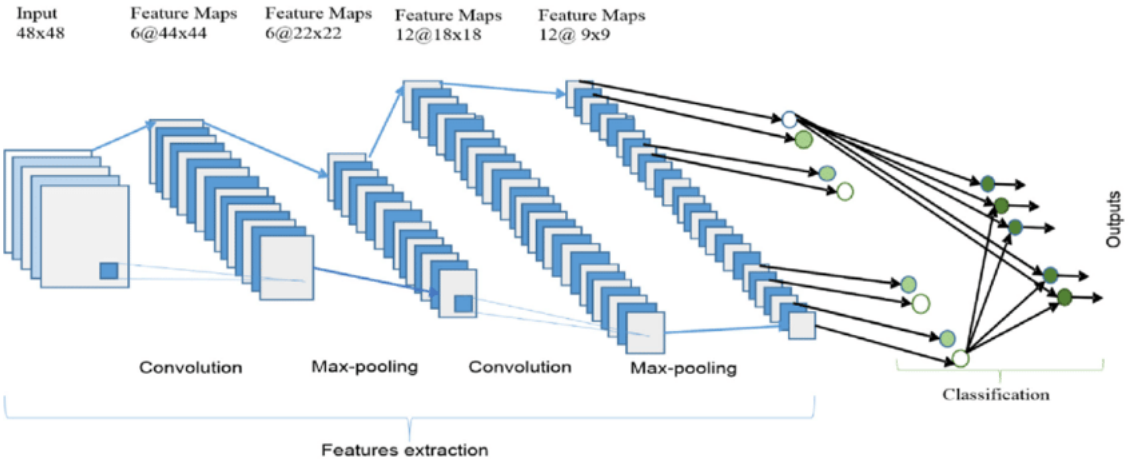


Figure 3.3 Example architecture for model that developed using convolutional neural network [156].

3.4.3 Bidirectional Long Short-Term Memory

LSTM, which is a special version of recurrent networks, is a neural network layer that can detect dependencies between data in a time series. LSTMs achieved significant improvement in many problems such machine translation, speech recognition, text completion and other time series problems due to its structure [157]–[160]. Unlike other recurrent networks, LSTMs overcome the problems of vanishing and exploding gradients thanks to persistent memory. There are three main structures in an LSTM cell: input gate, output gate, and forget gate. As in a recurrent network, there are hidden states in the LSTM structure, where $H(t-1)$ represents the previous state and $H(t)$ represents the current state. In addition to this, there are cell states represented by $C(t-1)$ and $C(t)$ in the LSTM structure. In this structure, the hidden state refers to short-term memory, while the cell state refers to long-term memory.

In the forget gate, it is decided which information is no longer needed in the cell state and which information should be deleted. In this context, the inputs $x(t)$ and $h(t-1)$ are multiplied by a weight matrix and a bias value is added to the output value. After this process, as in a traditional neural network, the output value undergoes a non-linear transformation with the help of the activation function. If the value is 0, the information is forgotten, otherwise the information continues to be stored. On the other hand, the input

gate contains additional information that will be useful to cell state and is regularized with the help of the sigmoid function. Finally, the part where the obtained information is presented represents the output door. By default, LSTM structures can capture the relationship in one-way serial data. However, as with protein structure prediction, some data may be correlated in both directions (forward and backward) of the series. As a solution to this problem, forward and backward LSTM structures are created separately and then connected to the same output layer. These structures, which are also used in the proposed models of this thesis, are called bidirectional LSTM. An example architecture of a bidirectional LSTM is shown in figure 3.4 [161].

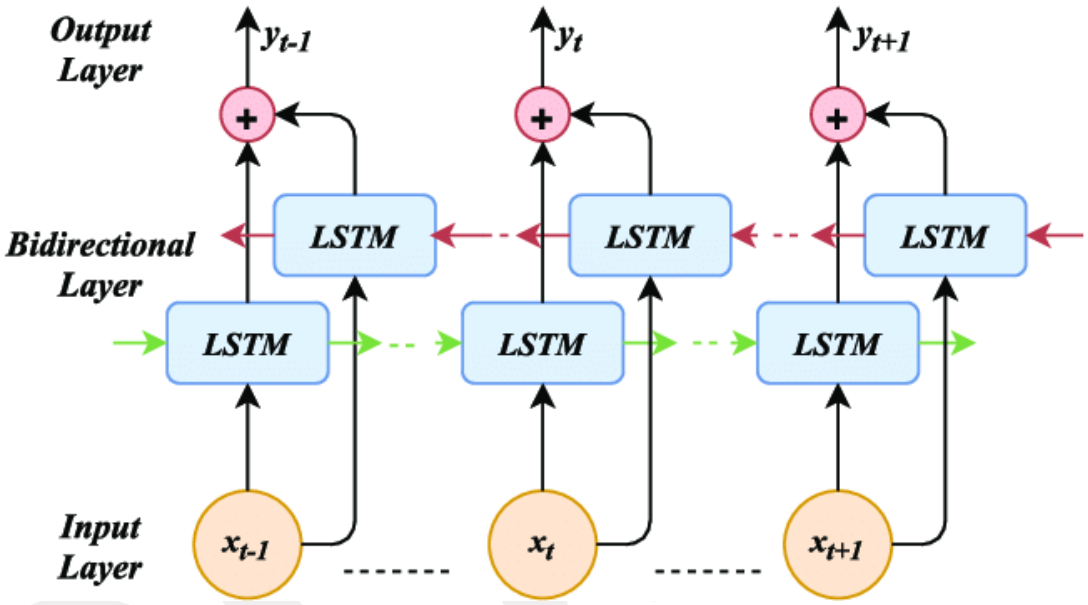


Figure 3.4 Example architecture for bidirectional long short-term memories [162].

3.4.4 Graph Convolutional Neural Networks

A graph can be defined as a data structure in which the connections between the data are indicated. In a standard graph, there are objects called nodes and structures called vertices, which represent the connections between the objects. Graphs, which can have many different types can be briefly considered as having 2 types in 2 different categories as weighted and unweighted, directed or undirected. Graphs can be used for many situations such as maps, social networks, computer networks, energy connections, protein networks and epidemics [163]. Recently, when graph information is used as input to deep learning models, it has improved the success rates of many problems. Because traditional deep neural network layers are not capable of processing graph type data, GCN have been proposed as a solution to this problem. A GCN contains convolution operation similar to

standard convolutional networks. However, in convolutional networks, the number of connections in the nodes is fixed and in order with the node, while the number of connections in GCN can vary and is not ordered with the node. An example for GCN is shown in figure 3.5. A GCN model needs an input matrix, in which the graph is represented. The GCN layer used in this thesis takes two inputs. The first of these is the feature vector, which is also used in standard machine learning models. The other input is the graph matrix that represents the interactions between amino acids in a protein sequence. Due to the nature of the problem predicted in this thesis, it is assumed that amino acids affect each other while the protein structure is being formed, but there is no relationship between the amino acid chains. In this context, separate and discrete graphs were created for each protein sequence.

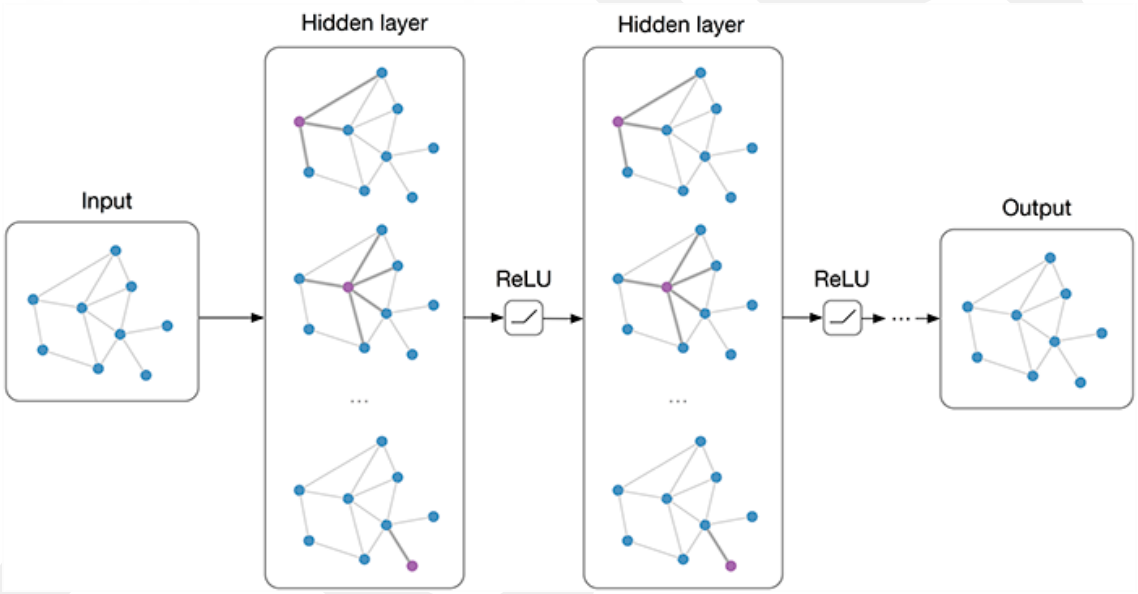


Figure 3.5 Example architecture for graph convolutional networks [164].

3.5 Graph Generation

Proposed models in this thesis contain a number of mGCN modules, which need a graph as an extra input. In this thesis, two different graph generation techniques were used: sliding window and graph generated with contact map prediction. The purpose of these graphs is to provide amino acid interaction information to models as an extra input. Graph, which was generated using sliding window method, was used for the first two models (IGPRED and IGPRED-MultiTask) and the other one was used for the third model (GraphUnet-SS).

In the sliding window method, a two-sided symmetric window was used to generate graph. For this context, it is assumed that amino acids that are close the each other in protein sequence have interactions. Therefore, a graph with 700 nodes is generated for each protein where each node represents an amino acid and edges represent interactions between amino acids, which can be summarized by an adjacency matrix of size 700 by 700. This graph is unweighted for which the adjacency matrix contains ones or zeros only. If there is an edge between nodes *m* and *n*, the value at row *m* and column *n* of the adjacency matrix is one, which represents an interaction between amino acids at positions *m* and *n*. This graph is also undirected with a symmetric adjacency matrix. This means that if there is an edge between nodes *m* and *n*, there will also be an edge between nodes *n* and *m*. Similar to feature extraction step, if the number of amino acids of the target protein was less than 700, zero padding was applied and a 700 by 700 adjacency matrix was generated. If the number of amino acids was more than 700, it was divided into non-overlapping amino acid segments of size 700, and a separate graph was generated for each segment considering pairwise interactions within the segment. For this second possibility, if the last segment contained less than 700 amino acids, zero padding was applied to complete its adjacency matrix to 700 by 700. As a result of this procedure, each amino acid segment was treated as a separate protein. A total of N disconnected graphs each with an adjacency matrix of size 700 by 700 is generated where N is the number of proteins in the dataset. This type of graph representation enables to capture short-range as well as long-range interactions by defining connections between interacting amino acids. In the sliding window method, short-range interactions are modeled only by selecting a two-sided symmetric window around each amino acid to define the interaction graph of proteins. Based on this constraint, a hyper-parameter called number of connections is introduced, which is equal to the number of interactions an amino acid makes on one side of the window. This value should be smaller than half of the number of amino acids in a protein. For example, if the number of connections is 10, then there will be an edge between a given amino acid and 10 amino acids that come before as well as 10 amino acids that come after according to the sequence representation of the protein. By the nature of the proteins, amino acids that are far away from each other in sequence may have interaction in 3-D space. The reverse of this idea is also true. Amino acids that are close to each other in sequence may not have interaction in 3-D space. Thus, we thought that the sliding window method used to generate graphs for the first two models can be improved in a smarter way. For this purpose, protein contact map prediction

(PCMP) was used to generate graph in the third model. PCMP indicates whether amino acid pairs are close or not in 3-D space. In this study, one of the best PCMP methods called SPOT-Contact was used to generate graphs. For this purpose, contact values of amino acid pairs for each protein were computed using the stand-alone version of SPOT-Contact, which is downloaded from Sparks Lab's web site [115]. This method predicts a contact value, which is a floating-point number between 0 to 1, for each amino acid of the target protein. Similar to the sliding window method, a graph with 700 nodes is generated for each protein. Initially an adjacency matrix of size 700 by 700 was generated and all values were set to 0. Then if the predicted contact value for two amino acids is greater than 0.5, the corresponding cell value of the adjacency matrix is mapped to 1, which means that the corresponding amino acid pair interacts, otherwise, the value of the cell stayed as 0 meaning that the amino acid pair does not interact. Using this pairwise interaction information, a graph was generated for each protein in the datasets (where a protein refers to an amino acid segment of size 700). At the end of this process, similar to sliding window approach, a total of N disconnected, unweighted and symmetric graphs were generated. Figure 3.6 summarizes the steps of the proposed classification models where AAindex values were only used for the first proposed model.

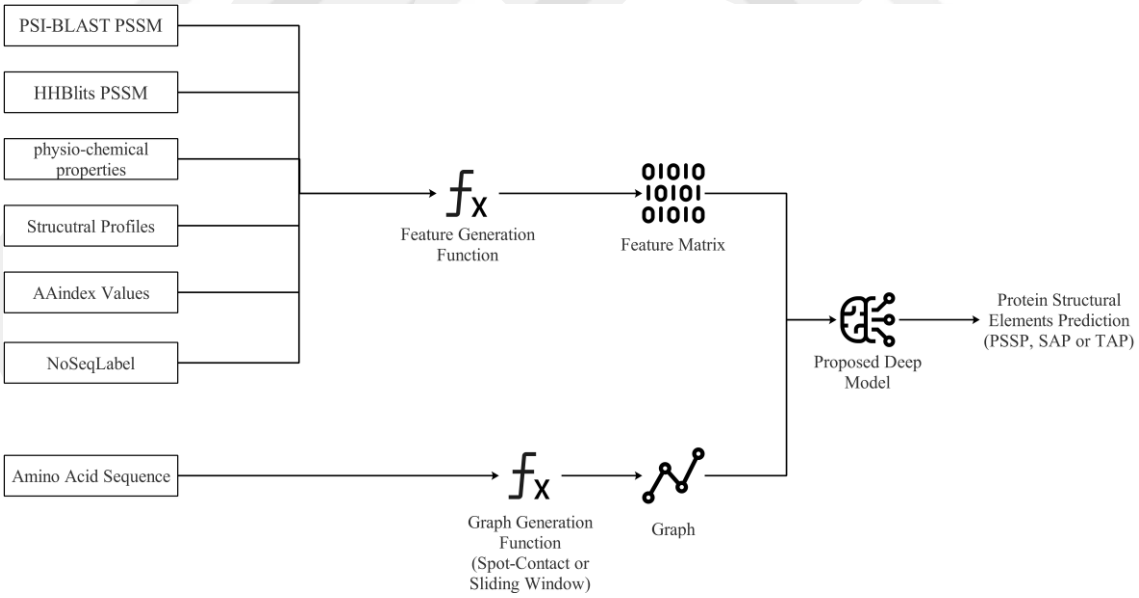


Figure 3.6 Summary of Classification Models

3.6 Proposed Models

In this thesis, a total of three deep learning models, which are called IGPRED, IGPRED-MultiTask and GraphUnet-SS, were proposed for PSSP, SAP or TAP.

3.6.1 IGPRED

The first model proposed in this thesis is a novel deep learning model called IGPRED, which consists of several CNN and GCN modules concatenated in different ways and followed by FCLs. Figure 3.7 shows the architecture of this model.

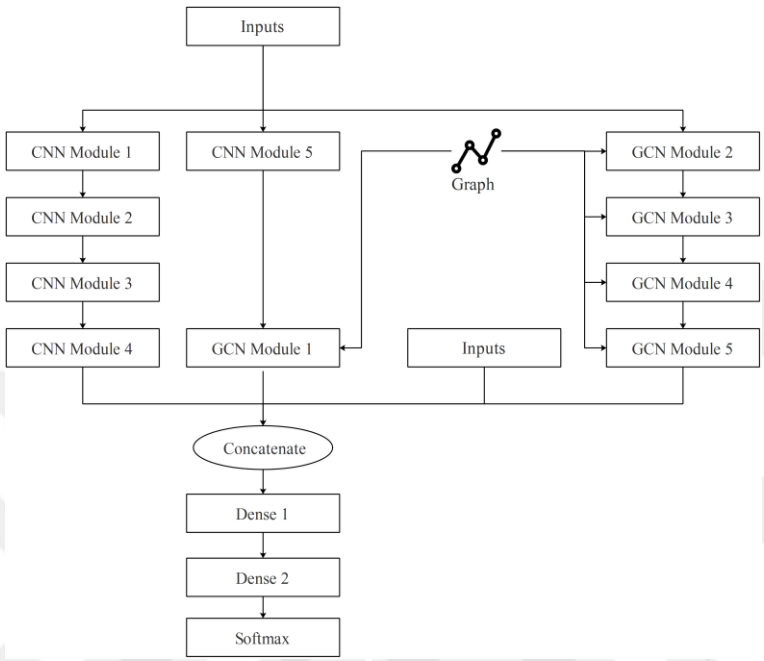


Figure 3.7 Architecture of IGPRED

Each CNN module is generated through six different convolutional layers with kernel sizes (1,M), (3,M), (5,M), (9,M), (11,M) and (15,M) that are connected in parallel as in inception module. Here, M represents the number of features because 1-D convolutional layers are used. These layers are connected using the inception architecture, which obtained significant improvements in many areas such as computer vision and bioinformatics [165]–[167]. Within each CNN module, the number of filters of the convolutional layers are identical and is a hyper-parameter that can be set for each module. As a result, different CNN modules can contain different number of filters. Each convolution layer consists of four operations in sequential order: convolution, batch normalization, activation, and dropout. Figure 3.8 shows the details of each CNN module. A GCN module is generated using a multi-graph convolutional layer (mGCN) [168], which consists of two operations in sequential order: a multi-graph convolution and dropout. An mGCN layer needs a graph as an extra input for each protein. As mentioned in Section 3.5 (i.e. Graph Generation section), the sliding window approach was used to generate graphs for this model and the number of connections was optimized. A softmax layer with *sparse_categorical_crossentropy* loss function is used to estimate the 3-state

secondary structure information. Adam is used as the optimization algorithm for estimating the weight coefficients of neural networks, where *beta_1* parameter is set to 0.95 and *beta_2* parameter to 0.99.

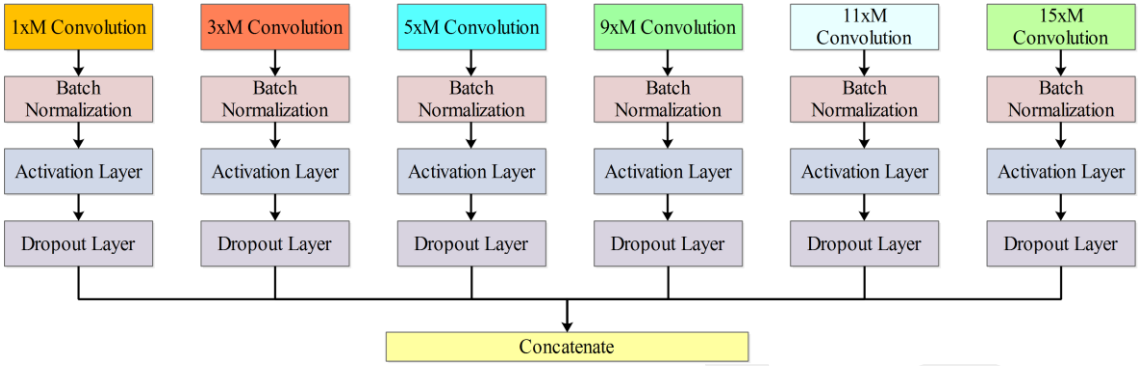


Figure 3.8 The layers inside each CNN module

3.6.2 IGPRED-MultiTask

The second proposed model of this thesis, which is called IGPRED-MultiTask, is a novel multi-task architecture based on CNN, GCN, biLSTM, and FCL layers. At the end of the model there are 3 output layers: a fully connected layer with Softmax activation function to predict secondary structure, a fully connected linear layer with one node to predict solvent accessibility and a fully connected linear layer with two nodes to predict phi and psi angles. Figure 3.9 summarizes the architecture of this model.

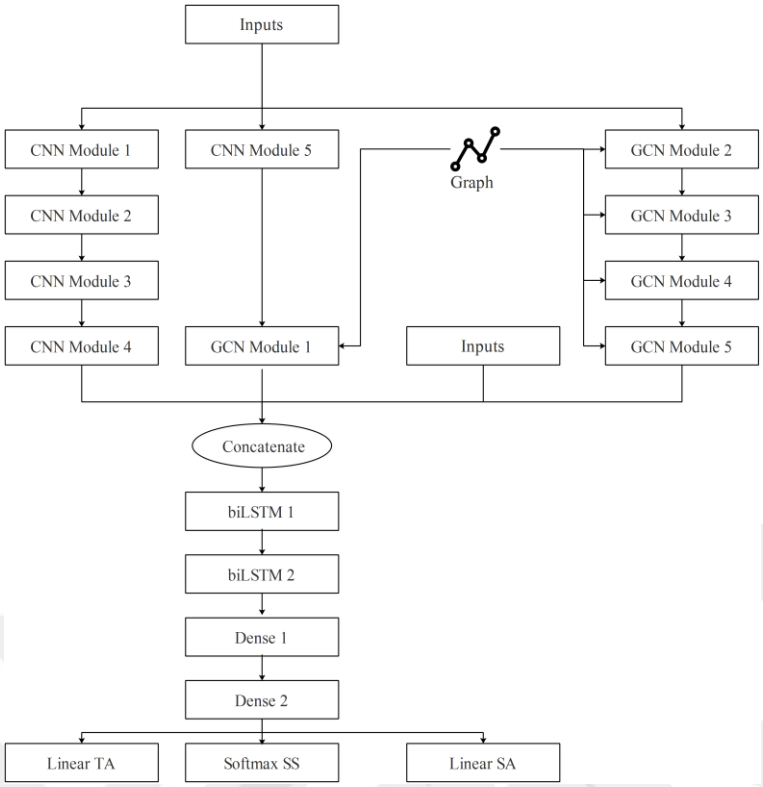


Figure 3.9 Architecture of IGPRED-MultiTask

Each CNN module consists of 5 different 1-D convolutional layers fed in parallel with kernel sizes (1,M), (3,M), (5,M), (9,M), and (15,M) where M represents number of features derived for each amino acid. Note that this is a form of an inception network similar to the IGPRED model. Except for the kernel sizes, all convolutional layers in the same module are identical. Four operations are applied to each layer in sequential order: convolution operation, batch normalization layer, activation layer with ReLu function and dropout. The outputs obtained from each dropout operation are concatenated to form the output of a CNN module, the architecture of which is depicted in Figure 3.8.

A GCN module of IGPRED-MultiTask consists of two inputs: feature data matrix and a graph representing interactions between amino acid pairs. In each GCN module, a multigraph convolutional layer (mGCN) is used as the model architecture [168], followed by batch normalization and a dropout layer. Similar to IGPRED, the sliding window approach was used to generate graph inputs for this model also and the number of connections was optimized.

In addition to CNN and GCN layers, IGPRED-MultiTask also contains BiLSTM layers. This is due to the fact that each amino acid interacts with its local neighbors that come before and after this amino acid (i.e. the interactions are two-sided). As in the CNN module, in each BiLSTM module, the three operations, which include batch

normalization, activation layer with ReLu function and dropout were followed after BiLSTM layers in sequential order. In each layer, the return_sequences parameter was set to true, thus long-range interaction information can be captured and transferred to the next layers and an output prediction can be obtained for each time-step (i.e. amino acids).

As can be seen in the model architecture, three parallel blocks are concatenated at the end of the first part of the proposed model. The first block only consists of CNN modules, the third block only consists of GCN modules and the second block consist of both CNN and GCN modules. By this way, amino acid features are embedded using only CNN modules, only GCN modules and a combination of them. It can be anticipated that faulty prediction of a block may be fixed by the others.

After the BiLSTM modules, two dense layers and three output layers follow implementing a multi-task architecture. There is an output layer for each of the prediction tasks including the prediction of torsion angle, secondary structure and solvent accessibility. However, for the datasets, which do not have solvent accessibility information, the corresponding output layer is removed. A softmax layer with *sparse_categorical_crossentropy* loss function is used to estimate 3-state secondary structure information. Linear layers with *mean_absolute_error loss* function are used to estimate real-valued torsion angle and solvent accessibility information. Adam is used as the optimization algorithm for estimating the weight coefficients of neural networks, where *beta_1* parameter is set to 0.95 and *beta_2* parameter to 0.99.

Several models have been generated that contain certain module blocks while excluding others to analyze the effect of each module. The first model IGPRED-SS is the same as the original model but excludes the output layers for torsion angle and solvent accessibility. Therefore, IGPRED-SS only predicts secondary structure at the output and does not perform multi-task learning. The second model IGPRED-TA is the same as the original model but excludes the output layers for secondary structure and solvent accessibility. Similar to IGPRED-SS, IGPRED-TA only predicts torsion angles at the output and does not perform multi-task learning. The remaining models are generated using various numbers of CNN, GCN and BiLSTM modules. For this purpose, a total of 12 models, which include IGPRED-CNN-free, IGPRED-CNN-1, IGPRED-CNN-2, IGPRED-CNN-3, IGPRED-CNN-4, IGPRED-GCN-free, IGPRED-GCN-1, IGPRED-GCN-2, IGPRED-GCN-3, IGPRED-GCN-4, IGPRED-BiLSTM-free and IGPRED-BiLSTM-1 are generated. IGPRED-CNN-free is the version of IGPRED-MultiTask that does not include any CNN Module blocks. IGPRED-CNN-1 is the version of IGPRED-

MultiTask that only includes CNN Module 1 as the convolutional module block. IGPRED-CNN-2 is the version of IGPRED-MultiTask that only includes CNN Modules 1 and 2 as the convolutional module blocks. IGPRED-CNN-3 is the version of IGPRED-MultiTask that only includes CNN Modules 1-3 as the convolutional module blocks. IGPRED-CNN-4 is the version of IGPRED-MultiTask that only includes CNN Modules 1-4 as the convolutional module blocks. The remaining versions are derived similarly. Detailed architecture of these models can be found at supplementary material of IGPRED-MultiTask paper [82].

3.6.3 GraphUnet-SS

The third proposed model of this thesis is a deep learning model called GraphUnet-SS that consists of several CNN, biLSTM and GCN layers concatenated based on U-net architecture and followed by fully connected layers. Firstly, modules for CNN, biLSTM and GCN layers, which are shown in figure 3.10, were developed.

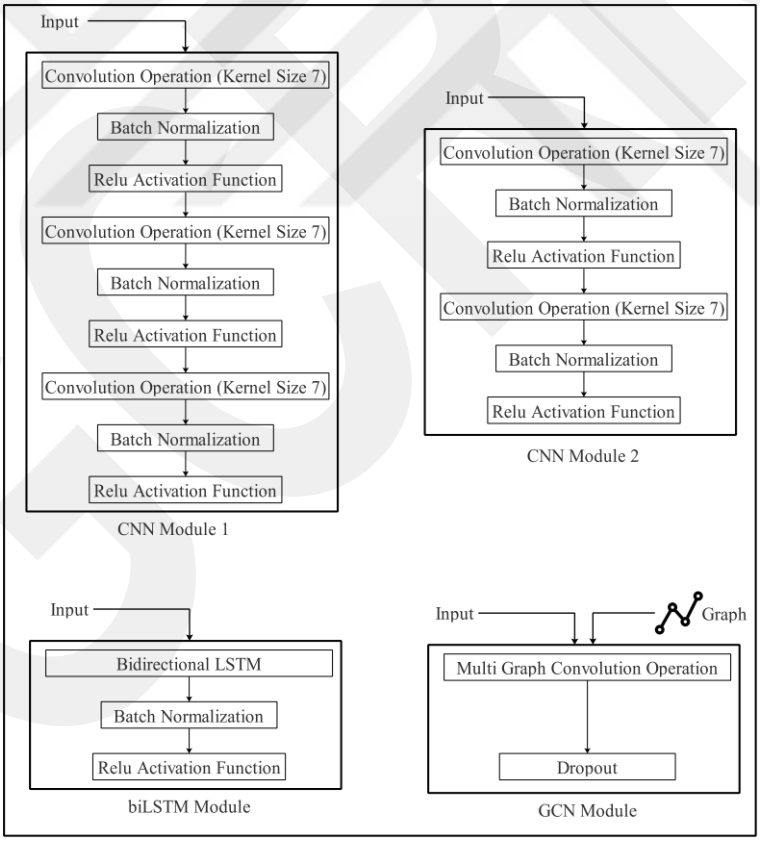


Figure 3.10 Architecture of modules that were used in GraphUnet-SS.

"CNN Module 1" contains 3 blocks and "CNN module 2" 2 blocks that are connected serially. Each block in CNN modules consists of 3 operations in sequential order including 1-D convolution operation with kernel size (7, M) where M represents the

number of features, batch normalization and Relu activation function. Similar to CNN modules, BiLSTM module also consists of 3 operations in sequential order: bi-directional long short-term memory operation, batch normalization and Relu activation function. Python Keras library [169] was used to implement biLSTM and CNN modules. Unlike the CNN and biLSTM modules, the GCN module takes two inputs where one of them is a feature matrix and the other one is a graph, which is in the form of an adjacency matrix that represents the pairwise interactions between amino acids. This module consists of 2 operations connected in sequential order: Multi Graph Convolution Operation (mGCN) [168] and dropout. Connecting these four modules following the U-net architecture and the fully connected dense layers a deep learning model was developed, the architecture of which is shown in figure 3.11.

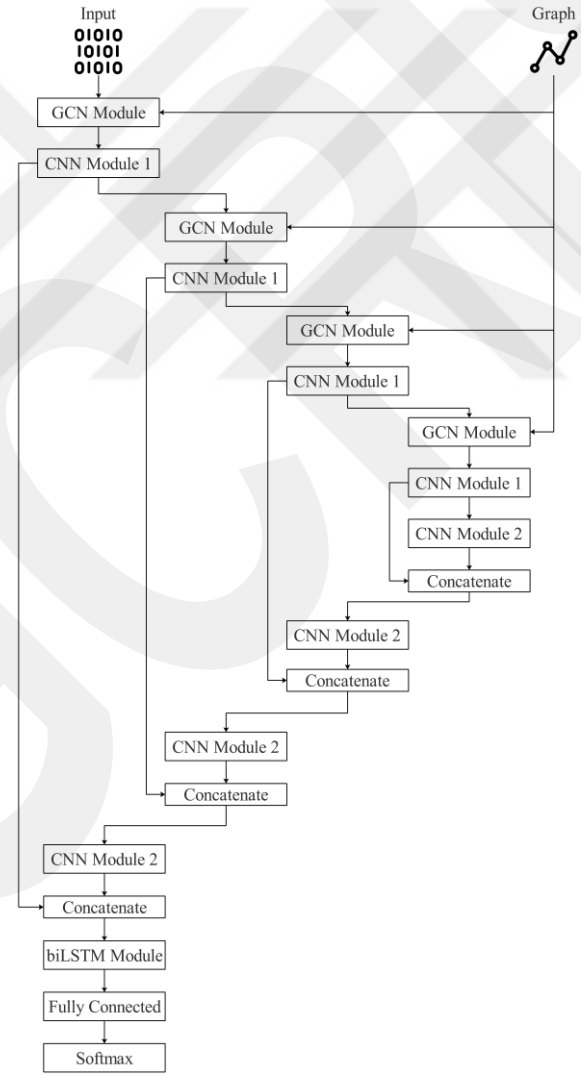


Figure 3.11 Architecture of GraphUnet-SS.

Similar to ProteinUnet [73] model, our proposed architecture also contains blocks based on the U-shaped backbone of the U-Net model [170]. Top side of the model architecture

is considered as as the contracting path and the bottom side (before the last biLSTM module) is considered as the expansive path. These structures can also be considered as the decoder and the encoder sections of the model. Unlike the ProteinUnet, our proposed model (GraphUnet-SS) contains multi graph convolution operations in the contracting path. Therefore, it needs an extra input in the form of a graph. In addition to this, GraphUnet-SS also contains biLSTM module at the end of the U-shaped backbone, because according to the literature, using LSTM layers can improve the accuracy of secondary structure prediction [27], [65], [69]. In GraphUnet-SS model all of the "CNN module 1" blocks in the contracting path are identical. In addition, convolution operations in "CNN module 1" are also identical. Therefore, the number of filters and the kernel size are the same for each convolution operation. Glorot Uniform in Keras library of Python [171] was used with seed as an initializer to make GraphUnet-SS more stable. As mentioned before, Relu was also used in GraphUnet-SS as an activation function at the end of each convolution operation. All of the "CNN module 2" blocks and convolution operations inside the "CNN module 2" are also identical. However, the number of filters parameter was optimized separately for CNN module 1 and CNN module 2. (i.e. there are two different number of filters parameters for the convolution operations). Dimension of the graph node embedding and the number of graph filters are also the same across the GCN modules with the initializer set to Glorot Uniform. For the biLSTM module, the dimensionality of the output space was optimized and "*return_sequences*" parameter was set to True. Default values were used for all the other parameters. A Softmax layer with *sparse_categorical_crossentropy* loss function was used to estimate 3-state secondary structure information. Adam was used as the optimization algorithm for estimating the weight coefficients of neural networks, where *beta_1* parameter was set to 0.95 and *beta_2* parameter to 0.99.

3.7 Hyper-Parameter Optimization

It is a well-known fact that, choosing the right hyper-parameters is important for a machine learning model to perform accurately. Therefore, hyper-parameters of a machine learning model are optimized by selecting different value combinations iteratively and choosing the particular combination that gives the highest performance. In grid search optimization, a parameter grid that contains a finite set of values for each hyper-parameter is defined, and then optimum hyper-parameter configuration is found by evaluating the

performance of the model for each value combination. One disadvantage of grid search is such that some of the intermediate values that are not represented in the parameter grid (and hence missed) may indeed cause the model to perform better if they were included in the parameter grid. To overcome this limitation, a denser parameter grid should be selected. However, the computational cost grows exponentially as the parameter grid contains more and more values. Considering the fact that the proposed models contain many hyper-parameters, grid search would not be the best approach to optimize them. Therefore, in this thesis, the hyper-parameters of the proposed neural network models are optimized using the Bayesian optimization algorithm. Studies show that Bayesian optimization technique outperforms the traditional optimization algorithms [41], [42]. It is also advantageous due to the fact that it can sample intermediate values in the parameter ranges.

Bayesian optimization algorithm was implemented using scikit-optimize library of Python [172]. Gp_minimize method was implemented using the following parameter settings: acq_func='EI' and n_calls=100 [172]. Here *acq_fnc* is the function to minimize over the Gaussian prior and *n_calls* is the number of calls to the function. All the other parameters of *gp_minimize* are set to their default values. This method uses a Gaussian process with two aims: modelling the surrogate function and optimizing the expected probability which is based on improving the existing best solutions through new trials. It assumes that the function values track a multivariate Gaussian distribution. A Gaussian kernel designates the covariance of the function values among the parameters. In each iteration, the next value of a parameter is selected through the acquisition function over the Gaussian prior.

Chapter 4

Experiment Results

4.1 Experiment Results of IGPRED

In order to assess the prediction performance of IGPRED, five different datasets were used as test sets including CullPDB-test, EVAset, CASP10, CASP11, and CASP12. For each benchmark, separate train sets were re-generated from the original CullPDB dataset by performing pairwise BLAST alignments as explained in section 3.1 benchmark datasets. Each dataset had two versions: the first one included structural profiles and the other one did not include structural profiles in the feature set. In addition to this, for the versions that included structural profiles, two dataset versions were further derived: the version that did not include AAindex, which shows in table 3.2, values and the version that included AAindex values in the feature set.

4.1.1 Hyper-parameter optimization

In the first step, the optimum hyper-parameters were found using CullPDB-train dataset. To achieve this, 10% of the proteins in CullPDB-train were randomly selected as the test set for optimization (i.e. validation set called CullPDB-val) and the remaining proteins were selected as the train set for optimization denoted as CullPDB-train-opt. For the IGPRED model, learning rate, the number of filters in convolution layers (n_filters_conv), batch size, the number of epochs, dropout rate, the number of hidden units (i.e. neurons) in dense layers (n_dense), the number of connections in graphs (nconn) and the number of outputs in multi-graph layers (out_dim_gcn) are optimized using the Bayesian optimization technique as explained in section 3.7 hyper-parameter optimization. Note that a separate number of filters parameter is defined for each CNN module (i.e. at the module level), a separate output dimension parameter is defined for each GCN module and a separate number of hidden units parameter for each dense layer. As there are five CNN modules, five GCN modules and two dense layers, considering the

other hyper-parameters as well, a total of 17 hyper-parameters were defined and optimized. table 4.1 shows the lowest and highest values of these hyper-parameters.

Table 4.1 Hyper-parameter ranges of IGPRED used for optimization

Parameter	Lowest	Highest
learning rate	10^{-6}	10^{-1}
n_filters_conv	50	150
batch size	2^2	2^7
epoch	10	200
dropout rate	0	0.6
n_dense	400	1400
nconn	5	50
out_dim_gcn	20	200

Table 4.2 shows the optimized values of the hyper-parameters for both versions of the datasets (with and without using structural profiles in feature sets). In this table, n_filters_conv, n_dense and out_dim_gcn rows show the hyper-parameters for each module in the order shown in figure 3.7. Note that these experiments did not include the AAindex values in the feature set.

Table 4.2 Optimum hyper-parameters for IGPRED

Parameter	Dataset without structural profiles	Dataset with structural profiles
learning rate	0.00011778	1.501×10^{-5}
n_filters_conv	103, 107, 96, 115, 98	119, 113, 115, 101, 112
batch size	4	4
Epoch	96	78
dropout rate	0.5	0.3
n_dense	547, 537	564, 473
Nconn	11	17
out_dim_gcn	123, 189, 87, 63, 71	101, 123, 116, 75, 25

4.1.2 Adding structural profiles

After hyper-parameter optimization, a total of ten different models were trained using the two versions of the five train sets (i.e. with and without structural profiles) and predictions are computed on the corresponding test sets. The same hyper-parameter configuration found for CullPDB-train is used in these experiments. Table 4.3 shows results for the

datasets that did not use structural profiles and table 4.4 includes the results when structural profiles are added to the feature set. In these tables, Accuracy represents the overall accuracy (i.e. Q3 measure), SOV [173] denotes the segment overlap measure, MCC [174] is the Matthew's correlation coefficient. MCC, recall and precision metrics are computed for each secondary structure class in a one versus all setting. CullPDB-train, CullPDB-train-EVAset, CullPDB-train-CASP10, CullPDB-train-CASP11, and CullPDB-train-CASP12 are used to train the models for CullPDB-test, EVAset, CASP10, CASP11, and CASP12, respectively.

Table 4.3 Accuracy measures of IGPRED without using structural profiles

Dataset	**Accuracy**	**SOV**	**MCC 'H'**	**MCC 'E'**	**MCC 'L'**	**Recall 'H'**	**Recall 'E'**	**Recall 'L'**	**Precision 'H'**	**Precision 'E'**	**Precision 'L'**
CullPDB-test	87.80%	83.12%	0.86	0.67	0.76	92.94%	69.43%	87.18%	90.29%	69.72%	89.15%
EVAset	86.12%	76.15%	0.82	0.64	0.72	89.53%	69.25%	86.32%	87.45%	64.27%	88.70%
CASP10	87.24%	75.95%	0.83	0.64	0.73	89.55%	55.62%	91.32%	86.95%	79.46%	88.24%
CASP11	85.26%	74.10%	0.80	0.59	0.68	89.15%	51.74%	88.90%	84.78%	79.98%	86.01%
CASP12	86.04%	78.01%	0.83	0.63	0.72	91.93%	65.92%	85.53%	86.13%	65.76%	89.67%

Table 4.4 Accuracy measures of IGPRED with structural profiles

Dataset	**Accuracy**	**SOV**	**MCC 'H'**	**MCC 'E'**	**MCC 'L'**	**Recall 'H'**	**Recall 'E'**	**Recall 'L'**	**Precision 'H'**	**Precision 'E'**	**Precision 'L'**
CullPDB-test	89.19%	87.15%	0.88	0.68	0.79	93.13%	67.73%	89.91%	92.29%	73.84%	89.24%
EVAset	86.34%	76.35%	0.82	0.61	0.73	89.50%	58.37%	88.31%	87.47%	70.32%	87.49%
CASP10	87.87%	75.19%	0.84	0.68	0.75	89.31%	68.28%	90.39%	88.54%	73.29%	89.76%
CASP11	85.76%	76.48%	0.82	0.61	0.71	88.73%	59.59%	88.64%	87.01%	70.40%	87.28%

CASP 12	86.54 %	77.50%	0.84	0.59	0.73	91.96%	50.66%	89.29%	87.00 %	76.27 %	87.37 %

According to these results, the models that used structural profiles obtained better accuracy rates than the models that did not use structural profiles. Using structural profiles increased the overall accuracy rate by 1.39% for CullPDB-test, 0.22% for EVAset, 0.63% for CASP10, 0.5% for CASP11, 0.50% for CASP12 compared with the models that did not use structural profiles.

When the MCC values of secondary structure classes are compared the highest values are obtained for helices, followed by loops, followed by strands. When the recall values are compared, except for CASP10, the same ordering is obtained: helices come first, followed by loops and then strands. For CASP10, the ordering is loops, followed by helices and then strands. Finally, when the precision values are compared, except for CullPDB-test, loops come first, followed by helices and then strands. For CullPDB-test, helix performance is the best, followed by loops and then strands. In all cases, the accuracy of beta-strands is lowest as compared to helices and loops. This is reasonable because the proposed model architecture can only capture local correlations between amino acids, which is characteristic of helices and loops. The lower accuracy rates for beta-strands can be due to non-local (i.e. distant or long-range) interactions present between amino acids of beta-strand segments. It can be anticipated that if such interactions are predicted a priori (e.g. in the form of contact maps) with sufficient accuracy and used as input to GCN, the accuracy rates of beta-strands may improve.

4.1.3 Adding physico-chemical features from AAindex database

In addition to using structural profile features, adding new physico-chemical properties of the amino acids to the feature set is also explored. For this purpose, 35 AAindex values were selected and added to the feature set that includes structural profiles (among with other feature categories) and train/test experiments are repeated for the five benchmarks. Table 4.5 shows the overall Q3 accuracies of IGPRED on five benchmarks.

Table 4.5 Q3 accuracy of IGPRED when new physico-chemical properties are added to feature set

CullPDB-test	EVAset	CASP10	CASP11	CASP12
89.17%	86.39%	87.66%	85.56%	86.24%

These results are comparable to those presented in table 4.4. Therefore, using AAindex values as additional physico-chemical features did not improve the overall prediction accuracy. Because of these reason, AAindex values were not used to train the other proposed models.

4.1.4 Comparison with the state-of-the-art

In the next step, IGPRED is compared with the state-of-the-art methods in the literature. Table 4.6 shows the overall accuracy of MUFOLD-SS [26], OPUS-TASS [27] and our model on CASP datasets.

Table 4.6 Q3 accuracy comparison between IGPRED and state-of-the-art methods on CASP datasets

Method	CASP10	CASP11	CASP12
MUFOLD-SS	86.49%	85.20%	83.36%
OPUS-TASS	------	------	85.47%
IGPRED without SP	87.24%	85.26%	85.76%
IGPRED with SP	87.87%	85.76%	86.54%

Based on these results, our model outperforms the state-of-the-art methods on all of the CASP datasets. The improvements obtained when structural profiles are included to the feature set are statistically significant according to a two-tailed Z-test [175] with p-values less than 0.01 for all CASP datasets. Note that the results of MUFOLD-SS and OPUS-TASS are taken from the corresponding paper, while our results are obtained on subsets of the CASP datasets (i.e. on CASP proteins that have PDB IDs and that are not short as described in Section 2.2.3). The reason for this difference is because the CASP datasets (including the label assignments) used by MUFOLD-SS are not shared publicly. Based on this, the results presented in table 4.6 contain variance components due to slightly different versions of the datasets being used. Nonetheless, obtaining improvements consistently on all datasets is promising.

4.1.5 Learning Curves

Another factor that can be analyzed is the overfitting behavior of the models. Deep learning models with many layers can be prone to overfitting due to large number of weight coefficients learned during training. In the proposed model architecture, batch normalization and dropout are used as regularization techniques to prevent overfitting. In

order to understand whether a model has learned the patterns in train set well enough and is able to generalize to new examples, learning curves can be used. Figure 4.1 shows the learning curves of the model for different datasets. In all of these experiments, structural profiles were used in the feature set. Each subfigure shows the accuracy of the model with respect to the number of epochs, which is directly proportional to model complexity. These curves show that our model did not suffer from significant amount of overfitting since the test curves follow the train curves closely.

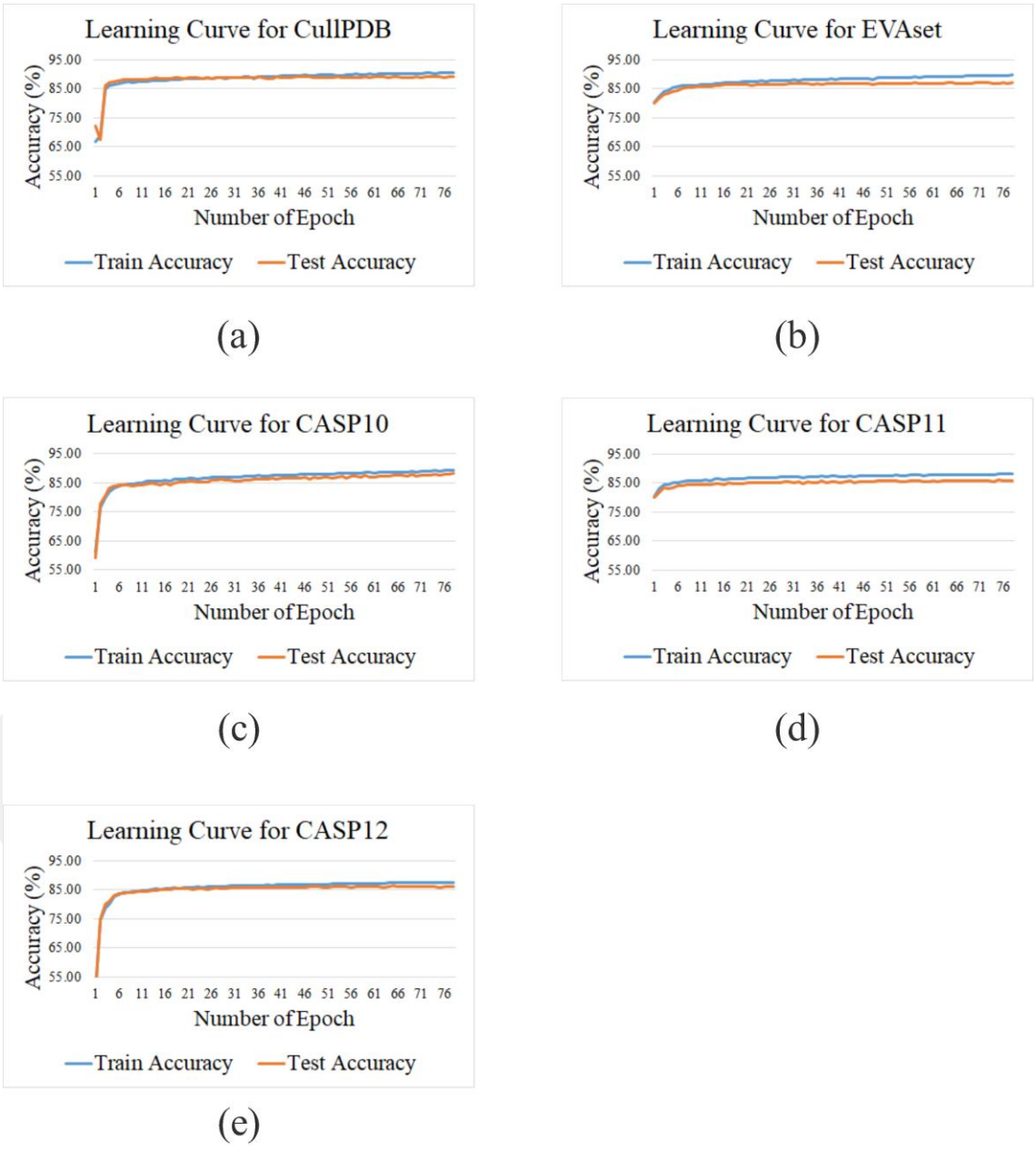


Figure 4.1 Learning curves for the version of IGPRED that uses structural profiles: (a) CullPDB, (b) EVAset, (c) CASP10, (d) CASP11 and (e) CASP12

4.1.6 Accuracy with respect to length of protein

In this thesis, long proteins were split naively using blocks of 700 amino acids as explained in section 3.3 feature generation. This was preferred for computational reasons. In order to analyze whether this type of splitting degrades performance, the overall accuracy values were computed for proteins belonging to different length intervals (including those that are longer than 700 amino acids) in EVAset, which is the only benchmark that has sufficient number of long proteins. The results of this experiment are given in table 4.7.

Table 4.7 Q3 accuracy of IGPRED on EVAset at different length intervals

Length Intervals	Q3 Accuracy
1 – 175 amino acids	86.39%
176 – 350 amino acids	86.44%
351 – 525 amino acids	86.41%
526 – 700 amino acids	86.23%
> 700 amino acids	86.11%

Based on these results, as the length of the protein increases, there is only a minor decrease in the overall prediction accuracy. For example, the accuracy obtained for the fourth length interval is around 0.2% lower than the accuracy of the third length interval both of which contain proteins with no splitting applied. This shows that as the protein length increases, the prediction accuracy may decrease slightly even if there is no splitting due to missed correlations caused by long-range interactions (the convolutional networks developed for IGPRED only model short-range interactions between the amino acids). On the other hand, the accuracy of the last length interval (that contains proteins longer than 700 amino acids) is only 0.1% lower than the accuracy of the fourth interval and only 0.3% lower than the top performing second and third intervals. Based on this result, although the naïve splitting approach ignores domain boundaries and any interactions between the split regions, it can be anticipated that the splitting process has little contribution to the decrease in accuracy. This can be because the proposed model already ignores long-range interactions and the decrease in performance for long proteins can be mainly due to this restriction. If there is an extra degrade in performance due to splitting, that may happen due to missed correlations/interactions around the boundaries of the feature matrix (i.e. at positions close to every 700th amino acid) which may affect a few amino acids only. As a result, the naïve splitting approach can be preferred due to its computational simplicity without degrading performance considerably.

4.2 Experiment Results of IGPRED-MultiTask

For IGPRED-MultiTask, a total of ten benchmark datasets, which include train, validation, CASP12, CASP13, CASPFM, TEST2016, TEST2018, CAMEO93, CAMEO93_HARD and HARD68, were used. The train set is used to train models, the validation set is used to optimize hyper-parameters and the remaining eight sets are used as test sets to measure performance of IGPRED-MultiTask.

4.2.1 Hyper-parameter optimization

In the first step, the optimum hyper-parameters were found using train and validation datasets. For the IGPRED-MultiTask model, learning rate, the number of filters in convolution layers (n_filters_conv), batch size, the number of epochs, dropout rate, the number of hidden units (i.e. neurons) in dense layers (n_denses), the number of connections in graphs (nconn), the number of outputs in multi-graph layers (out_dim_gcn) and number of units for LSTM layers (n_unit_lstm) were optimized using the Bayesian optimization technique as explained in section 3.7 hyper-parameter optimization. Note that a separate number of filters parameter is defined for each CNN module (i.e. at the module level), a separate output dimension parameter is defined for each GCN module, a separate number of hidden units parameter for each dense layer and a separate number of units for each LSTM modules. As there are five CNN modules, five GCN modules, two dense layers, and two LSTM modules considering the other hyper-parameters as well, a total of 19 hyper-parameters were defined and optimized. Table 4.8 shows the lowest and highest values of these hyper-parameters.

Table 4.8 Hyper-parameter ranges of IGPRED-MultiTask used for optimization

Parameter	Lowest	Highest
learning rate	10^{-6}	10^{-1}
n_filters_conv	20	200
batch size	2^{0}	2^{7}
epoch	10	200
dropout rate	0	0.6
n_denses	100	1500
nconn	0	75
out_dim_gcn	20	200
n_unit_lstm	20	200

Table 4.9 shows the optimized values of the hyper-parameters for IGPRED-MultiTask. These values are then used to train the neural network models. In this table, the values of n_filters_conv, n_denses, out_dim_gcn and n_unit_lstm are shown for each module following their sequential order in the architecture of IGPRED-MultiTask. For instance, a total of five n_filters_conv parameters were defined and optimized for CNN Modules 1-5 and the optimized values are presented in this order (i.e. 175 is the optimum value for this parameter for CNN Module 1). Note that these experiments include PSI-BLAST PSSMs, HHblits scores, physico-chemical properties and structural profiles in the feature set.

Table 4.9 Optimum hyper-parameters for IGPRED-MultiTask

Hyper-parameter types	Optimum Hyper-parameter
learning rate	0.000210060076
n_filters_conv	175, 86, 64, 112, 125
batch size	2^6
epoch	156
dropout rate	0.2
n_dense	675, 280
nconn	16
out_dim_gcn	96, 81, 79, 32, 28
n_unit_lstm	58, 30

4.2.2 Performance measures and comparison with the state-of-the-art

After hyper-parameter optimization, the IGPRED-MultiTask is trained on the original training set (i.e. train set) and predictions are computed on a total of eight test sets. Accuracy (ACC) for 3-state secondary structure and mean absolute error (MAE) for torsion angles were used in the literature to evaluate the model performance [26], [27], [72], [95], [99]. The results of these experiments are summarized in table 4.10, in which ACC for 3-state secondary structure and MAE for torsion angles are computed on eight test sets. Note that for each benchmark in table 4.10, a single train and a single test operation are performed. In these experiments, we obtain testing results on multiple independent data sets, which provides an estimate of the variation in performance results with respect to different data set conditions including the difficulty of the dataset.

In this table, IGPRED-MultiTask is our proposed model and is trained using the original training set. IGPRED-MultiTask* represents our proposed model trained on the reduced versions of the original training set (see section 3.1), which is a more difficult experimental setting. Our results are compared with the state-of-the-art methods OPUS-TASS [27], SPOT-1D [29], NetsurfP-2.0 [70] and MUFOLD [26], whenever possible for secondary structure and torsion angle predictions. Based on these results, our model (both IGPRED-MultiTask* and IGPRED-MultiTask) outperforms all the other state-of-the-art methods in all test sets and in all performance metrics. Note that IGPRED-MultiTask has the same experimental conditions as the other state-of-the-art methods (in terms of the training set used). Therefore, it is more convenient to compare IGPRED-MultiTask directly with the state-of-the-art. On the other hand, it is promising to observe that IGPRED-MultiTask* is also better than the state-of-the-art. Since IGPRED-MultiTask* is evaluated on the reduced training sets derived for each test set, the results obtained for IGPRED-MultiTask* can be regarded as the actual performance of our proposed model in the most stringent experimental conditions.

The reason for the improved performance over the state-of-the-art can be due to the following two factors. The first one can be related to the model architecture, which utilizes deep learning models including CNN, mGCN and BiLSTM modules jointly. For instance, one difference between our model and OPUS-TASS is the utilization of mGCN modules by our model, which also is not present in other state-of-the-art methods. The second can be due to the structural profile features employed as input to our model, which may have provided additional useful information.

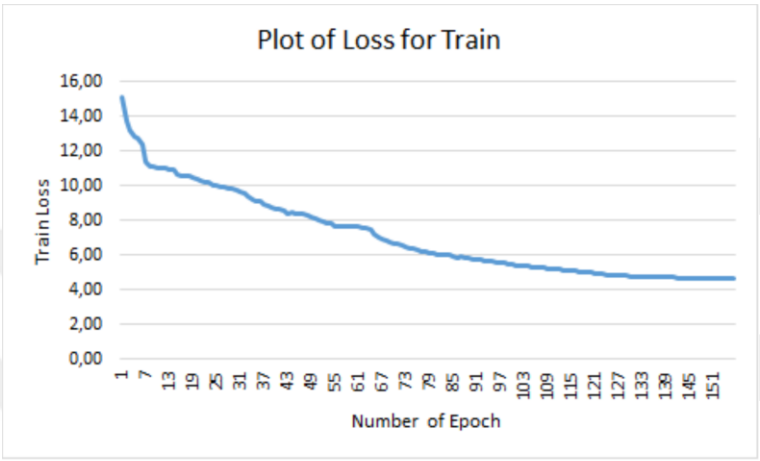
Table 4.10 Comparison of IGPRED-MultiTask with the state-of-the-art methods for secondary structure and torsion angle predictions. ([a]Results are taken from the paper of the OPUS-TASS method)

Models	Accuracy SS3	MAE psi	MAE phi
TEST2016			
SPOT-1D[a]	87.16%	16.27	23.26
OPUS-TASS	87.79%	15.78	22.46
IGPRED-MultiTask*	87.98%	15.13	22.29
IGPRED-MultiTask	**88.29%**	**15.04**	**21.81**
TEST2018			
MUFOLD[a]	84.78%	17.78	27.24
NetsurfP-2.0[a]	85.31%	17.90	26.63

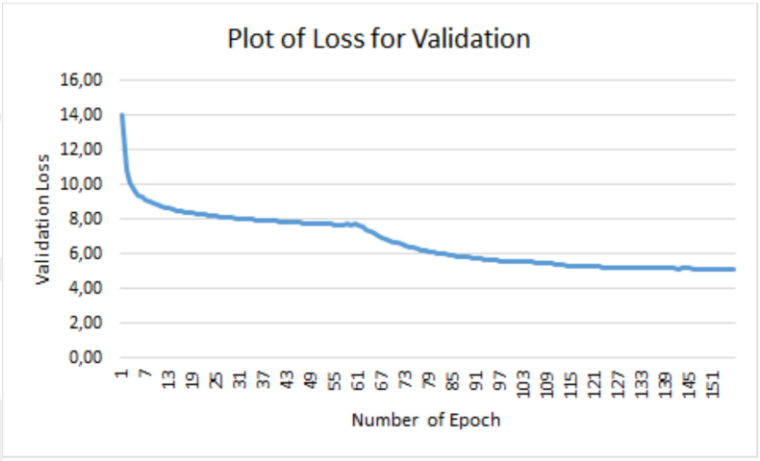
SPOT-1D[a]	86.18%	16.89	24.87
OPUS-TASS	86.84%	16.40	24.06
IGPRED-MultiTask*	87.35%	15.85	23.37
IGPRED-MultiTask	**87.64%**	**15.76**	**23.22**
CASP12			
MUFOLD	83.36%	------	------
SPOT-1D [a]	84.82%	18.44	26.90
OPUS-TASS	85.47%	18.08	25.98
IGPRED-MultiTask*	86.57%	17.63	24.81
IGPRED-MultiTask	**86.61%**	**17.57**	**24.78**
CASP13			
SPOT-1D [a]	86.53%	18.48	26.73
OPUS-TASS	87.62%	17.89	25.93
IGPRED-MultiTask*	88.27%	17.09	24.61
IGPRED-MultiTask	**88.41%**	**17.05**	**24.47**
CASPFM			
SPOT-1D [a]	82.37%	19.39	30.10
OPUS-TASS	83.40%	18.85	28.00
IGPRED-MultiTask*	84.18%	18.29	27.21
IGPRED-MultiTask	**84.24%**	**18.27**	**27.17**
CAMEO93			
SPOT-1D [a]	87.72%	16.89	23.02
OPUS-TASS	89.06%	16.56	21.98
IGPRED-MultiTask*	89.27%	**16.19**	21.74
IGPRED-MultiTask	**89.28%**	16.25	**21.65**
CAMEO93_HARD			
SPOT-1D [a]	82.31%	18.75	31.02
OPUS-TASS	82.56%	18.52	30.17
IGPRED-MultiTask*	**84.11%**	17.60	**27.57**
IGPRED-MultiTask	84.09%	**17.58**	27.64
HARD68			
SPOT-1D [a]	83.79%	18.35	27.77
OPUS-TASS	83.78%	18.03	27.16
IGPRED-MultiTask*	84.61%	17.51	26.54
IGPRED-MultiTask	**84.81%**	**17.38**	**26.32**

4.2.3 Loss Curves

As explained before IGPRED-MultiTask is trained on original training set and predictions are computed on test sets using this model. Figure 4.2 shows the loss curves of IGPRED-MultiTask for secondary structure prediction on training set and validation set. These curves show the loss until the optimum number of epochs after which the training is stopped. According to these figures, model training is performed successfully reaching the optimum validation loss. For the IGPRED-MultiTask* the behavior of losses is observed to be similar to IGPRED-MultiTask



A



B

Figure 4.2 Loss curves for IGPRED-MultiTask. (A) Train loss (B) Validation loss.

4.2.4 Two-tailed Z-test

Based on the performance results and state-of-the-art comparison, the improvements obtained by IGPRED-MultiTask* over the OPUS-TASS method (which is selected as the best method among state-of-the-art methods) are statistically significant according to a

two-tailed Z-test at $p <= 0.05$ for TEST2016, TEST2018, CASP12 (excluding the phi angle predictions), and CAMEO93_HARD (excluding the phi angle predictions). The results on the remaining test sets or prediction tasks can be regarded as comparable. Table 4.11 shows the p-values that are computed for the Z-test experiment that compares IGPRED-MultiTask* and OPUS-TASS results.

Table 4.11 P-values between IGPRED-MultiTask* and OPUS-TASS

Dataset	SS	PHI	PSI
TEST2016	0.001	0.001	0.204
TEST2018	0.045	0.030	0.016
CASP12	0.017	0.357	0.047
CASP13	0.307	0.293	0.131
CASPFM	0.161	0.342	0.280
CAMEO93	0.603	0.342	0.720
CAMEO93 HARD	0.029	0.213	0.005
HARD68	0.2187	0.412	0.441

4.2.5 Solvent accessibility results

Regarding solvent accessibility, relative solvent accessibility labels are available in TEST2016, TEST2018, validation and training set only. Since our model can also predict the relative solvent accessibility information, we evaluated the solvent accessibility prediction performance of our model on TEST2016 and TEST2018 data sets. The mean absolute error of IGPRED-MultiTask* is obtained as 14.09 for TEST2016 and 15.01 for TEST2018 data sets. We are not able to compare these results with the state-of-the-art because to the best of our knowledge there is no solvent accessibility prediction result presented in the literature for TEST2016 and TEST2018.

4.2.6 Error regions of secondary structure prediction

To give some details about error regions, Q3 accuracies are calculated for the amino acids that are at the beginning and the amino acids that are at the end of secondary structural segments. Table 4.12 shows these accuracies for IGPRED-MultiTask* on the eight test sets. This calculation is done for helix (H), strand (E) and loop (L) segments separately and also for all the segments, which is presented in “mean acc” column of table 4.12. In addition to this analysis, Q3 accuracy is calculated for the amino acids that are at the

beginning and at the end of the proteins in validation set which are presented in "acc begin-3" and "acc end-3" columns of table 4.12. For this purpose, three amino acids that are at the beginning and three amino acids that are at the end of each protein are selected. According to these results the proposed model has significantly lower accuracy at the terminals of secondary structure segments and slightly lower accuracy around the N-terminal and C-terminal of the proteins as compared to the accuracies obtained in table 4.10. This shows that it is more difficult to predict secondary structure information at the terminals of secondary structure segments and the at the terminal regions of amino acid chains. This can be due to the fact that these are transition regions for secondary structure elements. As local windows are taken around each amino acid to form the feature vectors, the composition of those vectors at segment ends will include information from multiple segments. Furthermore, the feature vectors at the terminals of segments may be located closer to class boundaries of secondary structure elements. All these factors may make the prediction task more difficult at the terminal regions of structural segments.

Table 4.12 Q3 accuracies of IGPRED-MultiTask* for regions

Dataset	H	E	L	mean acc	acc begin-3	acc end-3
TEST2016	80.52%	73.77%	76.21%	76.58%	87.32%	85.71%
TEST2018	80.01%	74.12%	75.10%	76.81%	86.11%	85.43%
CASP12	79.87%	73.15%	75.22%	76.37%	86.01%	84.12%
CASP13	82.26%	73.86%	77.12%	78.12%	86.12%	83.20%
CASPFM	75.26%	69.15%	79.10%	74.70%	82.11%	80.16%
CAMEO93	83.13%	72.91%	80.01%	78.14%	87.33%	85.14%
CAMEO93 HARD	74.19%	68.94%	80.06%	74.55%	81.06%	79.87%
HARD68	75.90%	69.29%	78.56%	73.86%	81.76%	80.03%

4.2.7 Capabilities of IGPRED-MultiTask in different conditions

In addition to evaluating the performance of IGPRED-MultiTask on benchmark data sets and comparing with the state-of-the-art, we performed several other experiments to analyze the capabilities of our model in different conditions. For this purpose, we derived new versions of IGPRED-MultiTask that contain certain module blocks while excluding others as explained in proposed models section. We also considered removing structural profile matrices from the input feature set. Detailed architecture of these models are shown in supplementary material of the paper for IGPRED-MultiTask [82]. Based on the

experimental results, it can be concluded that using all modules (including CNN, mGCN, and BiLSTM modules), multi-task learning and structural profiles have contribution in improving the accuracy of protein structure prediction tasks studied in this work. Table 4.13 shows the performance measures of all derived models and in table 4.14, we include detailed performance metrics of the proposed model including overall accuracy (Q3 measure), precision, recall, MCC and segment overlap (SOV) scores for protein secondary structure prediction, in which the MCC, recall and precision were computed for each secondary structure segment separately. In both tables, the performance metrics are computed for proteins in validation set.

Table 4.13 Results for the all derived models on the validation set

Model	Acc SS3	psi	phi
IGPRED-MultiTask*	87.85%	16.01	23.04
IGPRED-MultiTask-WO-SP	87.42%	16.09	23.11
IGPRED-SS	87.35%	--------	--------
IGPRED-TA	--------	16.24	23.40
IGPRED-GCN-free	87.20%	16.63	23.50
IGPRED-GCN-1	87.56%	16.32	23.40
IGPRED-GCN-2	87.53%	16.30	23.47
IGPRED-GCN-3	87.56%	16.33	23.20
IGPRED-GCN-4	87.78%	16.12	23.09
IGPRED-CNN-free	87.02%	16.65	23.62
IGPRED-CNN-1	86.95%	16.67	23.63
IGPRED-CNN-2	87.23%	16.36	23.51
IGPRED-CNN-3	87.61%	16.15	23.14
IGPRED-CNN-4	87.75%	16.07	23.07
IGPRED-BiLSTM-free	86.51%	17.01	23.82
IGPRED-BiLSTM-1	87.35%	16.12	23.12

Table 4.14 Detailed results of original model for secondary structure prediction on validation set

Accuracy	SOV	MCC	MCC	MCC	Recall 'H'	Recall 'E'	Recall 'L'	Precision 'H'	Precision 'E'	Precision 'L'
87.85%	75.14 %	0.83	0.67	0.74	89.51 %	68.21 %	90.27 %	88.61%	73.18%	89.67%

4.3 Experiment Results of GraphUnet-SS

As mentioned in section 3.1 benchmark datasets, similar to IGPRED model, a total of five datasets were used in GraphUnet-SS in which four of them were used to assess the performance of the model and the other one was used to generate validation and train sets. In the first phase CullPDB dataset was divided into two parts, which are CullPDB-train and CullPDB-validation, using BLAST algorithm with configuration that was mentioned in benchmark dataset section. After that, separate train sets were generated for each dataset, which are called CullPDB-train-EVAset, CullPDB-train-CASP10, CullPDB-train-CASP11 and CullPDB-train-CASP12, from CullPDB-train using BLAST algorithm with same configuration. The number of proteins and the number of amino acids for each dataset were given in section 3.1 benchmark datasets.

4.3.1 Hyper-parameter optimization

In GraphUnet-SS, number of filters for CNN module 1 (n_unit_conv_1), number of filters for CNN module 2 (n_unit_conv_2), dimension of graph node embedding for GCN module (n_unit_gcn_1), dimensionality of the output space for biLSTM module (n_unit_lstm_1), number of neurons for fully connected dense layer (n_unit_dense_1), dropout rate for all layers (dr_rate), learning rate for optimizer (lr), number of epochs (epoch) and batch size (batch) were optimized. Note that the same *dr_rate* was used for all layers and all CNN module 1, CNN module 2 and GCN modules were identical within themselves. Activation function of classification layer was Softmax, activation function for all remaining layers were Relu, loss function was sparse categorical cross entropy and weight optimizer was Adam. Non-optimized parameters were used by default. Unlike the IGPRED and IGPRED-MultiTask models, GraphUnet-SS did not include number of connections parameters, because contact map prediction was used to generate graph input for the GCN modules. Table 4.15 shows the lowest and highest values of these hyper-parameter.

Table 4.15 Hyper-parameter ranges of GraphUnet-SS used for optimization

Parameter	Lowest	Highest
lr	10^{-6}	10^{-1}
epoch	5	110
batch	1	16

dr_rate	0	0.6
n_unit_conv_1	20	180
n_unit_conv_2	20	180
n_unit_gcn_1	20	120
n_unit_lstm_1	10	60
n_unit_dense_1	100	1000

In the second phase, hyper-parameters of GraphUnet-SS were optimized using CullPDB-train and CullPDB-validation datasets where CullPDB-train was used to train model and CullPDB-validation was used to assess performance of model with determined hyper-parameters. For this purpose, scikit-optimize library were used with the parameter spaces shown in table 4.15. Table 4.16 shows the optimum hyper-parameters for GraphUnet-SS found using Bayesian optimization.

Table 4.16 Optimum hyper-parameters for GraphUnet-SS

Hyper-parameter types	Optimum Hyper-parameter
lr	0.008956293630918124
epoch	49
batch	8
dr_rate	0.2
n_unit_conv_1	95
n_unit_conv_2	137
n_unit_gcn_1	42
n_unit_lstm_1	40
n_unit_dense_1	446

4.3.2 Learning curves for training phase

A model, which is configured with optimum hyper-parameters, for each dataset was trained at the end of the hyper-parameter optimization. For this purpose, 4 different models were trained using CullPDB-train-EVAset, CullPDB-train-CASP10, CullPDB-train-CASP11 and CullPDB-train-CASP12 as train datasets for EVAset, CASP10, CASP11 and CASP12, respectively. For each model CullPDB-validation was used as the validation dataset. The optimum hyper-parameters listed in table 4.16 were used and the remaining hyper-parameters were assigned as explained in section 3.1 hyper parameter optimization. In addition to this two different callback functions, which are *lr_callback*

and *early_stopping_callback* from Keras [176], were used during training. Learning rate of the model was divided by 2 using *lr_callback*, when no improvement was seen on validation loss for 2 epochs. Training was stopped by using *early_stopping_callback*, when no improvement was seen on validation loss for 6 epochs. Figure 4.3 shows the training loss, validation loss and learning rate of each model during the training phase. It's obviously seen in this figure that, as the number of epochs increases, training loss and validation loss resemble each other. Although optimum number for epochs is 49, number of epochs in the figures are smaller than 49. The reason of this is because the original optimum number of epochs was found using Bayesian optimization on CullPDB-train and CullPDB-validation datasets but the new optimums were found using early-stopping (which can be seen as a second phase of optimization for the number of epochs) on the specific training set derived for each test set and CullPDB-validation

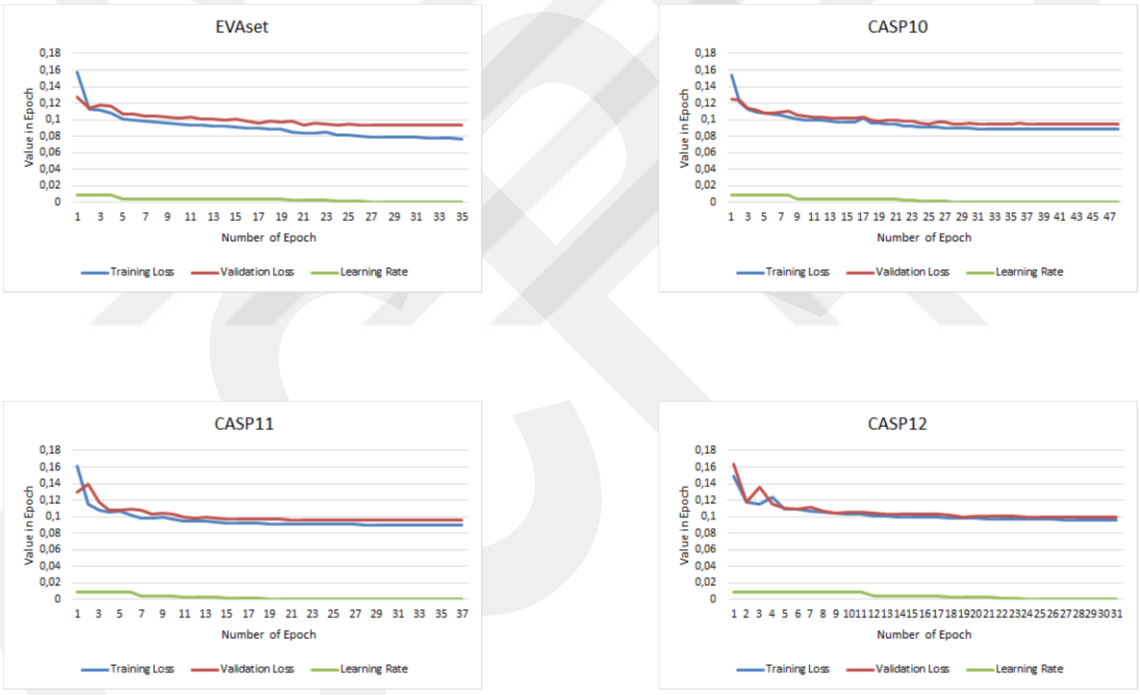


Figure 4.3 Training Loss, Validation Loss and Learning Rate on Each Epoch for EVAset, CASP10, CASP11 and CASP12

4.3.3 Performance measures

After training phase, the performance of GraphUnet-SS, which was trained separately using CullPDB-train-EVAset, CullPDB-train-CASP10, CullPDB-train-CASP11 and CullPDB-train-CASP12, was assessed on EVAset, CASP10, CASP11 and CASP12 respectively. The performance measures of GraphUnet-SS are shown in table 4.17, which

include Precision, Recall, Q3 Accuracy, the segment overlap score (SOV) [173] and the Matthews correlation coefficient (MCC) [174].

Table 4.17 Accuracy measures of GraphUnet-SS

Dataset	Q3 Accuracy	MCC 'H'	MCC 'E'	MCC 'L'	Precision 'H'	Precision 'E'	Precision 'L'	Recall 'H'	Recall 'E'	Recall 'L'	SOV
EVAset	86.82 %	0.84	0.67	0.75	88.22 %	70.96 %	87.86 %	89.70 %	61.09 %	88.66 %	79.49 %
CASP10	87.88 %	0.84	0.67	0.75	89.23 %	75.25 %	89.00 %	88.62 %	65.14 %	91.27 %	78.26 %
CASP11	86.30 %	0.82	0.64	0.72	88.18 %	72.53 %	87.29 %	87.70 %	62.62 %	89.61 %	75.14 %
CASP12	87.03 %	0.85	0.63	0.74	88.50 %	69.66 %	88.74 %	91.64 %	62.66 %	88.33 %	78.26 %

According to the results in this table, the best Q3 accuracy was obtained on CASP10 and the worst Q3 accuracy was obtained on EVAset. When the MCC values of secondary structure classes are compared the highest values are obtained for helices, followed by loops, followed by strands. When the recall values are compared, except for CASP10 and CASP11, the same ordering is obtained: helices come first, followed by loops and then strands. For CASP10 and CASP11, the ordering is loops, followed by helices and then strands. Finally, when the precision values are compared, except for CASP12, helices come first, followed by loops and then strands. For CASP12, loops performance is the best, followed by helices and then strands. In all cases, the accuracy of beta-strands is lowest as compared to helices and loops. These results are similar to the results obtained for IGPRED that shows in table 4.4.

4.3.4 Capabilities of GraphUnet-SS in different conditions

As can be seen from the architecture, GraphUnet-SS consist of several layers including CNN, GCN and biLSTM modules. Other versions of GraphUnet-SS were derived by excluding some layers or by changing the depth of model. For this purpose, a total of seven models, which include GraphUnet-SS-biLSTM-free, GraphUnet-SS-GCN-free, GraphUnet-SS-CNN-free, GraphUnet-SS-depth-2, GraphUnet-SS-depth-3, GraphUnet-SS-depth-5 and GraphUnet-SS-depth-6, were generated. GraphUnet-SS-biLSTM-free is the same version of GraphUnet-SS, however it did not include the biLSTM module before

the fully connected layer. Similar to GraphUnet-SS-biLSTM-free, GraphUnet-SS-GCN-free is the version of the original model that did not include the GCN modules and GraphUnet-SS-CNN-free is the version of original model that did not include the CNN modules. It is considered that GraphUnet-SS has a depth of four, because the number of contracting and expansive paths are four. GraphUnet-SS-depth-2 is the version of original model where the number of contracting and expansive paths are two. Similar to GraphUnet-SS-depth-2, GraphUnet-SS-depth-3, GraphUnet-SS-depth-5 and GraphUnet-SS-depth-6 were generated by changing number of contracting and expansive paths. Table 4.18 shows the Q3 accuracy of each model.

Table 4.18 Q3 accuracies of various model generated from GraphUnet-SS

Model	EVAset	CASP10	CASP11	CASP12
GraphUnet-SS-CNN-free	86.46%	87.52%	85.92%	86.75%
GraphUnet-SS-GCN-free	86.41%	87.45%	85.96%	86.71%
GraphUnet-SS-biLSTM-free	86.85%	87.44%	86.09%	86.89%
GraphUnet-SS-depth-2	86.45%	87.62%	85.81%	86.32%
GraphUnet-SS-depth-3	86.53%	87.56%	85.70%	86.64%
GraphUnet-SS-depth-5	86.84%	87.83%	86.06%	86.95%
GraphUnet-SS-depth-6	86.68%	87.73%	86.13%	86.71%

According to these result, using all modules, which include CNN, GCN and biLSTM layers with a depth of four obtained the best accuracy for CASP10, CASP11 and CASP12. Although, GraphUnet-SS-biLSTM-free and GraphUnet-SS-depth-5 models obtained slightly better Q3 accuracy for EVAset than the original model, GraphUnet-SS has better performance in the vast majority of datasets. In addition to this, GraphUnet-SS is faster than the GraphUnet-SS-depth-5, because of the computational reasons.

4.3.5 Comparison with the state-of-the-art

In the next step, our model is compared with the state-of-the-art methods in the literature. Table 4.19 shows the comparison of GraphUnet-SS with ProteinUnet [73] and IGPRED [28] on EVAset, CASP10, CASP11 and CASP12, with MUFOLD-SS [26] on CASP10, CASP11 and CASP12 and with OPUS-TASS [27] on CASP12 datasets. According to these results, GraphUnet-SS outperforms the OPUS-TASS on CASP12, MUFOLD-SS on all CASP datasets and ProteinUnet and IGPRED on all of the four datasets. Note that, the result of OPUS-TASS, MUFOLD-SS and IGPRED were taken from the

corresponding papers. ProteinUnet model was re-trained by us using similar steps as in GraphUnet-SS and its accuracy is computed using the newly trained model.

Table 4.19 Q3 accuracy comparison of GraphUnet-SS with the models in the literature.

Model	EVAset	CASP10	CASP11	CASP12
OPUS-TASS	---------	---------	---------	85.47%
MUFOLD-SS	---------	86.49%	85.20%	83.36%
ProteinUnet*	86.35%	87.26%	85.72%	86.54%
IGPRED	86.34%	87.87%	85.76%	86.54%
GraphUnet-SS	**86.82%**	**87.88%**	**86.30%**	**87.03%**

Chapter 5

Conclusions and Future Prospects

5.1 Conclusions

Protein structure prediction is one of the most challenging problems in the bioinformatics field. Protein structure prediction refers to the prediction of the 3-D structure of a protein. Because there are several difficulties in predicting the 3-D structure directly, preliminary predictions of protein secondary structure (PSSP), solvent accessibility (SAP) and torsion angles (TAP) are made. Studies show that, using the output of multiple sequence alignment algorithm, physico-chemical properties of amino acids and structural profiles as features improves the accuracy of PSSP, SAP and TAP [10], [35], [36], [55], [56]. As well as the feature sets, machine learning models also effect the prediction results. In these days machine deep learning models achieved significant improvement on PSSP, SAP and TAP [26], [27], [31], [69], [102].

In this thesis, three novel deep learning models were proposed for PSSP, SAP and TAP. Firstly, a rich feature set was generated using several alignment algorithms, structural properties and other properties of amino acids. For this purpose, target proteins were aligned using PSI-BLAST and HHBlits algorithms separately. As a result of these process, 20 PSSMs values were computed with PSI-BLAST and 30 scores were computed with HHBlits. In addition to this structural profiles of solvent accessibility and secondary structure were computed as the weighted average of label frequencies of the template proteins obtained using HHBlits. In the final step of the feature extraction phase, seven physico-chemical properties and 35 AAindex values were added to feature set for each amino acid.

Neural network models have many hyper-parameters such as learning rate, dropout rate, batch size and number of epochs. Unlike the traditional neural network models, deep learning models have several layers and each layer has its own hyper-parameters. It is known that hyper-parameters are among the most important factors that affect the

performance of model. Therefore, using optimum hyper-parameters is one of the crucial steps for deep learning models. In grid search technique, which is used for parameter optimization, parameter space is squeezed within a certain range. Moreover, if the number of parameters is large as in deep learning, grid search technique can take too much time. Because of this reason, hyper-parameters of the proposed models in this thesis are optimized using the Bayesian optimization technique, which is faster and better than the grid search [41], [42].

In the first study, a novel deep learning model to predict secondary structure was developed using CNN and GCN modules. In this model, each CNN module consists of six different convolutional layers with kernel sizes (1,M), (3,M), (5,M), (9,M), (11,M) and (15,M) that are connected in parallel as in inception module and each GCN module consists of multi-graph convolution operation. Because this model consists of a multi-graph convolution operation, it needs a graph as an extra input. For this context, a graph for each protein was generated using neighbor information of amino acids. It is assumed that amino acids which are close to each other in a 1-D sequence interacts each other in 3D space. This model was tested on five datasets including CullPDB-test, EVAset, CASP10, CASP11 and CASP12. For this purpose, a separate train set for each test set was generated with pairwise BLAST alignment using CullPDB-train set. Experiments were done on three different feature combinations. The first combination includes PSI-BLAST PSSMs, HHBlits scores, seven physico-chemical properties. In the second combination, structural profiles were added as extra features and finally in the third combination AAindex features were also added. Experiment results show that best accuracies were added using second combination and 89.19%, 86.34%, 87.87%, 85.76% and 86.54% Q3 accuracies were obtained for CullPDB-test, EVAset, CASP10, CASP11 and CASP12 respectively. In addition to this, accuracies were computed with respect to the length of the target protein and it is seen that the best accuracies were obtained for proteins that contain between 175 and 180 amino acids. It has been shown that our model outperformed the MUFOLD-SS [26] and OPUS-TASS [27] methods.

In the second study, the first model was updated by adding biLSTM layers and it was extended using the multi-tasking approach to predict solvent accessibility and torsion angles. CNN and GCN modules were the same as the first model and biLSTM modules contained forward and backward LSTM modules to capture long range interactions of amino acids. In this model, the same benchmark datasets as the OPUS-TASS [27] model were used to assess performance of our model including TEST2016, TEST2018,

CAMEO93, CAMEO93_HARD, CASP12, CASP13, CASPFM, HARD68, validation and train. Training set is used for model training and validation set is used as test data for hyper-parameter optimization, and the remaining were used as test sets. Similar to the first model, separate train sets were obtained for each test set by applying pairwise BLAST alignments. According to the experimental results, our model obtained better Q3 accuracies for PSSP and mean absolute error for phi and psi angles than MUFOLD-SS [26], OPUS-TASS [27], SPOT-1D [29] and NetsurfP-2.0 [70]. Because solvent accessibility labels were available for TEST2016 and TEST2018 datasets, solvent accessibility prediction was performed on these datasets only. Mean absolute errors of 14.09 and 15.01 were achieved for TEST2016 and TEST2018, respectively. However, SAP is not compared with the state-of-the-art models because to the best of our knowledge there is no solvent accessibility prediction result presented in the literature for these datasets. In this study, several other versions of the model were also generated to measure the effect of each module on performance. In addition to these analysis, Q3 accuracies are calculated for the amino acid that is at the beginning and the amino acid that is at the end of secondary structural segments and results shows that amino acids at the terminal points had lower accuracy than the amino acids that are located in between.

In the last study, a novel deep model was developed based on the U-net architecture using CNN, GCN and biLSTM modules for PSSP. Unlike the other models, a graph was generated using contact map prediction, which enables to incorporate long-range interactions between amino acids. Similar to the second study, other version of the model was generated by excluding some layers or by changing the depth of the model. CullPDB-train dataset was used to generate training datasets and EVAset, CASP10, CASP11 and CASP12 were used to assess the performance of models. The best accuracies of the models were achieved as 86.85%, 87.83%, 86.13 and 86.95% for EVAset, CASP10, CASP11 and CASP12, respectively. According to the experimental results, our model outperformed OPUS-TASS [27], MUFOLD-SS [26] and ProteinUnet [73]. Note that, in this experiment, the result of OPUS-TASS [27] and MUFOLD-SS [26] are taken from the corresponding papers and the result of ProteinUnet [73] was obtained by us by re-training this model on our training sets and computing the accuracies on test sets.

5.2 Societal Impact and Contribution to Global Sustainability

Thanks to Human Genome Project [4], a large number of protein sequences were generated day by day, however the 3-D structure of many is unknown. The proteins whose 3-D structure has been experimentally solved accounts for less than 0.6% of the known proteins. This proportion was 0.7%, 1,2% and 2% in 2008, 2007 and 2004 respectively [177]. When the overall growth of the released structures per year is examined, it is observed that the proportion of proteins with known structures gradually decreases [178]. Besides that, the cost of experimentally solving the structure of a new protein is estimated to be around $100,000 [179]. Proteins are used in many areas such as designing novel drugs or enzymes. In addition to this, they are the primary source of nutrition for the living organisms. Climate change affects the yield and quality of the agriculture products, which affects the proteins contained in these resources [180]. It is foreseen that artificial food production will increase with the decrease in soil fertility. If the relationship between the amino acid sequence and the function of proteins can be understood completely, it will be possible to design new proteins, understand the working principles of proteins, and modify proteins to perform specific tasks. Since there is a close relationship between the structure and function of proteins, structure information is important to understand the function of a protein. Because of all these reasons it is useful to predict the 3-D structure of proteins more accurately. In this thesis, novel deep learning models were developed for protein secondary structure, solvent accessibility and torsion angle predictions. Performance evaluations on several benchmark test sets show that our models outperformed state-of-the-art. Because secondary structure, solvent accessibility and torsion angle information can be used for 3-D structure prediction, more accurate prediction of these properties will potentially improve the accuracy of 3-D structure prediction methods. With the contribution of the thesis to the prediction of protein structure, it also contributes to the sustainability of the world due to the reasons mentioned.

5.3 Future Prospects

All studies in this thesis focus on predicting structural properties of proteins, which is useful for 3-D protein structure prediction. Although significant improvements were obtained for PSSP, SAP and TAP, there is still room for further improvement. As a future work, we are planning to upgrade our models using other deep learning models such as attention and transformers layers. We are also aiming to combine the best performing models using ensembles to further improve the prediction accuracy.

One of the main factors affecting the performance of our methods is the graph used as input for GCN modules. In the first two studies, local neighbor information was used to generate the input graphs and in the last study, graphs were generated using contact map prediction, which allowed us to incorporate long-range interactions between amino acids. The first and the third studies concentrated on PSSP, which were analyzed on the same datasets. Experimental results showed that the third study outperformed the first study. Therefore, it can be argued that generating input graphs using more advanced approaches may have the potential to improve the performance of PSSP, SAP and TAP. Because of this reason, as a future work, we will try to make more accurate contact map predictions, which will allow us to generate more accurate input graphs for GCN modules of our deep learning models.

In our prediction models, we used structural profiles as a subset of the input features. To derive these profiles, we used the HHblits alignment method. We set the percentage of sequence similarity threshold to 20, which includes structurally distant templates only. This represents the most stringent experimental condition. As a future work, we are planning to compute structural profiles more accurately using more advanced alignment methods. Furthermore, we would like to train our models using structural profiles that are computed using other values of the sequence similarity threshold. This might enable our models to perform more accurately when structurally similar templates are available.

Deep learning models benefit considerably from large training sets. When more data samples are available in training set, this might potentially improve the prediction accuracy further. For this purpose, we are planning to increase the number of proteins in our training sets by including newer proteins that are added to Protein Data Bank, re-train our models and test their performance on test sets. We can also form newer test sets using proteins published in recent CASP competitions or using recent proteins that are deposited to PDB.

The main aim of all the studies in this thesis is to contribute to the 3-D structure prediction. Because of this reason, after getting additional improvements in predicting structural properties of proteins, we are planning to incorporate our prediction models into the pipeline of state-of-the-art 3-D structure prediction methods and improve the accuracy of 3-D protein structure prediction. All the methods developed will also be made available as a web service application and will be integrated to our existing web server at http://psp.agu.edu.tr.

BIBLIOGRAPHY

[1] "Protein yapısı," *Vikipedi*. Mar. 27, 2022. Accessed: Jun. 30, 2022. [Online]. Available: https://tr.wikipedia.org/w/index.php?title=Protein_yap%C4%B1s%C4%B1&oldid=27438129

[2] "Amino asit," *Vikipedi*. Apr. 01, 2022. Accessed: Jun. 30, 2022. [Online]. Available: https://tr.wikipedia.org/w/index.php?title=Amino_asit&oldid=27488924

[3] "Amino acid," *Wikipedia*. Jun. 25, 2022. Accessed: Jun. 30, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Amino_acid&oldid=1094963538

[4] "İnsan Genom Projesi," *Vikipedi*. May 12, 2022. Accessed: Jul. 01, 2022. [Online]. Available: https://tr.wikipedia.org/w/index.php?title=%C4%B0nsan_Genom_Projesi&oldid=27767197

[5] "Scientists Say: Amino Acid," *Science News Explores*, Dec. 04, 2017. https://www.snexplores.org/article/scientists-say-amino-acid (accessed Jul. 23, 2022).

[6] X.-Q. Yao, H. Zhu, and Z.-S. She, "A dynamic Bayesian network approach to protein secondary structure prediction," *BMC Bioinformatics*, vol. 9, no. 1, p. 49, Jan. 2008, doi: 10.1186/1471-2105-9-49.

[7] W. Yang, K. Wang, and W. Zuo, "A fast and efficient nearest neighbor method for protein secondary structure prediction," in *2011 3rd International Conference on Advanced Computer Control*, Jan. 2011, pp. 224–227. doi: 10.1109/ICACC.2011.6016402.

[8] G. Pollastri, A. J. Martin, C. Mooney, and A. Vullo, "Accurate prediction of protein secondary structure and solvent accessibility by consensus combiners of sequence and structure information," *BMC Bioinformatics*, vol. 8, no. 1, p. 201, Jun. 2007, doi: 10.1186/1471-2105-8-201.

[9] J. Martin, J.-F. Gibrat, and F. Rodolphe, "Analysis of an optimal hidden Markov model for secondary structure prediction," *BMC Struct. Biol.*, vol. 6, no. 1, p. 25, Dec. 2006, doi: 10.1186/1472-6807-6-25.

[10] Z. Aydin, A. Singh, J. Bilmes, and W. S. Noble, "Learning sparse models for a dynamic Bayesian network classifier of protein secondary structure," *BMC Bioinformatics*, vol. 12, no. 1, p. 154, May 2011, doi: 10.1186/1471-2105-12-154.

[11] G. Pollastri and A. McLysaght, "Porter: a new, accurate server for protein secondary structure prediction," *Bioinformatics*, vol. 21, no. 8, pp. 1719–1720, Apr. 2005, doi: 10.1093/bioinformatics/bti203.

[12] B. Yang, Q. Wu, Z. Ying, and H. Sui, "Predicting protein secondary structure using a mixed-modal SVM method in a compound pyramid model," *Knowl.-Based Syst.*, vol. 24, no. 2, pp. 304–313, Mar. 2011, doi: 10.1016/j.knosys.2010.10.002.

[13] A. A. Salamov and V. V. Solovyev, "Prediction of Protein Secondary Structure by Combining Nearest-neighbor Algorithms and Multiple Sequence Alignments," *J. Mol. Biol.*, vol. 247, no. 1, pp. 11–15, Mar. 1995, doi: 10.1006/jmbi.1994.0116.

[14] M. H. Zangooei and S. Jalili, "Protein secondary structure prediction using DWKF based on SVR-NSGAII," *Neurocomputing*, vol. 94, pp. 87–101, Oct. 2012, doi: 10.1016/j.neucom.2012.04.015.

[15] G. Pollastri, P. Baldi, P. Fariselli, and R. Casadio, “Prediction of coordination number and relative solvent accessibility in proteins,” *Proteins Struct. Funct. Bioinforma.*, vol. 47, no. 2, pp. 142–153, 2002, doi: 10.1002/prot.10069.
[16] G. Pugalenthi, K. Kumar Kandaswamy, K.-C. Chou, S. Vivekanandan, and P. Kolatkar, “RSARF: Prediction of Residue Solvent Accessibility from Protein Sequence Using Random Forest Method,” *Protein Pept. Lett.*, vol. 19, no. 1, pp. 50–56, Jan. 2012, doi: 10.2174/092986612798472875.
[17] K. Joo, S. J. Lee, and J. Lee, “Sann: Solvent accessibility prediction of proteins by nearest neighbor method,” *Proteins Struct. Funct. Bioinforma.*, vol. 80, no. 7, pp. 1791–1797, 2012, doi: 10.1002/prot.24074.
[18] R. Adamczak, A. Porollo, and J. Meller, “Accurate prediction of solvent accessibility using neural networks–based regression,” *Proteins Struct. Funct. Bioinforma.*, vol. 56, no. 4, pp. 753–767, 2004, doi: 10.1002/prot.20176.
[19] E. Faraggi, B. Xue, and Y. Zhou, “Improving the prediction accuracy of residue solvent accessibility and real-value backbone torsion angles of proteins by guided-learning through a two-layer neural network,” *Proteins Struct. Funct. Bioinforma.*, vol. 74, no. 4, pp. 847–856, 2009, doi: 10.1002/prot.22193.
[20] O. Zimmermann and U. H. E. Hansmann, “Support vector machines for prediction of dihedral angle regions,” *Bioinformatics*, vol. 22, no. 24, pp. 3009–3015, Dec. 2006, doi: 10.1093/bioinformatics/btl489.
[21] R. Kuang, C. S. Leslie, and A.-S. Yang, “Protein backbone angle prediction with machine learning approaches,” *Bioinformatics*, vol. 20, no. 10, pp. 1612–1621, Jul. 2004, doi: 10.1093/bioinformatics/bth136.
[22] E. Faraggi, Y. Yang, S. Zhang, and Y. Zhou, “Predicting Continuous Local Structure and the Effect of Its Substitution for Secondary Structure in Fragment-Free Protein Structure Prediction,” *Structure*, vol. 17, no. 11, pp. 1515–1527, Nov. 2009, doi: 10.1016/j.str.2009.09.006.
[23] Z. Aydin, J. Thompson, J. Bilmes, D. Baker, and W. S. Noble, “Protein Torsion Angle Class Prediction by a Hybrid Architecture of Bayesian and Neural Networks,” p. 7.
[24] C. Mooney, A. Vullo, and G. Pollastri, “Protein Structural Motif Prediction in Multidimensional ø-ψ Space Leads to Improved Secondary Structure Prediction,” *J. Comput. Biol.*, vol. 13, no. 8, pp. 1489–1502, Oct. 2006, doi: 10.1089/cmb.2006.13.1489.
[25] P. Kountouris and J. D. Hirst, “Prediction of backbone dihedral angles and protein secondary structure using support vector machines,” *BMC Bioinformatics*, vol. 10, no. 1, p. 437, Dec. 2009, doi: 10.1186/1471-2105-10-437.
[26] C. Fang, Y. Shang, and D. Xu, “MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction,” *Proteins Struct. Funct. Bioinforma.*, vol. 86, no. 5, pp. 592–598, 2018, doi: 10.1002/prot.25487.
[27] G. Xu, Q. Wang, and J. Ma, “OPUS-TASS: a protein backbone torsion angles and secondary structure predictor based on ensemble neural networks,” *Bioinformatics*, vol. 36, no. 20, pp. 5021–5026, Dec. 2020, doi: 10.1093/bioinformatics/btaa629.
[28] Y. Görmez, M. Sabzekar, and Z. Aydın, “IGPRED: Combination of convolutional neural and graph convolutional networks for protein secondary structure prediction,” *Proteins Struct. Funct. Bioinforma.*, vol. 89, no. 10, pp. 1277–1288, 2021, doi: 10.1002/prot.26149.
[29] J. Hanson, K. Paliwal, T. Litfin, Y. Yang, and Y. Zhou, “Improving prediction of protein secondary structure, backbone angles, solvent accessibility and contact numbers by using predicted contact maps and an ensemble of recurrent and residual

convolutional neural networks," *Bioinformatics*, vol. 35, no. 14, pp. 2403–2410, Jul. 2019, doi: 10.1093/bioinformatics/bty1006.
[30] B. Zhang, L. Li, and Q. Lü, "Protein Solvent-Accessibility Prediction by a Stacked Deep Bidirectional Recurrent Neural Network," *Biomolecules*, vol. 8, no. 2, Art. no. 2, Jun. 2018, doi: 10.3390/biom8020033.
[31] M. Kaleel, M. Torrisi, C. Mooney, and G. Pollastri, "PaleAle 5.0: prediction of protein relative solvent accessibility by deep learning," *Amino Acids*, vol. 51, no. 9, pp. 1289–1296, Sep. 2019, doi: 10.1007/s00726-019-02767-6.
[32] J. Lyons *et al.*, "Predicting backbone Cα angles and dihedrals from protein sequences by stacked sparse auto-encoder deep neural network," *J. Comput. Chem.*, vol. 35, no. 28, pp. 2040–2046, 2014, doi: 10.1002/jcc.23718.
[33] C. Fang, Y. Shang, and D. Xu, "Prediction of Protein Backbone Torsion Angles Using Deep Residual Inception Neural Networks," *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 16, no. 3, pp. 1020–1028, May 2019, doi: 10.1109/TCBB.2018.2814586.
[34] C. Mirabello and G. Pollastri, "Porter, PaleAle 4.0: high-accuracy prediction of protein secondary structure and relative solvent accessibility," *Bioinformatics*, vol. 29, no. 16, pp. 2056–2058, Aug. 2013, doi: 10.1093/bioinformatics/btt344.
[35] 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," *Bioinformatics*, vol. 28, no. 1, pp. 32–39, Jan. 2012, doi: 10.1093/bioinformatics/btr611.
[36] Z. Aydin, N. Azginoglu, H. I. Bilgin, and M. Celik, "Developing structural profile matrices for protein secondary structure and solvent accessibility prediction," *Bioinformatics*, vol. 35, no. 20, pp. 4004–4010, Oct. 2019, doi: 10.1093/bioinformatics/btz238.
[37] Z. Aydin, D. Baker, and W. S. Noble, "Constructing Structural Profiles for Protein Torsion Angle Prediction:," in *Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms*, Lisbon, Portugal, 2015, pp. 26–35. doi: 10.5220/0005208500260035.
[38] S. F. Altschul *et al.*, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, no. 17, pp. 3389–3402, Sep. 1997, doi: 10.1093/nar/25.17.3389.
[39] M. Remmert, A. Biegert, A. Hauser, and J. Söding, "HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment," *Nat. Methods*, vol. 9, no. 2, Art. no. 2, Feb. 2012, doi: 10.1038/nmeth.1818.
[40] S. Kawashima, P. Pokarowski, M. Pokarowska, A. Kolinski, T. Katayama, and M. Kanehisa, "AAindex: amino acid index database, progress report 2008," *Nucleic Acids Res.*, vol. 36, no. suppl_1, pp. D202–D205, Jan. 2008, doi: 10.1093/nar/gkm998.
[41] D. R. Jones, "A Taxonomy of Global Optimization Methods Based on Response Surfaces," p. 39.
[42] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, and S.-H. Deng, "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimizationb," *J. Electron. Sci. Technol.*, vol. 17, no. 1, pp. 26–40, Mar. 2019, doi: 10.11989/JEST.1674-862X.80904120.
[43] "Protein structure prediction," *Wikipedia*. Mar. 16, 2022. Accessed: Jul. 07, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Protein_structure_prediction&oldid=1077551533

[44] "Four levels of Protein Structure," 2022. https://www.mun.ca/biology/scarr/iGen3_06-04.html (accessed Jul. 23, 2022).
[45] E. Kurt, "Proteinlerin Yapısı ve Özellikleri-2." 2022.
[46] E. Çakmak, "Derin öğrenme yöntemi ile protein ikincil yapı tahmini," 2021, Accessed: Jul. 14, 2022. [Online]. Available: https://acikerisim.sakarya.edu.tr/handle/20.500.12619/97173
[47] "Proteins Secondary Structure Predictions Structural Bioinformatics," 2022. https://slideplayer.com/slide/7398751/ (accessed Aug. 07, 2022).
[48] "Accessible surface area," *Wikipedia*. Jun. 25, 2022. Accessed: Jul. 20, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Accessible_surface_area&oldid=1094960442
[49] F. Mataeimoghadam *et al.*, "Enhancing protein backbone angle prediction by using simpler models of deep neural networks," *Sci. Rep.*, vol. 10, no. 1, Art. no. 1, Nov. 2020, doi: 10.1038/s41598-020-76317-6.
[50] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, 1983, doi: 10.1002/bip.360221211.
[51] Y. Wang, H. Mao, and Z. Yi, "Protein secondary structure prediction by using deep learning method," *Knowl.-Based Syst.*, vol. 118, pp. 115–123, Feb. 2017, doi: 10.1016/j.knosys.2016.11.015.
[52] L. Pauling, R. B. Corey, and H. R. Branson, "The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain," *Proc. Natl. Acad. Sci.*, vol. 37, no. 4, pp. 205–211, Apr. 1951, doi: 10.1073/pnas.37.4.205.
[53] N. Qian and T. J. Sejnowski, "Predicting the secondary structure of globular proteins using neural network models," *J. Mol. Biol.*, vol. 202, no. 4, pp. 865–884, Aug. 1988, doi: 10.1016/0022-2836(88)90564-5.
[54] L. H. Holley and M. Karplus, "Protein secondary structure prediction with a neural network.," *Proc. Natl. Acad. Sci.*, vol. 86, no. 1, pp. 152–156, Jan. 1989, doi: 10.1073/pnas.86.1.152.
[55] L. J. McGuffin, K. Bryson, and D. T. Jones, "The PSIPRED protein structure prediction server," *Bioinformatics*, vol. 16, no. 4, pp. 404–405, Apr. 2000, doi: 10.1093/bioinformatics/16.4.404.
[56] M. R. Uddin, S. Mahbub, M. S. Rahman, and M. S. Bayzid, "SAINT: self-attention augmented inception-inside-inception network improves protein secondary structure prediction," *Bioinformatics*, vol. 36, no. 17, pp. 4599–4608, Nov. 2020, doi: 10.1093/bioinformatics/btaa531.
[57] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *J. Mol. Biol.*, vol. 292, no. 2, pp. 195–202, Sep. 1999, doi: 10.1006/jmbi.1999.3091.
[58] S. Hua and Z. Sun, "A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach1 1Edited by B. Holland," *J. Mol. Biol.*, vol. 308, no. 2, pp. 397–407, Apr. 2001, doi: 10.1006/jmbi.2001.4580.
[59] 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," *J. Comput. Chem.*, vol. 33, no. 3, pp. 259–267, 2012, doi: 10.1002/jcc.21968.
[60] Y.-F. Huang and S.-Y. Chen, "Protein secondary structure prediction based on physicochemical features and PSSM by SVM," in *2013 IEEE Symposium on*

Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), Apr. 2013, pp. 9–15. doi: 10.1109/CIBCB.2013.6595382.

[61] C. N. Magnan and P. Baldi, "SSpro/ACCpro 5: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, machine learning and structural similarity," *Bioinformatics*, vol. 30, no. 18, pp. 2592–2597, Sep. 2014, doi: 10.1093/bioinformatics/btu352.

[62] A. Drozdetskiy, C. Cole, J. Procter, and G. J. Barton, "JPred4: a protein secondary structure prediction server," *Nucleic Acids Res.*, vol. 43, no. W1, pp. W389–W394, Jul. 2015, doi: 10.1093/nar/gkv332.

[63] Y. Wang, J. Cheng, Y. Liu, and Y. Chen, "Prediction of protein secondary structure using support vector machine with PSSM profiles," in *2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference*, May 2016, pp. 502–505. doi: 10.1109/ITNEC.2016.7560411.

[64] S. Wang, J. Peng, J. Ma, and J. Xu, "Protein Secondary Structure Prediction Using Deep Convolutional Neural Fields," *Sci. Rep.*, vol. 6, no. 1, Art. no. 1, Jan. 2016, doi: 10.1038/srep18962.

[65] R. Heffernan, Y. Yang, K. Paliwal, and Y. Zhou, "Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility," *Bioinformatics*, vol. 33, no. 18, pp. 2842–2849, Sep. 2017, doi: 10.1093/bioinformatics/btx218.

[66] Z. Aydin, O. Kaynar, Y. Görmez, and Y. E. Işik, "Comparison of machine learning classifiers for protein secondary structure prediction," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, May 2018, pp. 1–4. doi: 10.1109/SIU.2018.8404547.

[67] Z. Aydin, O. Kaynar, and Y. Görmez, "Dimensionality reduction for protein secondary structure and solvent accesibility prediction," *J. Bioinform. Comput. Biol.*, vol. 16, no. 05, p. 1850020, Oct. 2018, doi: 10.1142/S0219720018500208.

[68] M. Torrisi, G. Pollastri, and Q. Le, "Deep learning methods in protein structure prediction," *Comput. Struct. Biotechnol. J.*, vol. 18, pp. 1301–1310, Jan. 2020, doi: 10.1016/j.csbj.2019.12.011.

[69] M. Torrisi, M. Kaleel, and G. Pollastri, "Deeper Profiles and Cascaded Recurrent and Convolutional Neural Networks for state-of-the-art Protein Secondary Structure Prediction," *Sci. Rep.*, vol. 9, no. 1, Art. no. 1, Aug. 2019, doi: 10.1038/s41598-019-48786-x.

[70] M. S. Klausen *et al.*, "NetSurfP-2.0: Improved prediction of protein structural features by integrated deep learning," *Proteins Struct. Funct. Bioinforma.*, vol. 87, no. 6, pp. 520–527, 2019, doi: 10.1002/prot.25674.

[71] P. Kumar, S. Bankapur, and N. Patil, "An enhanced protein secondary structure prediction using deep learning framework on hybrid profile based features," *Appl. Soft Comput.*, vol. 86, p. 105926, Jan. 2020, doi: 10.1016/j.asoc.2019.105926.

[72] S. Zhou, H. Zou, C. Liu, M. Zang, and T. Liu, "Combining Deep Neural Networks for Protein Secondary Structure Prediction," *IEEE Access*, vol. 8, pp. 84362–84370, 2020, doi: 10.1109/ACCESS.2020.2992084.

[73] K. Kotowski, T. Smolarczyk, I. Roterman-Konieczna, and K. Stapor, "ProteinUnet—An efficient alternative to SPIDER3-single for sequence-based prediction of protein secondary structures," *J. Comput. Chem.*, vol. 42, no. 1, pp. 50–59, 2021, doi: 10.1002/jcc.26432.

[74] Z. Guo, J. Hou, and J. Cheng, “DNSS2: Improved ab initio protein secondary structure prediction using advanced deep learning architectures,” *Proteins Struct. Funct. Bioinforma.*, vol. 89, no. 2, pp. 207–217, 2021, doi: 10.1002/prot.26007.
[75] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, Art. no. 7873, Aug. 2021, doi: 10.1038/s41586-021-03819-2.
[76] Y. Zhao and Y. Liu, “OCLSTM: Optimized convolutional and long short-term memory neural network model for protein secondary structure prediction,” *PLOS ONE*, vol. 16, no. 2, p. e0245982, Feb. 2021, doi: 10.1371/journal.pone.0245982.
[77] W. Yang, Y. Liu, and C. Xiao, “Deep metric learning for accurate protein secondary structure prediction,” *Knowl.-Based Syst.*, vol. 242, p. 108356, Apr. 2022, doi: 10.1016/j.knosys.2022.108356.
[78] G. Xu, Q. Wang, and J. Ma, “OPUS-X: an open-source toolkit for protein torsion angles, secondary structure, solvent accessibility, contact map predictions and 3D folding,” *Bioinformatics*, vol. 38, no. 1, pp. 108–114, Jan. 2022, doi: 10.1093/bioinformatics/btab633.
[79] Y. Gao, Y. Zhao, Y. Ma, and Y. Liu, “Prediction of Protein Secondary Structure Based on WS-BiLSTM Model,” *Symmetry*, vol. 14, no. 1, Art. no. 1, Jan. 2022, doi: 10.3390/sym14010089.
[80] W. Yang, Z. Hu, L. Zhou, and Y. Jin, “Protein secondary structure prediction using a lightweight convolutional network and label distribution aware margin loss,” *Knowl.-Based Syst.*, vol. 237, p. 107771, Feb. 2022, doi: 10.1016/j.knosys.2021.107771.
[81] G. Urban, C. N. Magnan, and P. Baldi, “SSpro/ACCpro 6: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, deep learning and structural similarity,” *Bioinformatics*, vol. 38, no. 7, pp. 2064–2065, Apr. 2022, doi: 10.1093/bioinformatics/btac019.
[82] Y. Gormez and Z. Aydin, “IGPRED-MultiTask: A Deep Learning Model to Predict Protein Secondary Structure, Torsion Angles and Solvent Accessibility,” *IEEE/ACM Trans. Comput. Biol. Bioinform.*, pp. 1–12, 2022, doi: 10.1109/TCBB.2022.3191395.
[83] M. J. Thompson and R. A. Goldstein, “Predicting solvent accessibility: Higher accuracy using Bayesian statistics and optimized residue substitution classes,” *Proteins Struct. Funct. Bioinforma.*, vol. 25, no. 1, pp. 38–47, 1996, doi: 10.1002/(SICI)1097-0134(199605)25:1<38::AID-PROT4>3.0.CO;2-G.
[84] X. Li and X.-M. Pan, “New method for accurate prediction of solvent accessibility from protein sequence,” *Proteins Struct. Funct. Bioinforma.*, vol. 42, no. 1, pp. 1–5, 2001, doi: 10.1002/1097-0134(20010101)42:1<1::AID-PROT10>3.0.CO;2-N.
[85] H. Naderi-Manesh, M. Sadeghi, S. Arab, and A. A. Moosavi Movahedi, “Prediction of protein surface accessibility with information theory,” *Proteins Struct. Funct. Bioinforma.*, vol. 42, no. 4, pp. 452–459, 2001, doi: 10.1002/1097-0134(20010301)42:4<452::AID-PROT40>3.0.CO;2-Q.
[86] S. Ahmad and M. M. Gromiha, “NETASA: neural network based prediction of solvent accessibility,” *Bioinformatics*, vol. 18, no. 6, pp. 819–824, Jun. 2002, doi: 10.1093/bioinformatics/18.6.819.
[87] Z. Yuan, K. Burrage, and J. S. Mattick, “Prediction of protein solvent accessibility using support vector machines,” *Proteins Struct. Funct. Bioinforma.*, vol. 48, no. 3, pp. 566–570, 2002, doi: 10.1002/prot.10176.

[88] S. Ahmad and M. M. Gromiha, “Design and training of a neural network for predicting the solvent accessibility of proteins,” *J. Comput. Chem.*, vol. 24, no. 11, pp. 1313–1320, 2003, doi: 10.1002/jcc.10298.
[89] S. Ahmad, M. M. Gromiha, and A. Sarai, “Real value prediction of solvent accessibility from amino acid sequence,” *Proteins Struct. Funct. Bioinforma.*, vol. 50, no. 4, pp. 629–635, 2003, doi: 10.1002/prot.10328.
[90] H. Kim and H. Park, “Prediction of protein relative solvent accessibility with support vector machines and long-range interaction 3D local descriptor,” *Proteins Struct. Funct. Bioinforma.*, vol. 54, no. 3, pp. 557–562, 2004, doi: 10.1002/prot.10602.
[91] M. N. Nguyen and J. C. Rajapakse, “Prediction of protein relative solvent accessibility with a two-stage SVM approach,” *Proteins Struct. Funct. Bioinforma.*, vol. 59, no. 1, pp. 30–37, 2005, doi: 10.1002/prot.20404.
[92] J. Sim, S.-Y. Kim, and J. Lee, “Prediction of protein solvent accessibility using fuzzy k-nearest neighbor method,” *Bioinformatics*, vol. 21, no. 12, pp. 2844–2849, Jun. 2005, doi: 10.1093/bioinformatics/bti423.
[93] L. Deng, C. Fan, and Z. Zeng, “A sparse autoencoder-based deep neural network for protein solvent accessibility and contact number prediction,” *BMC Bioinformatics*, vol. 18, no. 16, p. 569, Dec. 2017, doi: 10.1186/s12859-017-1971-7.
[94] O. Keskin, D. Yuret, A. Gursoy, M. Turkay, and B. Erman, “Relationships between amino acid sequence and backbone torsion angle preferences,” *Proteins Struct. Funct. Bioinforma.*, vol. 55, no. 4, pp. 992–998, 2004, doi: 10.1002/prot.20100.
[95] S. Wu and Y. Zhang, “ANGLOR: A Composite Machine-Learning Algorithm for Protein Backbone Torsion Angle Prediction,” *PLOS ONE*, vol. 3, no. 10, p. e3400, Oct. 2008, doi: 10.1371/journal.pone.0003400.
[96] B. Xue, O. Dor, E. Faraggi, and Y. Zhou, “Real-value prediction of backbone torsion angles,” *Proteins Struct. Funct. Bioinforma.*, vol. 72, no. 1, pp. 427–433, 2008, doi: 10.1002/prot.21940.
[97] M.-S. Cheung, M. L. Maguire, T. J. Stevens, and R. W. Broadhurst, “DANGLE: A Bayesian inferential method for predicting protein backbone dihedral angles and secondary structure,” *J. Magn. Reson.*, vol. 202, no. 2, pp. 223–233, Feb. 2010, doi: 10.1016/j.jmr.2009.11.008.
[98] R. Heffernan *et al.*, “Improving prediction of secondary structure, local backbone angles and solvent accessible surface area of proteins by iterative deep learning,” *Sci. Rep.*, vol. 5, no. 1, Art. no. 1, Jun. 2015, doi: 10.1038/srep11476.
[99] H. Li, J. Hou, B. Adhikari, Q. Lyu, and J. Cheng, “Deep learning methods for protein torsion angle prediction,” *BMC Bioinformatics*, vol. 18, no. 1, p. 417, Sep. 2017, doi: 10.1186/s12859-017-1834-2.
[100] J. Gao, Y. Yang, and Y. Zhou, “Grid-based prediction of torsion angle probabilities of protein backbone and its application to discrimination of protein intrinsic disorder regions and selection of model structures,” *BMC Bioinformatics*, vol. 19, no. 1, p. 29, Feb. 2018, doi: 10.1186/s12859-018-2031-7.
[101] Y. Gao, S. Wang, M. Deng, and J. Xu, “RaptorX-Angle: real-value prediction of protein backbone dihedral angles through a hybrid method of clustering and deep learning,” *BMC Bioinformatics*, vol. 19, no. 4, p. 100, May 2018, doi: 10.1186/s12859-018-2065-x.
[102] G. Xu, Q. Wang, and J. Ma, “OPUS-Refine: A Fast Sampling-Based Framework for Refining Protein Backbone Torsion Angles and Global Conformation,” *J. Chem. Theory Comput.*, vol. 16, no. 2, pp. 1359–1366, Feb. 2020, doi: 10.1021/acs.jctc.9b01054.

[103] M. A. H. Newton, F. Mataeimoghadam, R. Zaman, and A. Sattar, “Secondary structure specific simpler prediction models for protein backbone angles,” *BMC Bioinformatics*, vol. 23, no. 1, p. 6, Jan. 2022, doi: 10.1186/s12859-021-04525-6.
[104] J. Cheng and P. Baldi, “Improved residue contact prediction using support vector machines and a large feature set,” *BMC Bioinformatics*, vol. 8, no. 1, p. 113, Apr. 2007, doi: 10.1186/1471-2105-8-113.
[105] W. H. Chan and M. S. Mohamad, “Prediction of Protein Residue Contact Using Support Vector Machine,” in *Knowledge Technology*, Berlin, Heidelberg, 2012, pp. 323–332. doi: 10.1007/978-3-642-32826-8_33.
[106] P. Fariselli, O. Olmea, A. Valencia, and R. Casadio, “Prediction of contact maps with neural networks and correlated mutations,” *Protein Eng. Des. Sel.*, vol. 14, no. 11, pp. 835–843, Nov. 2001, doi: 10.1093/protein/14.11.835.
[107] G. Shackelford and K. Karplus, “Contact prediction using mutual information and neural nets,” *Proteins Struct. Funct. Bioinforma.*, vol. 69, no. S8, pp. 159–164, 2007, doi: 10.1002/prot.21791.
[108] P. Di Lena, K. Nagata, and P. Baldi, “Deep architectures for protein contact map prediction,” *Bioinformatics*, vol. 28, no. 19, pp. 2449–2457, Oct. 2012, doi: 10.1093/bioinformatics/bts475.
[109] W. Ding, J. Xie, D. Dai, H. Zhang, H. Xie, and W. Zhang, “CNNcon: Improved Protein Contact Maps Prediction Using Cascaded Neural Networks,” *PLOS ONE*, vol. 8, no. 4, p. e61533, Apr. 2013, doi: 10.1371/journal.pone.0061533.
[110] P. Lena, K. Nagata, and P. Baldi, “Deep Spatio-Temporal Architectures and Learning for Protein Structure Prediction,” in *Advances in Neural Information Processing Systems*, 2012, vol. 25. Accessed: Jul. 24, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2012/hash/8c19f571e251e61cb8dd3612f26d5ecf-Abstract.html
[111] A. N. Tegge, Z. Wang, J. Eickholt, and J. Cheng, “NNcon: improved protein contact map prediction using 2D-recursive neural networks,” *Nucleic Acids Res.*, vol. 37, no. suppl_2, pp. W515–W518, Jul. 2009, doi: 10.1093/nar/gkp305.
[112] R. M. MacCallum, “Striped sheets and protein contact prediction,” *Bioinformatics*, vol. 20, no. suppl_1, pp. i224–i231, Aug. 2004, doi: 10.1093/bioinformatics/bth913.
[113] J. Bacardit, P. Widera, A. Márquez-Chamorro, F. Divina, J. S. Aguilar-Ruiz, and N. Krasnogor, “Contact map prediction using a large-scale ensemble of rule sets and the fusion of multiple predicted structural features,” *Bioinformatics*, vol. 28, no. 19, pp. 2441–2448, Oct. 2012, doi: 10.1093/bioinformatics/bts472.
[114] J. Hanson, K. Paliwal, T. Litfin, Y. Yang, and Y. Zhou, “Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks,” *Bioinformatics*, vol. 34, no. 23, pp. 4039–4045, Dec. 2018, doi: 10.1093/bioinformatics/bty481.
[115] J. Hanson, “SPOT-Contact: Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks • Jack Hanson,” *Sparks Lab*, Feb. 10, 2020. http://sparks-lab.org/server/spot-contact/ (accessed May 28, 2022).
[116] G. Wang and R. L. Dunbrack Jr, “PISCES: a protein sequence culling server,” *Bioinformatics*, vol. 19, no. 12, pp. 1589–1591, Aug. 2003, doi: 10.1093/bioinformatics/btg224.
[117] I. Y. Y. Koh *et al.*, “EVA: evaluation of protein structure prediction servers,” *Nucleic Acids Res.*, vol. 31, no. 13, pp. 3311–3315, Jul. 2003, doi: 10.1093/nar/gkg619.

[118] "CASP - Home," 2016. https://predictioncenter.org/casp12/index.cgi (accessed Jul. 23, 2022).
[119] thuxugang, "OPUS-TASS." Apr. 07, 2022. Accessed: Jul. 26, 2022. [Online]. Available: https://github.com/thuxugang/opus_tass
[120] "Nucleotide BLAST: Align two or more sequences using BLAST." https://blast.ncbi.nlm.nih.gov/Blast.cgi?BLAST_SPEC=blast2seq&LINK_LOC=align2seq&PAGE_TYPE=BlastSearch (accessed Jul. 26, 2022).
[121] M. Steinegger and J. Söding, "MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets," *Nat. Biotechnol.*, vol. 35, no. 11, pp. 1026–1028, Nov. 2017, doi: 10.1038/nbt.3988.
[122] R. Bhaskaran and P. K. Ponnuswamy, "Positional flexibilities of amino acid residues in globular proteins," *Int. J. Pept. Protein Res.*, vol. 32, no. 4, pp. 241–255, 1988, doi: https://doi.org/10.1111/j.1399-3011.1988.tb01258.x.
[123] C. C. Bigelow, "On the average hydrophobicity of proteins and the relation between it and protein structure," *J. Theor. Biol.*, vol. 16, no. 2, pp. 187–211, Aug. 1967, doi: 10.1016/0022-5193(67)90004-5.
[124] J. Pontius, J. Richelle, and S. J. Wodak, "Deviations from Standard Atomic Volumes as a Quality Measure for Protein Crystal Structures," *J. Mol. Biol.*, vol. 264, no. 1, pp. 121–136, Nov. 1996, doi: 10.1006/jmbi.1996.0628.
[125] M. Charton and B. I. Charton, "The structural dependence of amino acid hydrophobicity parameters," *J. Theor. Biol.*, vol. 99, no. 4, pp. 629–644, Dec. 1982, doi: 10.1016/0022-5193(82)90191-6.
[126] H. Cid, M. Bunster, M. Canales, and F. Gazitúa, "Hydrophobicity and structural classes in proteins," *Protein Eng. Des. Sel.*, vol. 5, no. 5, pp. 373–375, Jul. 1992, doi: 10.1093/protein/5.5.373.
[127] U. Bastolla, M. Porto, H. E. Roman, and M. Vendruscolo, "Principal eigenvector of contact matrices and hydrophobicity profiles in proteins," *Proteins Struct. Funct. Bioinforma.*, vol. 58, no. 1, pp. 22–30, 2005, doi: https://doi.org/10.1002/prot.20240.
[128] H. Zhou and Y. Zhou, "Quantifying the effect of burial of amino acid residues on protein stability," *Proteins Struct. Funct. Bioinforma.*, vol. 54, no. 2, pp. 315–322, 2004, doi: https://doi.org/10.1002/prot.10584.
[129] R. V. Wolfenden, P. M. Cullis, and C. C. Southgate, "Water, protein folding, and the genetic code," *Science*, vol. 206, no. 4418, pp. 575–577, Nov. 1979, doi: 10.1126/science.493962.
[130] A. Kidera, Y. Konishi, M. Oka, T. Ooi, and H. A. Scheraga, "Statistical analysis of the physical properties of the 20 naturally occurring amino acids," *J. Protein Chem.*, vol. 4, no. 1, pp. 23–55, Feb. 1985, doi: 10.1007/BF01025492.
[131] G. D. Fasman, *Prediction of Protein Structure and the Principles of Protein Conformation*. Springer Science & Business Media, 2012.
[132] W. R. Krigbaum and B. H. Rubin, "Local interactions as a structure determinant for globular proteins," *Biochim. Biophys. Acta BBA - Protein Struct.*, vol. 229, no. 2, pp. 368–383, Feb. 1971, doi: 10.1016/0005-2795(71)90196-6.
[133] M. F. Perutz, J. V. Kilmartin, K. Nagai, A. Szabo, and S. R. Simon, "Influence of globin structures on the state of the heme. IV. Ferrous low spin derivatives," *Biochemistry*, vol. 15, no. 2, pp. 378–387, Jan. 1976, doi: 10.1021/bi00647a022.
[134] B. Robson and D. J. Osguthorpe, "Refined models for computer simulation of protein folding: Applications to the study of conserved secondary structure and flexible hinge points during the folding of pancreatic trypsin inhibitor," *J. Mol. Biol.*, vol. 132, no. 1, pp. 19–51, Jul. 1979, doi: 10.1016/0022-2836(79)90494-7.

[135] A. W. Lee *et al.*, *Comparing the Polarities of the Amino Acids: Side-Chain Distribution Coefficients between the Vapor Phase, Cyclohexane, 1 -0ctano1, and Neutral Aqueous Solutiont*.
[136] M. A. Roseman, "Hydrophilicity of polar amino acid side-chains is markedly reduced by flanking peptide bonds," *J. Mol. Biol.*, vol. 200, no. 3, pp. 513–522, Apr. 1988, doi: 10.1016/0022-2836(88)90540-2.
[137] V. Veljkovic, I. Cosic, Dimitrijevic, and D. Lalovic, "Is it Possible to Analyze DNA and Protein Sequences by the Methods of Digital Signal Processing?," *IEEE Trans. Biomed. Eng.*, vol. BME-32, no. 5, pp. 337–341, May 1985, doi: 10.1109/TBME.1985.325549.
[138] P. K. Warme and R. S. Morgan, "A survey of amino acid side-chain interactions in 21 proteins," *J. Mol. Biol.*, vol. 118, no. 3, pp. 289–304, Jan. 1978, doi: 10.1016/0022-2836(78)90229-2.
[139] R. Wolfenden, L. Andersson, P. M. Cullis, and C. C. B. Southgate, "Affinities of amino acid side chains for solvent water," *Biochemistry*, vol. 20, no. 4, pp. 849–855, Feb. 1981, doi: 10.1021/bi00507a030.
[140] J. Kjær, L. Høj, Z. Fox, and J. D. Lundgren, "Prediction of phenotypic susceptibility to antiretroviral drugs using physiochemical properties of the primary enzymatic structure combined with artificial neural networks," *HIV Med.*, vol. 9, no. 8, pp. 642–652, 2008, doi: https://doi.org/10.1111/j.1468-1293.2008.00612.x.
[141] J. M. Zimmerman, N. Eliezer, and R. Simha, "The characterization of amino acid sequences in proteins by statistical methods," *J. Theor. Biol.*, vol. 21, no. 2, pp. 170–201, Nov. 1968, doi: 10.1016/0022-5193(68)90069-6.
[142] R. Grantham, "Amino Acid Difference Formula to Help Explain Protein Evolution," *Science*, vol. 185, no. 4154, pp. 862–864, Sep. 1974, doi: 10.1126/science.185.4154.862.
[143] K. Takano and K. Yutani, "A new scale for side-chain contribution to protein stability based on the empirical stability analysis of mutant proteins," *Protein Eng. Des. Sel.*, vol. 14, no. 8, pp. 525–528, Aug. 2001, doi: 10.1093/protein/14.8.525.
[144] H. Meirovitch, S. Rackovsky, and H. A. Scheraga, "Empirical Studies of Hydrophobicity. 1. Effect of Protein Size on the Hydrophobic Behavior of Amino Acids," *Macromolecules*, vol. 13, no. 6, pp. 1398–1405, Nov. 1980, doi: 10.1021/ma60078a013.
[145] J. A. Stekol, *Amino Acids and Serum Proteins*, vol. 44. AMERICAN CHEMICAL SOCIETY, 1964. doi: 10.1021/ba-1964-0044.
[146] L. Acid, D. Citrulline, and D. Hci, "Heat capacities absolute entropies and entropies of formation of amino acids and related compounds," in *Handbook of biochemistry and molecular biology*, 1984.
[147] J.-L. Fauchère, M. Charton, L. B. Kier, A. Verloop, and V. Pliska, "Amino acid side chain parameters for correlation studies in biology and pharmacology," *Int. J. Pept. Protein Res.*, vol. 32, no. 4, pp. 269–278, 1988, doi: https://doi.org/10.1111/j.1399-3011.1988.tb01261.x.
[148] G. D. Fasman, *Handbook of Biochemistry: Section D Physical Chemical Data, Volume I*. CRC Press, 2018.
[149] J. Gu, T. Zhang, C. Wu, Y. Liang, and X. Shi, "Refined Contact Map Prediction of Peptides Based on GCN and ResNet," *Front. Genet.*, vol. 13, 2022, Accessed: Jul. 28, 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fgene.2022.859626
[150] S. Siami-Namini, N. Tavakoli, and A. Siami Namin, "A Comparison of ARIMA and LSTM in Forecasting Time Series," in *2018 17th IEEE International*

Conference on Machine Learning and Applications (ICMLA), Dec. 2018, pp. 1394–1401. doi: 10.1109/ICMLA.2018.00227.
[151] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater Image Processing and Object Detection Based on Deep CNN Method," *J. Sens.*, vol. 2020, p. e6707328, May 2020, doi: 10.1155/2020/6707328.
[152] P. Arena, A. Basile, M. Bucolo, and L. Fortuna, "Image processing for medical diagnosis using CNN," *Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip.*, vol. 497, no. 1, pp. 174–178, Jan. 2003, doi: 10.1016/S0168-9002(02)01908-3.
[153] A. Zhao and Y. Yu, "Knowledge-enabled BERT for aspect-based sentiment analysis," *Knowl.-Based Syst.*, vol. 227, p. 107220, Sep. 2021, doi: 10.1016/j.knosys.2021.107220.
[154] C. Pelletier, G. I. Webb, and F. Petitjean, "Temporal Convolutional Neural Network for the Classification of Satellite Image Time Series," *Remote Sens.*, vol. 11, no. 5, Art. no. 5, Jan. 2019, doi: 10.3390/rs11050523.
[155] M. V. Valueva, N. N. Nagornov, P. A. Lyakhov, G. V. Valuev, and N. I. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Math. Comput. Simul.*, vol. 177, pp. 232–243, Nov. 2020, doi: 10.1016/j.matcom.2020.04.031.
[156] M. Z. Alom *et al.*, "A State-of-the-Art Survey on Deep Learning Theory and Architectures," *Electronics*, vol. 8, no. 3, Art. no. 3, Mar. 2019, doi: 10.3390/electronics8030292.
[157] L. Jian, H. Xiang, and G. Le, "LSTM-Based Attentional Embedding for English Machine Translation," *Sci. Program.*, vol. 2022, p. e3909726, Mar. 2022, doi: 10.1155/2022/3909726.
[158] S. Bhaskar and Thasleema T. M., "LSTM model for visual speech recognition through facial expressions," *Multimed. Tools Appl.*, Apr. 2022, doi: 10.1007/s11042-022-12796-1.
[159] N. Fatima, A. S. Imran, Z. Kastrati, S. M. Daudpota, and A. Soomro, "A Systematic Literature Review on Text Generation Using Deep Neural Network Models," *IEEE Access*, vol. 10, pp. 53490–53503, 2022, doi: 10.1109/ACCESS.2022.3174108.
[160] H. Abbasimehr and R. Paki, "Improving time series forecasting using LSTM and attention models," *J. Ambient Intell. Humaniz. Comput.*, vol. 13, no. 1, pp. 673–691, Jan. 2022, doi: 10.1007/s12652-020-02761-x.
[161] "Recurrent neural network," *Wikipedia*. Sep. 08, 2022. Accessed: Sep. 19, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1109264340
[162] I. K. Ihianle, A. O. Nwajana, S. H. Ebenuwa, R. I. Otuka, K. Owa, and M. O. Orisatoki, "A Deep Learning Approach for Human Activities Recognition From Multimodal Sensing Devices," *IEEE Access*, vol. 8, pp. 179028–179038, 2020, doi: 10.1109/ACCESS.2020.3027979.
[163] W. S. Cleveland, "Graphs in Scientific Publications," *Am. Stat.*, vol. 38, no. 4, pp. 261–269, Nov. 1984, doi: 10.1080/00031305.1984.10483223.
[164] "How powerful are Graph Convolutional Networks?" http://tkipf.github.io/graph-convolutional-networks/ (accessed Aug. 05, 2022).
[165] M. Muralikrishnan and R. Anitha, "Comparison of Breast Cancer Multi-class Classification Accuracy Based on Inception and InceptionResNet Architecture," in *Emerging Trends in Computing and Expert Technology*, Cham, 2020, pp. 1155–1162. doi: 10.1007/978-3-030-32150-5_118.

[166] E. Y. Walker *et al.*, "Inception loops discover what excites neurons most using deep predictive models," *Nat. Neurosci.*, vol. 22, no. 12, Art. no. 12, Dec. 2019, doi: 10.1038/s41593-019-0517-x.
[167] J. Wang *et al.*, "Deep learning for quality assessment of retinal OCT images," *Biomed. Opt. Express*, vol. 10, no. 12, pp. 6057–6072, Dec. 2019, doi: 10.1364/BOE.10.006057.
[168] "Keras Deep Learning on Graphs," *Keras Deep Learning on Graphs*, 2020. https://vermamachinelearning.github.io/keras-deep-graph-learning/
[169] "Keras: the Python deep learning API," May 22, 2022. https://keras.io/ (accessed May 25, 2022).
[170] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Cham, 2015, pp. 234–241. doi: 10.1007/978-3-319-24574-4_28.
[171] "tf.keras.initializers.GlorotNormal | TensorFlow Core v2.9.1," *TensorFlow*, May 26, 2022. https://www.tensorflow.org/api_docs/python/tf/keras/initializers/GlorotNormal (accessed May 26, 2022).
[172] "skopt module," *skopt module*, 2019. https://scikit-optimize.github.io/
[173] A. Zemla, Č. Venclovas, K. Fidelis, and B. Rost, "A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment," *Proteins Struct. Funct. Bioinforma.*, vol. 34, no. 2, pp. 220–223, 1999, doi: https://doi.org/10.1002/(SICI)1097-0134(19990201)34:2<220::AID-PROT7>3.0.CO;2-K.
[174] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochim. Biophys. Acta BBA - Protein Struct.*, vol. 405, no. 2, pp. 442–451, Oct. 1975, doi: 10.1016/0005-2795(75)90109-9.
[175] "Z Score Calculator for 2 Population Proportions," Oct. 02, 2018. https://www.socscistatistics.com/tests/ztest/Default2.aspx
[176] K. Team, "Keras documentation: Callbacks API," Jun. 02, 2022. https://keras.io/api/callbacks/ (accessed Jun. 02, 2022).
[177] Y. Zhang, "Protein Structure Prediction: Is It Useful?," *Curr. Opin. Struct. Biol.*, vol. 19, no. 2, pp. 145–155, Apr. 2009, doi: 10.1016/j.sbi.2009.02.005.
[178] R. P. D. Bank, "PDB Statistics: Overall Growth of Released Structures Per Year." https://www.rcsb.org/stats/growth/growth-released-structures (accessed Aug. 10, 2022).
[179] K. Maxfield, "Why Structure Prediction Matters," *DNASTAR*, Jul. 21, 2020. https://www.dnastar.com/blog/structural-biology/why-structure-prediction-matters/ (accessed Aug. 10, 2022).
[180] S. Asseng *et al.*, "Climate change impact and adaptation for wheat protein," *Glob. Change Biol.*, vol. 25, no. 1, pp. 155–173, Jan. 2019, doi: 10.1111/gcb.14481.

CURRICULUM VITAE

2010 – 2015	B.Sc., Computer Engineering, Melikşah University, Kayseri, TURKEY
2015 – 2017	M.Sc., Electrical and Computer Engineering, Abdullah Gül University, Kayseri, TURKEY
2018 – ------	Doctoral Candidate, Electrical and Computer Engineering, Abdullah Gül University, Kayseri, TURKEY

SELECTED PUBLICATIONS AND PRESENTATIONS

J1) Z. Aydın, O. Kaynar, Y. Görmez, Dimensionality reduction for protein secondary structure and solvent accesibility prediction published in Journal of Bioinformatics and Computational Biology (Oct. 2018).

J2) Y. Görmez, Z. Aydın, R. Karademir, V. C. Güngör, A deep learning approach with Bayesian optimization and ensemble classifiers for detecting denial of service attacks published in International Journal of Communication Systems (May 2020).

J3) Y. Görmez, M. Sabzekar, Z. Aydın, IGPRED: Combination of convolutional neural and graph convolutional networks for protein secondary structure prediction published in Proteins: Structure, Function, and Bioinformatics (Oct. 2021).

J4) Y. Görmez, Z. Aydın, IGPRED-MultiTask: A Deep Learning Model to Predict Protein Secondary Structure, Torsion Angles and Solvent Accessibility published in IEEE/ACM Transactions on Computational Biology and Bioinformatics (July 2022).

C1) Y. Görmez, Z. Aydın, ROSE: A Novel Approach for Protein Secondary Structure Prediction in The International Conference on Artificial Intelligence and Applied Mathematics in Engineering (April 2020).

C2) Y. Görmez, Z. Aydın, Protein Secondary Strcuture Predicition using deep learning model based on graph convolutional network in 14th International Conference on Engineering and Natural Sciences (July 2022)