Ömmu Gülsüm UZUT OPTIMIZING CLASSIFIERS FOR PROTEIN SECONDARY STRUCTURE PREDICTION AGU 2017

OPTIMIZING CLASSIFIERS FOR PROTEIN SECONDARY 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
MASTER'S

By
Ömmu Gülsüm UZUT
July 2017

OPTIMIZING CLASSIFIERS FOR PROTEIN SECONDARY STRUCTURE PREDICTION

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF NATURAL SCIENCES OF
ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER'S

By
Ömm Gülsüm UZUT
July 2017

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: Ömmu Gülsüm UZUT
Signature :

REGULATORY COMPLIANCE

M. Sc. thesis titled Optimizing Classifiers for Protein Secondary Structure Prediction has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By

Ömmu Gülsüm UZUT

Advisor

Assis. Prof. Zafer AYDIN

Head of the Electrical and Computer Engineering Program
Assoc. Prof. Vehbi Çağrı GÜNGÖR

ACCEPTANCE AND APPROVAL

M. Sc. thesis titled Optimizing Classifiers for Protein Secondary Structure Prediction and prepared by Ömmu Gülsüm UZUT has been accepted by the jury in the Electrical and Computer Engineering Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

…….. /…….. / ……..

(Thesis Defense Exam Date)

JURY:

Advisor : Assist. Prof. Zafer AYDIN

Member : Prof. Bülent YILMAZ

Member : Assist. Prof. Ufuk NALBANTOĞLU

APPROVAL:

The acceptance of this M. Sc. 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

OPTIMIZING CLASSIFIERS FOR PROTEIN SECONDARY STRUCTURE PREDICTION

Ömmu Gülsüm UZUT
Master's program in Electrical and Computer Engineering Department
Supervisor: Assist. Prof. Zafer AYDIN

July 2017

Protein secondary structure prediction is important for understanding protein structure and function. PSSP can be seen as a bridge between amino acid sequence and 3D structure of a protein. Many methods have been performed to improve prediction accuracy rate and get good achievement. There are multiple situations that will affect the performance of a method. One of these situations is selection of correct parameter. Hyperparameters are parameters that cannot be directly learned from the regular training process. Although the methods have default hyperparameter values, it is possible to improve performance of methods by using those hyperparameters with different values which can be more convenient. Parameter optimization plays an important role at this stage. It applies to methods to find best hyperparameter values to apply methods.

In our thesis, computational methods such as Random forest, Support vector machines and deep convolutional neural fields have been used and optimized on CB513 dataset. We have aimed to optimize methods with different values to improve the results and show the importance of parameter optimization in protein structure prediction. We also tried to use some ensemble methods to compare our results with individual classifiers to see the improvement of results.

Keywords: Bioinformatics, Optimizing Models, Ensemble Methods, Protein Structure Prediction.

ÖZET

PROTEİN İKİNCİL YAPISININ TAHMİNİ İÇİN SINIFLANDIRMA YÖNTEMLERİNİN OPTİMİZASYONU

Ömmu Gülsüm UZUT
Elektrik ve Bilgisayar Mühendisliği Ana Bilim Dalı, Yüksek Lisans Programı
Tez Yöneticisi: Yrd. Doç. Dr. Zafer AYDIN
Temmuz 2017

Protein ikincil yapı tahmini, proteinin yapısını ve fonsiyonunu anlamak için önemlidir. Protein ikincil yapı tahmini, bir protein dizisiyle o proteinin 3 boyutlu yapısı arasında bir köprü olarak görülebilir. Şimdiye kadar, tahmin doğruluğu oranını artırmak ve iyi bir başarı elde etmek için birçok yöntem gerçekleştirildi. Bir yöntemin performansını etkileyecek birden fazla durum vardır. Bu durumlardan biri doğru parametrenin seçilmesidir. Hiperparametreler, işlem sürecinde doğrudan öğrenilemeyen parametrelerdir. Yöntemlerin varsayılan hiperparametre değerleri olmasına rağmen, daha uygun olabilecek farklı değerlere sahip hiperparametreler kullanılarak yöntemlerin performansını artırmak mümkündür. Parametre optimizasyonu bu aşamada önemli bir rol oynamaktadır. Yöntemlerde kullanılacak en iyi hiperparametre değerlerini bulmak için metotlarda uygulanmaktadır.

Tezimizde, sonuçları iyileştirmek ve protein yapı tahmini için parametre optimizasyonunun önemini göstermek amacıyla metotları optimize ettik. Sonuçların iyileştirilmesini görmek için sonuçlarımızı bireysel sınıflandırıcılarla karşılaştırmak için bazı topluluk yöntemleri de kullanmaya çalıştık.

Anahtar Kelimeler: Biyoinformatik, Model Optimizasyonu, Topluluk Yöntemleri, Protein Yapı Tahmini.

Acknowledgements

I would like to express my deepest regards to my adviser Assistant Professor Zafer AYDIN, for his interest and invaluable helpfulness. I am so grateful to study with him.

I would like to thank my family especially my father Zülküf UZUT and my mother Nisbet UZUT for being in favor of me in any case and believing in me.

All calculations in this thesis is made thanks to TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA Resources).

Table of Contents

1. INTRODUCTION 1

2. STRUCTURE OF A PROTEIN 4

2.1 PROTEIN STRUCTURE LEVELS 5

2.1.1 Primary Structure 5

2.1.2 Secondary Structure 5

2.1.3 Tertiary Structure 6

2.1.4 Quaternary Structure 7

2.2 TORSION ANGLE 7

2.3 SOLVENT ACCESSIBILITY 7

3. PROTEIN STRUCTURE PREDICTION 8

3.1 IMPORTANCE OF SECONDARY STRUCTURE PREDICTION 8

3.1.1 Secondary Structure Types 9

3.2 MEASURES FOR PREDICTION ACCURACY 10

4. METHODS 12

4.1 DSPRED METHOD 12

4.2 FEATURE EXTRACTION 16

4.2.1 PSSM 17

4.2.2 Position Specific Iterative BLAST (PSI-BLAST) 17

4.2.3 HMM Profile Matrices 17

4.2.4 Structural Profiles 17

4.2.5 Datasets 20

4.3 SUPPORT VECTOR MACHINES 21

4.3.1 Linear Separation 21

4.3.2 Non-linear Separation 23

4.3.3 MULTICLASS SVM 23

4.4 RANDOMFOREST 24

4.5 DEEP CONVOLUTİONAL NEURAL FIELDS 25

4.5.1 Deep Convolutional Neural Networks 26

4.5.2 Conditional Neural Fields 26

4.6 PARAMETER OPTIMIZATION 27
4.6.1 Parameter Optimization for Support Vector Machines 28
4.6.2 Parameter Optimization for Random Forest 29
4.6.3 Parameter Optimization for Deep Convolutional Neural Fields 29
5. ENSEMBLE METHODS 31
5.1 COMBINING HIDDEN LAYERS FOR DEEP CNF 31
5.2 MODEL AVERAGING 31
5.2.1 SVM+Randomforest Model Averaging 32
5.2.2 SVM+DeepCNF Model Averaging 32
5.2.3 Randomforest+DeepCNF Model Averaging 33
5.2.4 SVM+Randomforest+DeepCNF Model Averaging 33
6. RESULTS 34
6.1 7-FOLD CROSS VALIDATION EXPERIMENT WITH CB513 DATASET 34
6.2 OPTIMIZATION RESULTS 34
6.2.1 SVM Results 35
6.2.2 Random Forest Results 36
6.2.3 Deep CNF Results 37
6.3 TRAIN-TEST RESULTS 43
6.4 ENSEMBLE METHODS RESULTS 46
6.4.1 Deep CNF Hidden Layer Combination 47
6.4.2 Model Averaging Results on Validation and Test Sets 48
7. CONCLUSION 69
8. BIBLIOGRAPHY 71

List of Figures

Figure 2.1 Structure of an amino acid 4
Figure 2.1.1.1 Primary structure of an amino acid 5
Figure 2.1.2.1 Secondary structure of an amino acid 6
Figure 2.2.1 The phi (φ), psi (ψ) and omega (ω) angles in the protein backbone 7
Figure 2.3.1 Van der Waals area and solvent accessible area. 7
Figure 3.1.1 Three state of a protein secondary structure prediction. 9
Figure 4.1.1 Estimation of the 3-state secondary structure with the hybrid model using dynamic bayes networks and classifier together 13
Figure 4.1.2 A dynamic bayesian network for protein secondary structure prediction 15
Figure 4.2.4.1 A structural profiles for 3 state of secondary structure representation 19
Figure 4.2.4.2 An example for obtaining feature vectors 20
Figure 4.3.1.1 Linear SVM classifier 21
Figure 4.3.2.1 The non-linear SVM 23
Figure 4.3.3.1 Multiclass SVM 24
Figure 4.5.2.1 Design of DeepCNF 27
Figure 4.6.1.1 The effect of C constant on the decision boundary 28
Figure 4.6.1.2 The effect of kernel parameter on decision boundary 28

List of Tables

Table 3.1.1.1.1.1 8 class secondary structure prediction..... 9
Table 6.2.1.1 $Q3$ accuracy of the SVM method for parameter optimization on CB513 validation dataset..... 35
Table 6.2.1.2 $Q3, QH, QE, QL$ accuracies of the SVM method for training on CB513 validation datasets. 36
Table 6.2.2.1 $Q3$ accuracy of the Random forest method for parameter optimization on CB513 validation dataset..... 36
Table 6.2.2.2 $Q3, QH, QE, QL$ accuracies of the Random Forest method for training on CB513 validation datasets. 37
Table 6.2.3.1.1 $Q3$ accuracy of the Deep CNF method with 3 hidden layers for parameter optimization on CB513 validation datasets..... 38
Table 6.2.3.1.2 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 3 hidden layers for training on CB513 validation datasets..... 39
Table 6.2.3.2.1 $Q3$ accuracies of the Deep CNF method with 4 hidden layers for parameter optimization on CB513 validation datasets..... 40
Table 6.2.3.2.2 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 4 hidden layers for training on CB513 validation datasets..... 40
Table 6.2.3.3.1 $Q3$ accuracies of the Deep CNF method with 5 hidden layers for parameter optimization on CB513 validation datasets..... 41
Table 6.2.3.3.2 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 5 hidden layers for training on CB513 validation datasets..... 42
Table 6.3.1 $Q3, QH, QE, QL$ accuracies of the SVM method for training on CB513 test datasets. 43
Table 6.3.2 $Q3, QH, QE, QL$ accuracies of the Random forest method for training on CB513 test datasets. 44
Table 6.3.3 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 3 hidden layers for training on CB513 test dataset..... 45
Table 6.3.4 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 4 hidden layers for training on CB513 test dataset..... 45
Table 6.3.5 $Q3, QH, QE, QL$ accuracies of the Deep CNF method with 5 hidden layers for training on CB513 validation and test dataset..... 46
Table 6.4.1.1 $Q3, QH, QE, QL$ accuracies of the Deep CNF hidden layer ensemble method for train validation on CB513 dataset. 47
Table 6.4.1.2 $Q3, QH, QE, QL$ accuracies of the Deep CNF hidden layer ensemble method for train test on CB513 dataset..... 48
Table 6.4.2.1.1 $Q3, QH, QE, QL$ accuracies of the SVM and Random forest model averaging ensemble method for validation on CB513 dataset..... 49
Table 6.4.2.1.2 $Q3, QH, QE, QL$ accuracies of the SVM and Random forest model averaging ensemble method for test on CB513 dataset. 49
Table 6.4.2.2.1 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset. 50
Table 6.4.2.2.2 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset..... 51

Table 6.4.2.3.1 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 3 hidden layers model averaging ensemble method for validation on CB513 dataset. 52
Table 6.4.2.3.2 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 3 hidden layers model averaging ensemble method for test on CB513 dataset..... 52
Table 6.4.2.4.1 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 4 hidden layers model averaging ensemble method for validation on CB513 dataset. 53
Table 6.4.2.4.2 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 4 hidden layers model averaging ensemble method for test on CB513 dataset..... 53
Table 6.4.2.5.1 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 5 hidden layers model averaging ensemble method for validation on CB513 dataset. 54
Table 6.4.2.5.2 $Q3, QH, QE, QL$ accuracies of the SVM and Deep CNF with 5 hidden layers model averaging ensemble method for test on CB513 dataset..... 55
Table 6.4.2.6.1 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset. 55
Table 6.4.2.6.2 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset..... 56
Table 6.4.2.7.1 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 3 hidden layers model averaging ensemble method for validation on CB513 dataset. 57
Table 6.4.2.7.2 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 3 hidden layers model averaging ensemble method for test on CB513 dataset..... 57
Table 6.4.2.8.1 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 4 hidden layers model averaging ensemble method for validation on CB513 dataset. 58
Table 6.4.2.8.2 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 4 hidden layers model averaging ensemble method for test on CB513 dataset..... 59
Table 6.4.2.9.1 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 5 hidden layers model averaging ensemble method for validation on CB513 dataset. 59
Table 6.4.2.9.2 $Q3, QH, QE, QL$ accuracies of the RF and Deep CNF with 5 hidden layers model averaging ensemble method for test on CB513 dataset..... 60
Table 6.4.2.10.1 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset. 61
Table 6.4.2.10.2 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset..... 61
Table 6.4.2.11.1 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 3 hidden layers model averaging ensemble method for validation on CB513 dataset..... 62
Table 6.4.2.11.2 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 3 hidden layers model averaging ensemble method for test on CB513 dataset. 63
Table 6.4.2.12.1 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 4 hidden layers model averaging ensemble method for validation on CB513 dataset..... 64
Table 6.4.2.12.2 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 4 hidden layers model averaging ensemble method for test on CB513 dataset. 64
Table 6.4.2.13.1 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 5 hidden layers model averaging ensemble method for validation on CB513 dataset..... 65
Table 6.4.2.13.2 $Q3, QH, QE, QL$ accuracies of the SVM, RF and Deep CNF with 5 hidden layers model averaging ensemble method for test on CB513 dataset. 66
Table 6.4.2.14.1 $Q3, QH, QE, QL$ accuracies of the all methods used for test on CB513 dataset. 67
Table 6.4.2.14.2 $Q3, QH, QE, QL$ accuracies of the all methods used for validation on CB513 dataset..... 68

This thesis is dedicated to my family

Chapter 1

1.Introduction

Predict of three dimensional structure of a protein from its amino acid sequence is called as protein structure prediction [1]. Protein structure prediction has been an essential issue in the field of bioinformatics because of understanding of protein function. Three dimensional structure of a protein gives crucial information about its function. So, to understand function of a protein, protein tertiary structure has to be known. Despite being accurate, experiments are expensive and time consuming. Also, prediction tertiary structure directly has computational complexity, so some approaches aim to make predictions about features of protein's structure such as protein torsion angle, solvent accessibility, protein secondary structure.

Using computational techniques has a big increase in bioinformatics because of complexity of biological data and widespread availability of information about protein sequences. Machine learning approaches which is one of most using computational techniques have been used in some bioinformatic fields especially protein tasks. The problems with proteins such as protein structure prediction [2], protein sequences [3], protein fold recognition [4, 5, 6] have been tried to solve with this approaches.

Protein secondary structure prediction is one of the most important problems in bioinformatics. PSSP is generally defined with grouping the amino acids with three letters. These three letters are H, E and L. H is for helix E is for strand and L is used for loop.

Secondary Structure prediction has been handled by different learning algorithms. Support vector machines, one of these algorithms, has a good performance when we compare with other algorithms.

SVM has been used for protein secondary structure on the full set of 1460 proteins with three state predictions and gets 77.07 % accuracy by cross-validation [7]. Hua and Sun also have showed that it is feasible to get improvement by adding PSI-BLAST generated profiles in the SVMs method [8]. Kim and Park developed a new method, SVMpsi, to improve the current prediction rate by PSI-BLAST PSSM profiles.

The method was performed on different datasets which was RS126 and CB513 and showed that Q_3 scores for both datasets are improved by 4.9 and 3.1% [9]. Gubbi and Lai applied SVM method to CB513 dataset with seven fold cross validation technique and obtained 77.9% accuracy [10]. Also there are more studies which SVM outperforms when compare with other machine learning classifiers [11].

Ensemble learning which has an increase in using day by day is an important technique in pattern recognition, machine learning and data mining. The main idea behind ensemble learning is combining the classifiers for improving the accuracy measure [12].

In recent works, we see that many studies have been applied to improve and getting better results by combining methods. For example, King used ensemble methods that combine some machine learning approaches such as voting and get better accuracy than individual classifiers accuracies on CASP dataset [13]. Kountouris also combined machine learning techniques for secondary structure prediction and showed this technique can improve the quality of the predictions, especially SOV score [14]. Alirezaa also used a machine learning approach which was ensemble of neural networks with different voting combination methods for class imbalance problem of protein secondary structure prediction and showed their ensemble system has better performance when compare with individual classifier they used [15]. Bhola and Yadav get overall accuracy of 89.20% for predicting the protein function by using 857 sequence-derived features with different classifiers such as Random Forest, k-Nearest Neighbor (k-NN), fuzzy k-Nearest Neighbor and Support Vector Machine (SVM) [16]. Melvin and Weston developed a hybrid machine learning approach which combines nearest neighbor method and multiclass SVMs for classifying proteins [17]. Pollastri and Baldi have studied on ensembles of bidirectional recurrent neural network to improve contact and accessibility prediction [18]. Also there are more studies to show performance of ensemble approaches that much better than individual classifiers [19, 20, 21, 22, 23, 24, 25].

In this thesis, we optimized classifiers and developed new ensemble methods for protein secondary structure prediction. These ensemble methods improve the accuracy rate of secondary structure prediction and allow us to show importance of ensemble methods for protein structure prediction problem.

Protein structure prediction has vital importance to understand function of a protein. When we have information about three dimensional structure of a protein, we

will also have information about its function and by this we have power to diagnose disease and design new drugs,

This thesis is organized as follows: section 2 gives information about proteins and their structures. In section 3, protein structure prediction and importance of secondary structure prediction is explained briefly. Pre-processing methods about preparing feature vectors as the first stage of sequence alignment is elucidated in section 4 and then, we clarified methods we used and parameter optimization for prediction as a second stage. Ensemble methods used stated in section 5 and in section 6 we explain our results and give a conclusion.

Chapter 2

2.Structure of a Protein

Proteins are polymers of amino acids, consisting of one or more long one polypeptide chains and each protein has a specific function. “Proteins perform a vast array of functions within organisms, including catalysing metabolic reactions, DNA replication, responding to stimuli, and transporting molecules from one location to another. Proteins differ from one another primarily in their sequence of amino acids, which is dictated by the nucleotide sequence of their genes, and which usually results in protein folding into a specific three-dimensional structure that determines its activity.”[26]

Amino acids are constituents of proteins and they specify three dimension structure of proteins. Despite of having the same general structure, the side chain(R group) of each amino acid is different(Figure 2.1). Difference of proteins is possible with different sequence of amino acids and these side chains provide different structure attributes to protein.

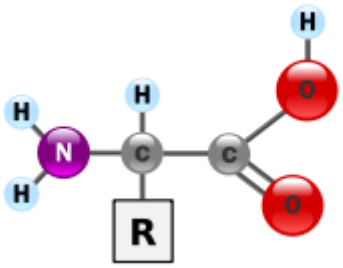


Figure 0 The structure of an amino acid[27].

Proteins are large macromolecule class as a result of aminoacids connecting to each other in a sequence. There are commonly twenty types of amino acids in nature. These amino acids have different physical and chemical character. Characters such as electrostatic charge, acid separation coefficient, hydrophobic, size, functional group differs from one amino acid to another. These properties plays an important role in determining the structure of a protein [27].

2.1 Protein Structure Levels

There are four basic levels of a protein structure: primary, secondary, tertiary and quaternary.

Primary structure=amino acid sequence

Secondary structure=hydrogen bonds

Tertiary structure= three dimensional structure of an amino acid chain

Quaternary structure=three dimensional structure of multiple amino acid chains

2.1.1 Primary Structure

Primary structure basically can be defined as amino acid sequence of a protein. Amino acid sequence gives us information about history of protein. The primary structure of a protein consist of all necessary information to determine the protein in the 3D structure. This structure is crucial for being informed about function of a protein. The alteration one of amino acid in sequence can change entire protein.

In summary, primary structure is important;

- to predict secondary and tertiary structure.

-many genetic diseases occurs due to abnormal amino acid sequences

-to get information about the molecular system of proteins

-to determine evolutionary history of protein

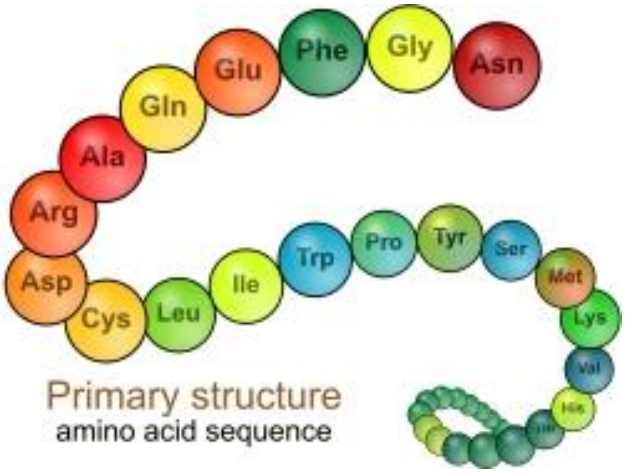


Figure 0.1 Primary structure of an amino acid[27].

2.1.2 Secondary Structure

Secondary structure of a protein is local structural conformation that formed by coiling and folding of polypeptide chains. There are two regular and major structural element of secondary structure. These are α helix and β strand.

Coiling occurs, forming a **α helix**, due to the formation of hydrogen bonds between the nitrogen of one amino acid and the oxygen of another located in another

part of the polypeptide chain. It is usually right-handed in proteins. And each helix contain from 5 to 40 amino acids.

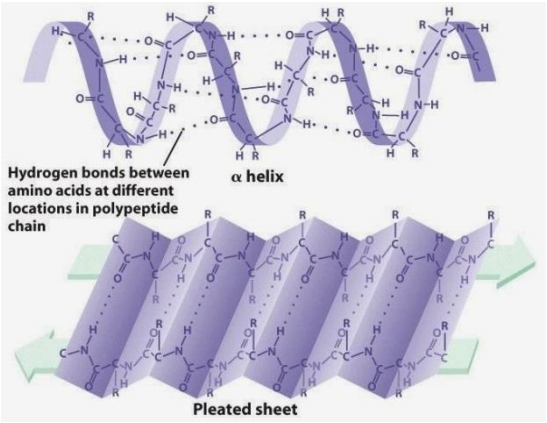


Figure 0.2.1 Secondary structure of an amino acid[28]

Folding occurs, forming **β pleated sheets**, due to hydrogen bonding between different polypeptide chains lying side by side.

There are two types of pleated sheets which are parallel and antiparallel. Parallel β pleated sheets occur when two peptide run in the same direction. For anti parallel, they must run in the opposite directions.

Loops are not regular secondary structure in proteins unlike alpha helices and beta sheets. They are forming near the helix and beta sheet and usually found at surface of protein. Loops contains turns, random coils and strands which connect the alpha helices and beta strands.

2.1.3 Tertiary Structure

Tertiary structure is three dimensional form of an entire protein molecule. Every protein has unique three dimensional structure and the function of protein depends on it. The protein with known structure gives crucial information about its function.

By understanding function of structure, we can try to diagnose diseases, design drugs and investigate new treatment models.

3D structure is determined by X-ray crystallography and NMR(Nuclear Magnetic Resonance) spectroscopy. The known structures have come to light through these methods. The atomic coordinates of most of these structures are collected in a database known as the Protein Data Bank(PDB). PDB allows to analyze the tertiary structures of known proteins. Despite of getting information about structure of protein, these methods are also expensive and time consuming. So, computational methods are tried to applied in bioinformatic fields.

2.1.4 Quaternary Structure

Quaternary structure is the combination of two or more tertiary units. It is stabilized by non-covalent and disulfide bond which also stabilizes tertiary structures.

2.2 Torsion Angle

Torsion angles are defined as angles of rotation of protein backbone around the bonds. Torsion angles are also called as dihedral angles. There are three types of torsion angles; phi, psi and omega. Rotation angle between N and C_α is called as **phi**, Rotation angle between C_α and C atom of C=O groups is called as **psi** and rotation angle of the protein around the peptide bond is identified as **omega(**ω). In general, omega is flat and fixed to 180 degrees, so phi and psi provide the flexibility required for the polypeptide backbone to adopt a certain fold[29].

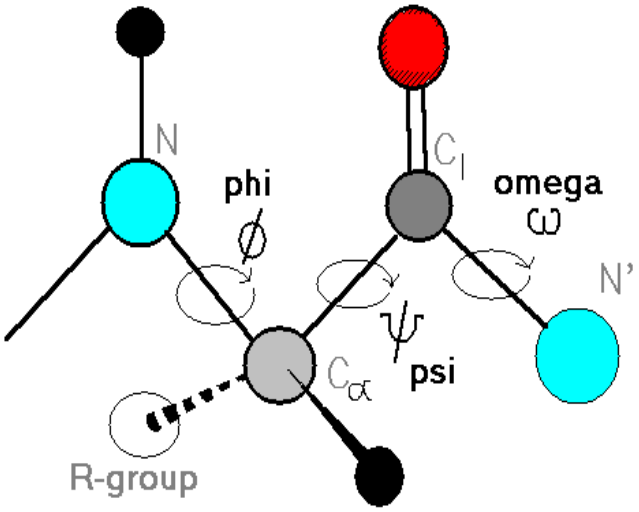


Figure 2.2.1 The phi (φ), psi (ψ) and omega (ω) angles in the protein backbone[30]

2.3 Solvent Acessibility

The accessible surface is identified as the area of a biomolecule on the surface that can be reachable by water. All molecular surface area is not accessible to solvent because of the presence of small spaces. Solvent accesible surface area is one of most important measure to understand interaction tendencies of a protein.

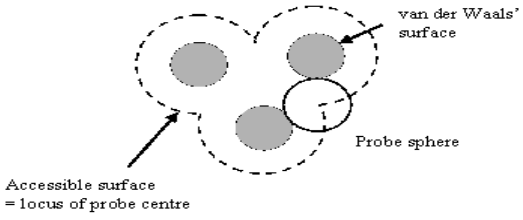


Figure 2.2.2 Van der Waals area and solvent accessible area[31].

Chapter 3

3.Protein Structure Prediction

Protein structure prediction is a crucial topic in bioinformatic due to relation between a function of a protein and its three dimensional structure.

The main reason for tackling the protein structure prediction problem is necessity to understand all functionalities they are related to which configurate its ill effect on others as well as its good effect. Although a lot of researches on protein structure prediction have been handled, the structure prediction problem has not been solved yet.

3.1 Importance of Secondary Structure Prediction

Secondary structure prediction is the identification of secondary structural elements starting from the sequence information of proteins. The aim with this prediction is to assign secondary structure class (helix, strands, loop) which corresponds to each amino acid.(figure 1)

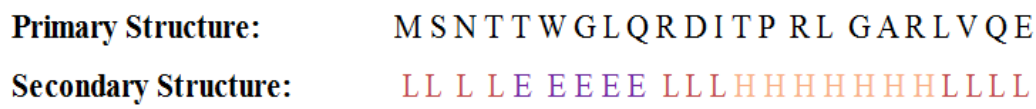


Figure 2.2.1 Three state of a protein secondary structure prediction. The first line represents the amino acid sequence, the second line represents the secondary (H: coil, E: beta strand L: loop).

For secondary structure prediction, generally supervised learning approaches are used. In Supervised learning, a model is trained from the database with known secondary structure to make prediction for unknown structure. The expectation is to estimate unknown information thanks to information which is already obtained from model.

3.1.1 Secondary Structure Types

3.1.1.1 DSSP

DSSP (dictionary of secondary structure prediction) is a program that defines secondary structure of a protein with an 8 state assignment denoted by single letter codes.

3.1.1.1.1 8 class secondary structure prediction

H	α - helix
G	3_{10} – helix
I	π - helix (extremely rare)
E	β - strand
B	β - bridge
T	β- turn
S	Bend
L	the rest

Table 4.2.5.1.1.1 8 class secondary structure prediction

However, prediction methods are generallly trained and evaluated for three states which are H,E,L. H refers to helix, E refers to strands and L refers to coil(loop). For this evaluation, 8 states is converted to 3 states. There are many published 8 to 3 states reduction methods.

There are three standard reduction methods which are mostly used. They are defined by programs DSSP[32], DEFINE[33] andSTRIDE[34].

3.1.1.1.2 3 class secondary structure prediction

DSSP programs use reduction methods as below:

{G (3_{10} – helix), H (α- helix)} ⟶ H (helix)

{E (β-strand), B (β-bridge)} ⟶ E (strands)

all the rest ⟶ L (coil)

DEFINE programs use reduction methods as below:

{G (3_{10} – helix), H (α- helix)} ⟶ H (helix)

{E (β-strand)} ⟶ E (strands)

all the rest ⟶ L (coil)

STRIDE programs use reduction methods as below:

{G (3_{10} – helix), H (α- helix)} ⟶ H (helix)

{E (β-strand), B (isolated β-bridge)} ⟶ E (strands)

all the rest ⟶ L (coil)

3.2 Measures for Prediction Accuracy

Q_3 is most popular measure in literature for protein secondary structure prediction methods when evaluating performance of classifiers. It can be definable as percentage of all amino acids that have correct matches for the three states (H, E, L).

$$Q_3 = \frac{100x(TP_H + TP_E + TP_L)}{N} \tag{3.2.1}$$

where TP_H is used for true positive number of helix, TP_E is used for true positive number of strand, TP_L is used for true positive number of loop and N is for number of total data.

According to 3.2.1 equation we can obtain each class label of secondary structure prediction.

CL ∈ {H, E, L} $$Q_{CL} = \frac{100xTP_{CL}}{N_{CL}} \tag{3.2.2}$$

where Q_{CL} is rate of class label individually for protein secondary structure prediction, TP_{CL} is number of residues correctly in state CL and N_{CL} is the number of residues in state CL.

Also there are other measures for evaluation. For example, Q_8 is measure for eight states, while SOV is used for testing average overlap between the observed and the predicted segments rather than individual residues [35].

Chapter 4

4.Methods

Making analyze of very large quantities of data is not possible manually. So, at this stage we get help from machine learning methods. The main purpose of machine learning methods is to make predictions for the future using the past data.

In our thesis, firstly we used dspred method which is a hybrid approach for the prediction of one dimensional features and then used feature extraction methods. Also, we used support vector machines, random forest and deep convolutional neural field methods as classifiers.

4.1 DSPRED METHOD

In our thesis, a hybrid approach, DSPRED, will be followed that uses both dynamic Bayesian networks and a classifier together for the estimation of one dimensional local features such as secondary structure, dihedral angle and solvent accessibility. This approach has been previously applied for dihedral angle estimation and successful results have been obtained [40]. In this method, there are separate dynamic Bayesian network (DBN) for position specific scoring matrices obtained from PSI-BLAST and Hidden Markov Models (PSIBLAST PSSM and HHMAKE PSSM) by using proteins with known secondary structure labels.

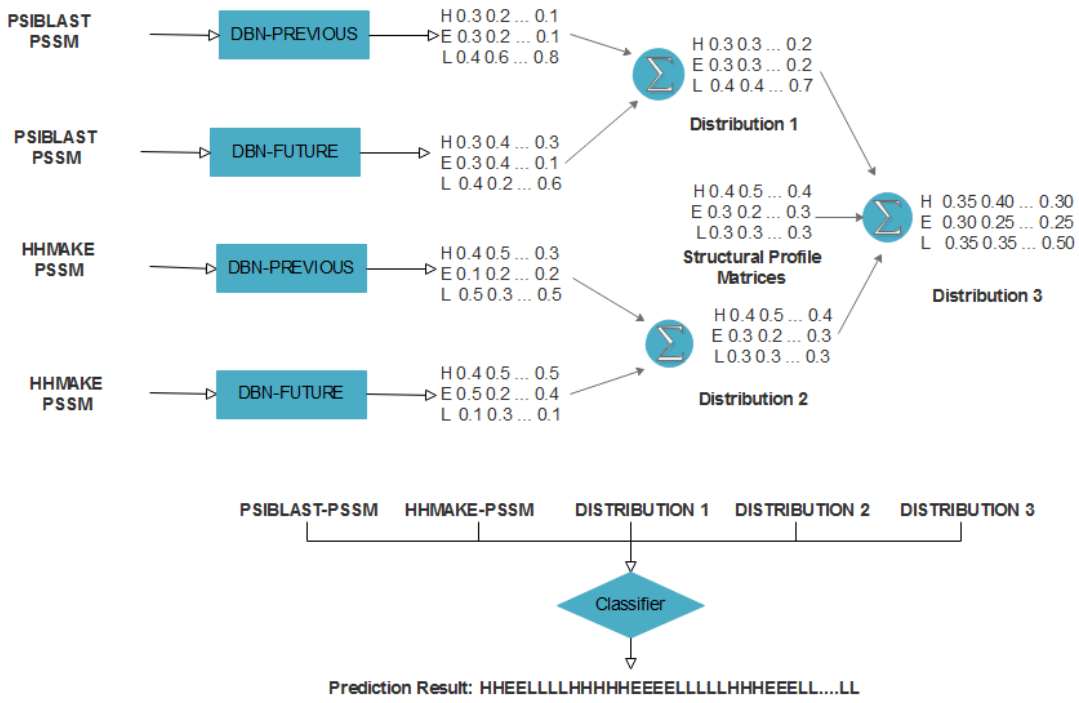
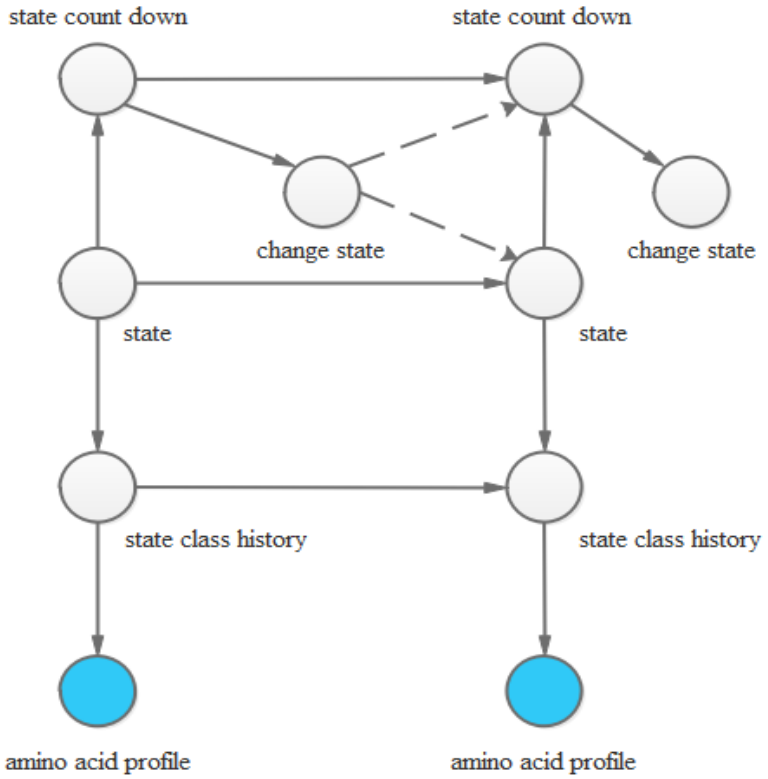


Figure 0.1Estimation of the 3-state secondary structure with the hybrid model using dynamic bayes networks and artificial neural networks together

Here, DBN-Previous represents the model in which the profile vector at a given position is bound to its previous positions and DBN-future represents the model in which the position of the profile vector at a given position is dependent on subsequent positions (the vectors are the columns of the profile matrix and there are as many columns as the number of amino acids). Therefore, two types of DBN models are trained for each profile matrix (four DBN in total). The average of probability distributions for the secondary structure class labels from these DBN classifiers gets in various combinations. For example, distribution 1 represents the average of the predictive distributions from the PSI-BLAST profile, distribution 2 represents the mean of the distributions from the profile matrices derived from the hidden Markov model, and distribution 3 is used for the average of the distributions 1 and 2. In this problem, since the 3-degree secondary structure class is estimated, the size of Distribution 1, 2, and 3 is 3×U where U is the number of amino acids. Consequently, each column contains the estimated probabilities of secondary structure classes at that position. In the later stage, the profile matrices (PSI-BLAST and HHMAKE) used for DBN are

combined with Distributions 1, 2, and 3 and sent to classifiers which will be explained in later chapter. Here a symmetrical window is taken around the amino acid position at which the secondary structure class is predicted. Then, the columns of the profile matrices and the columns 1, 2, and 3 corresponding to these windowed positions are used as the input parameter (feature). Finally, the output from classifiers gives an estimate of the secondary structure class of the amino acid in the middle of the window. In our thesis, in addition to PSI-BLAST and HHMAKE profiles, profile matrices derived from structural profile approach is used as input parameters and separate DBN models is trained for these matrices. Then, after multiplying by the various coefficients of the probability distributions obtained from these DBN models, averaging is performed and the final distribution is calculated.



(A)

State	H	H	H	H	H	L	L	L	H	H	H	E	E	E	E	E	L	L	L	L	L	L
state count down	5	4	3	2	1	3	2	1	3	2	1	5	4	3	2	1	5	5	4	3	2	1
change state	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0

(B)

Figure 0.2 A) dynamic bayesian network for protein secondary structure prediction. B) the variables used in DBN is shown. State variable is for secondary structure class label. The state count down with Dmax=5 shows the value of remaining class label from current position until next label. Change state is used for transition from one label to another. Transition state is showed with 1 and when staying in same class label, it is shown with 0.

The dynamic Bayesian network model used for predicting secondary structure class with 3 states is shown in Figure 1 [40]. Dynamic bayesian network (DBN) is the superset of Hidden Markov Model (HMM) [41]. We performed the DBN which is designed by [40]. The nodes in DBN show random variables. The model we used includes five variables. Blue circles show the profile matrices and the others show hidden conditions. The state variable represents the secondary structure class label to be estimated. In training part, the observation is made on state variable and in execution time for estimation, it is hidden. The amino acid profile variable includes the data which is observed. This observation data is the PSSM scores with 20 dimensional vector which is produced from PSI-BLAST and HHMAKE methods. The current and previous SS label information is kept with state class history variable. The state count down identifies each class label with its length number in a descending order. There is a Dmax value for this variable. If length of a class label is equal or less than this number, then the state count down value is equal to the number of this class label letters which start from current position until next different class label. If length of a class label is bigger than Dmax value, then the state count down value is starts with this number and does not change until the number of residues of class label is equal to Dmax. When equality is achieved, then the procedure which explained below is applied. Change state shows passing from one class label to another.

DBNs model the generation of input parameters from hidden class variable within the probability rules. In our DBN model, the probability of producing a particular profile matrix from a given class of structures is modeled with a gaussian probability distribution. After the proteins with known structured labels are trained with DBN models, the state sequence with the highest probability is computed by efficient algorithms. In our thesis, Linux

based GMTK software package (Graphical Models Toolkit) is used to create DBN models [42]. The GMTK uses the EM algorithm for model training and the junction tree algorithm for estimates.

4.2 Feature Extraction

The success of three-dimensional structure estimation approaches using particle selection is closely related to the quality of selected particles. For example, it is important that these particles are the same as the correct structure or similar to the correct structure. Therefore, the better particle selection makes possibility of better prediction for three dimensional structures. Various one dimensional structural properties of the protein such as profile matrices and secondary structure, dihedral angle and solvent accessibility are used as input for particle selection. Therefore, it is important to estimate these matrices and their structural properties with low misperceptions for selection of the correct particles and prediction of the three-dimensional structure correctly. Although many advanced methods have been proposed for predicting the structural properties and three-dimensional structure of proteins, the scores obtained from the PSIBLAST algorithm are often used as input for the profile matrix. In addition to the matrices obtained from the PSI-BLAST method as profile matrices, matrices obtained from hidden Markov models and structural profile matrices is used in our thesis. Structural profile matrices have recently begun to be used for secondary structure estimation and particle selection, but the profile matrices obtained from hidden Markov models are quite new. Once the profile matrices have been computed, one dimensional structural property such as protein secondary structure will be estimated.

4.2.1 PSSM

A PSSM depend on the frequencies of each amino acid in a specific position of a multiple alignment. A profile is formed from a series of functionally related aligned sequence. After profile is derived from a series of sequence, a window with length you specified is run. According to this window length, you will have feature number. For example, if you have a window with length 4, 9 features will be derived for each amino acid that 4 from the right side of amino acid, 4 from the left side of amino acid and 1 from itself. For all amino acid, 9*20=180 feature will be derived.

4.2.2 Position Specific Iterative BLAST (PSI-BLAST)

PSIBLAST is an iterative method that searches database by using sequences of multiple alignments which has found in high score in each search to achieve a new PSSM to use in the next time of searching. At the end iteration a profile matrix size 20*N occurs by PSI-BLAST method. N is the number of amino acids in the target protein. Proteins with structural similarity whose sequence similarity is low can be discovered with the use of profiles in alignment. Therefore, they are applied to the profile evaluation. Because of being fast and simple to implement, possibility to search PSSMs on large database, providing efficiency and sensitivity, PSI-BLAST is the most commonly used profile matrix derivation method for structure prediction. However, PSI-BLAST method can not only detect more distant protein similarities, but also perform mismatches. So, the profile matrices derived from this method contain a certain noise. Although this noise plays a difficult role in predicting the structural properties, the first profile matrix to be used in this project will be obtained by the PSI-BLAST method due to widely using and having a certain level of accuracy.

4.2.3 HMM Profile Matrices

Profiles based on hidden Markov models (HMM) can be obtained, after multiple alignments are executed with the proteins found by sequence

alignment algorithms. HMM can be iteratively used for profile sequence alignment or profile-profile alignment [36]. HMM-profiles is generally used as a classifier in field of hand writing recognition, sound recognition, bioinformatics, gene prediction. They are applied to models as a classifier in previous work for speech recognition [37], protein secondary structure prediction [38], protein torsion angle prediction [39].

HMM profiles are more sensitive than standard profiles and explore more distant protein similarities. However, profile alignments based on Hidden Markov Model are slower when compared with PSI-BLAST method. In our thesis, HMMs derived at the end of first iteration with HHBlits method will be converted to PSSM and used as the second type profile matrix.

4.2.4 Structural Profiles

In addition to profile matrices based on multiple alignments of amino acid sequences for structural prediction, structural profile matrices have also been used as attributes. Structural profile matrices are constructed using the structural sequences of the proteins found by sequence alignment methods. The dimension of a structural profile matrix formed for the secondary structure estimation is $3 \times N$. Here, N is the number of amino acids in the target protein and each column has the observation score of one of the three states of secondary structures for that amino acid. Since structural profilers also use structural information of template proteins that are similar to the target protein, they can be evaluated separately from the methods that use sequence profiles. Another category is directly using secondary structural information of pattern proteins that has a sequence similarity to target protein over a certain level for prediction. In this case, the situation in which structural profiles are used can be considered as category between the case where the sequential profiles are used and the case where the proteins are directly used for prediction.

The protein sequences used in the construction of structural profiling are so healing that the predictions of the target protein are very similar. Furthermore, in cases where the target protein resembles a sub-region instead of the entire protein (local similarity), there is some improvement in the prediction of the

secondary structure and a more significant improvement in the estimation of the secondary structure can be achieved when the resembling region is longer. It is aimed to capture distant or local similarities by using structural profiles. On this basis, structural information of local similarities can be used to estimate the structural properties of the target protein.

HMM-Profile model obtained by NR alignment in the first stage of HHBlits is aligned with the profile-profile alignment by the HMM-profile models established for PDB proteins in the second stage. Structural profiles matrix is constructed with the secondary structure labels of PDB proteins above a certain value of the percentage of sequence identity and by using the proteins above the probability score threshold. The frequencies of the other secondary structure tags that match each amino acid in the input protein are calculated as a matrix and the columns are normalized in itself [38].

Our structural profile is a 3 × N matrix. 3 is used for secondary structure class labels and N is used for amino acids in the target. In Fig. 5.1.4.1, an example for structural profile is shown.

	1	2	…	N
H	0.4	0.5	…	0.2
E	0.3	0.5	…	0.3
L	0.3	0	…	0.5

Figure 4.2.4.1A structural profiles for 3 state of secondary structure representation. Rows show the secondary structure classes and columns demonsrate the amino acids of the target. Sum of each column is equal to 1.

Once the profile matrices for proteins have been generated, the feature vector for each amino acid is obtained by taking a window around that amino

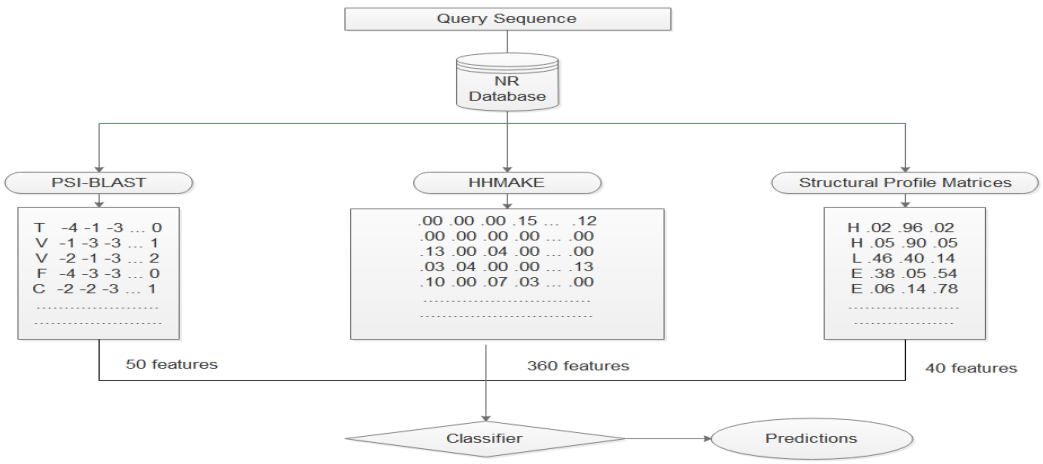


Figure 4.2.4.2 An example for obtaining feature vectors

acid and concatenating the values from the respective columns of PSI-BLAST, HHMAKE and structural profile matrix. An example for obtaining feature vectors is shown in figure 4.2.4.2

4.2.5 Datasets

We make our experiment with one dataset: CB513[43] dataset containing 513 proteins and 84,119 residues. It contains 396 sequences, usually named CB396 and the 117 sequences from RS126 [44] database.

CB513 is a popular dataset in bioinformatic fields. It was used in many studies about protein secondary structure prediction.

4.2.5.1 K-fold Cross Validation

Datasets used in machine learning, data mining generally split into two parts as train and test set to evaluate predictive models. Train set is to train the model and test set is to evaluate it.

K-fold cross validation is a method to test the model used by dividing the data set into k equal size subsets. In k fold cross validation technique, each time, one of the k subsets is used as the test set and the other k-1 subsets are acting as a training set. This repeats k times until all subsets are used as test set.

For example, in our thesis, we used 7 fold cross validation. Our k value is 7. We divide our dataset into 7 equal size subsets. After we randomly divide our training dataset into 7 part, one of these is used as test to validate and the other 6 subsets are used as training. After selecting all each fold as test set and evaluate the model, we have 7 accuracy rate. Overall accuracy is calculated by taking average of 7 results obtained.

Equation can be written as below:

$$t_i \in DS \qquad Overall\ accuracy = \frac{\sum_{i=0}^{k} CF(t_i, DS - t_i)}{k} \qquad \textbf{(4.2.5.1.1)}$$

CF is classification function, DS is dataset, k is fold number used and t is for selected each test set from dataset.

Overall accuracy is found by dividing the sum of classification function result to the number of k.

Cross validation also is a solution to find optimum parameters. By this way, we make optimization on validation sets and we make our work on test with optimum parameters which is found from validation dataset. And this gives us a crucial convenience about finding optimum parameters which provides a rise on accuracy rate.

4.3 Support Vector Machines

Simplifying datas and making prediction are two main goals in classification. In general, machine learning approaches have been applied to classification problem. Support vector machine is one of classification method used in machine learning. Main purpose of SVM is to identify hyperplane which makes the most appropriate discrimination between two or more classes[45].

SVM can classify both linear and nonlinear datasets. Let say we have two classes and we want to classify them. We can draw an infinite number of plane between these two classes. At this point, the aim of SVM is to find the hyperplane that maximizes the distance between the closest points of classes to itself.

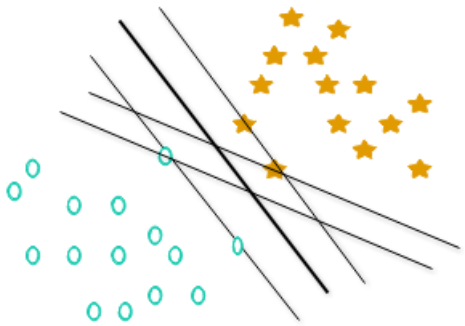


Figure 4.3.1.1 Linear SVM classifier

4.3.1Linear Separation

In linear separation, datas from different classes can be linearly separated from each other in different ways. As seen in figure 1, datas in different class can be separated with lots of number planes. Two

hyperplanes (H_1, H_2), which are farthest from each other, are considered to be the most suitable way. The H_0 hyperplane, which forms the center of these two hyperplanes, is a linear hyperplane that separates the two classes of datas. These H_0 hyperplane is identified as the optimal separation hyperplane.

Given a training set D which can shown as;

$$\boldsymbol{D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}} \quad \textbf{(5.1.1.1)}$$

n is the number of element in D dataset and y is accepted as element of $\{+1, -1\}$.

The H_0 plane in terms of points on a hyperplane can be expressed as:

$$\boldsymbol{H_0: w^T X + b = 0} \quad \textbf{(5.1.1.2)}$$

Also can be written as:

$$\sum_{i=1}^{n} \boldsymbol{w_i x_i + b = 0} \quad \textbf{(5.1.1.3)}$$

where **w** is weight vector as W=$\{w_1, w_2, w_3, \dots, w_n\}$. b is a constant number.

The $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ hyperplanes can be expressed azs:

$$\boldsymbol{H_1: w^T X + b = 1} \quad \textbf{(5.1.1.4)}$$

$$\boldsymbol{H_2: w^T X + b = -1} \quad \textbf{(5.1.1.5)}$$

The points above the hyperplane which is formulated in the first formula correspond to the following inequality:

$$\boldsymbol{w^T X + b > 0, \quad y_1 = +1} \quad \textbf{(5.1.1.6)}$$

Similarly, the points at the bottom of the hyperplane correspond to the following inequality:

$$\boldsymbol{w^T X + b < 0, \quad y_2 = -1} \quad \textbf{(5.1.1.7)}$$

4.3.2 Non-linear Separation

In real word problems, many datasets are not linearly separable. In this case, the datas can not be split by a linear function, so non-linear mapping approach [46] is applied. The two dimensional dataset is processed into three dimensional feature space and linear separation of the dataset is achieved with this approach.

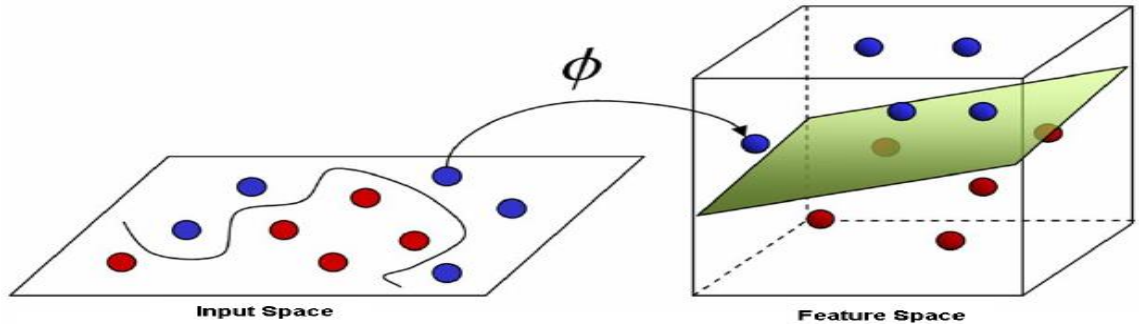


Figure 4.3.2.1 The non-linear SVM[46]

4.3.3 MULTICLASS SVM

SVM is used to split only two classes. If there are three or more classes to be classified, it is identified as multi class SVM. Different approaches can be used for multi class SVM.

One versus one (OVO): This is the most used technique. In this approach, problem is reduced to binary groups. For example, we have three classes as 0, 1 and 2. Firstly, these classes are trained by SVM method in binary groups and the SVM multipliers of the classes which relative to each other are generated. According to results, which one of these classes is more seen as an output of these queries, a new sample is accepted as closer to this class.

As seen in figure, a separate SVM has been trained for both classes.

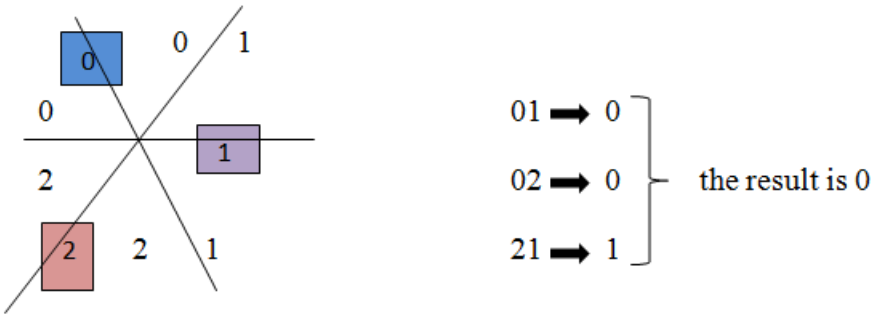


Figure 4.3.3.1 Multiclass SVM

One versus all (OVA): In OVA technique, problem is separated into binary classes. One class is considered as positive and all other classes as negative. Then, SVM algorithm is implemented and this is applied for all classes individually.

Multi Class Ranking: A SVM algorithm works on the classes to make a classification that includes all the members of these classes according to multi class ranking approach. This approach is the least preferred one. Because, the execution time is unbelievably high when compared with other approaches.

4.4 RANDOMFOREST

Randomforest is a method that forms a forest of random trees that used for classification or regression with different sets of features [47]. It is also definable as an ensemble classifier because of using many decision tree models. Application areas of random forest are very large. For example, it uses in biomedical [48], physics, health, bioinformatics etc.

Random forests construct decision tree with n number over the available training set. A new training dataset is created with displacement from the actual dataset for each tree. For this reason, each tree is different from the others. In the same way, selected attributes for constructing trees are again chosen randomly

with m number. And it starts to form tree by using gini index to specify branching criteria in the random forest. When it gets number of trees which determinated from the beginning, output of each tree is collected and combined by weighted values to find the final classifier.

Random forests are preferred for a number of reasons. One of this advantage and may be most important one is overcoming the problem of overfitting. Random forest can avoid this problem by optimization tuning parameter, number of trees and training features. By choosing most suitable model parameters based on cross validation will form a model without overfitting. Also, overfitting problem can be prevented by number of trees. Because, if number of trees is getting larger, the forest will more overcome of this problem. This implies that you get more alternative to select, because each tree in random forest is learning some aspect of the training data.

The other advantage of random forest that makes it an active research subject is with regard to efficient performance on big datas. Fastness of learning, convenience about setting parameters, handling missing values are demonstrable as other advantages of random forests.

In recent works, random forests are compared with other classification models. For example, a comparison made between randomforest and SVM for microarray-based cancer classification and seen that SVM outperform random forest [49]. Also, Lee et al. [50] compared random forest with SVM to identify protein function with features which obtain from protein sequences attributes. They applied a correlation-based feature selection to models and compared them with SVM and Random forest without feature selection.

4.5 Deep Convolutional Neural Fields

Deep learning is a machine learning approaches to find solution for problems in field such as bioinformatics [51, 52], natural language processing[53, 54] with deep neural network architectures. The goal of deep learning approaches is to learn an attribute order with high level attributes of the

order which is formed by the connection of lower level attributes[55]. Learning attributes at multiple levels of abstraction provide opportunity for a a system about resolving complicated functions that maps the inputs to an output directly from datas.

Deep learning has been improved by the time. Firstly, deep belief network with RBM[10] which is an an algorithm that trains a layer at each time was introduced. After a while later, connected algorithms formed on auto-encoders were introduced and then, many algorithms started to propose.

4.5.1 Deep Convolutional Neural Networks

Convolutional neural networks are enviable deep learning architecture because of the successful training of the hierarchical layers. In Convolutional Neural Networks, the convolution has replaced the general matrix multiplication in standard NNs. In this way, the number of weights is decreased, thereby reducing the complexity of the network. Another advantage of CNN is requirement minimal pre-processing.

4.5.2 Conditional Neural Fields

DeepCNF can be define as a improved form of CNF which has a combine between CRF and DCNN. Conditional neural fields is created due to the inconvenience of resolving the nonlinear connection between input features and output layer of CRF, especially for sequence labeling [56]. DEEP CNF is a combine of CNF [56] and DCNN [57]. It receives both advantages of CNF and DCNN. DeepCNF also design connection between contiguous secondary structure labels, while CNF design only complicated and nonlinear contact between sequence-structure [58]. So, we can say it is better than CNF with this attribute.

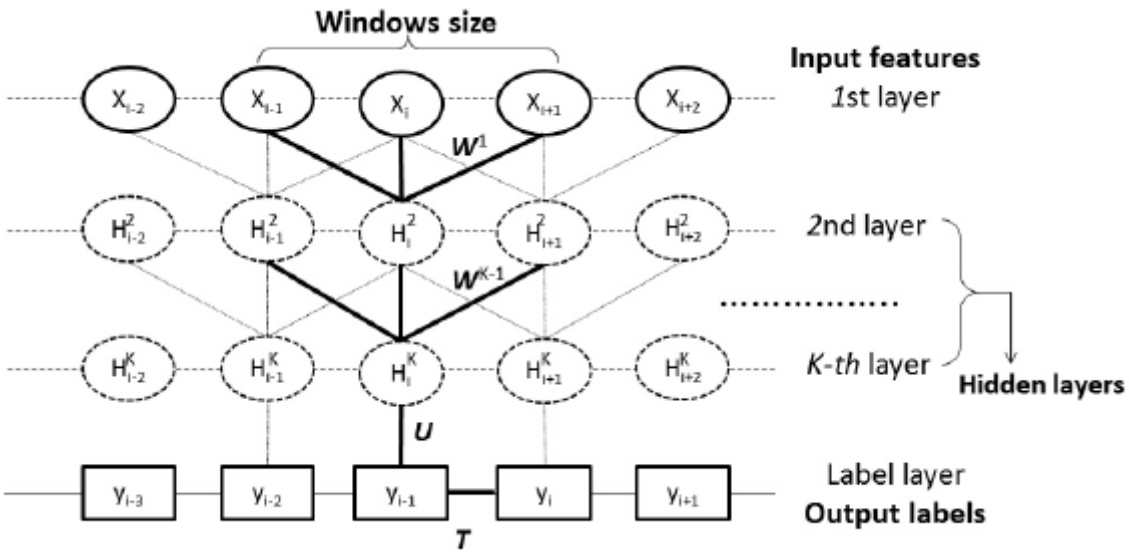


Figure 4.5.2.1Design of DeepCNF[59] , where i is the position index and X_i the associated input features, while H^k is for the k-th hidden layer and Y fo the output label. All the layers from the first to the top layer constitutes a DCNN with parameter $W^k\{k = 1,2, \dots, K\}$. The top layer and the label layer constitutes a CRF with U and T as a model parameters. U determines association between output of the top layer and the label layer and T is used for adjacent label correlation.

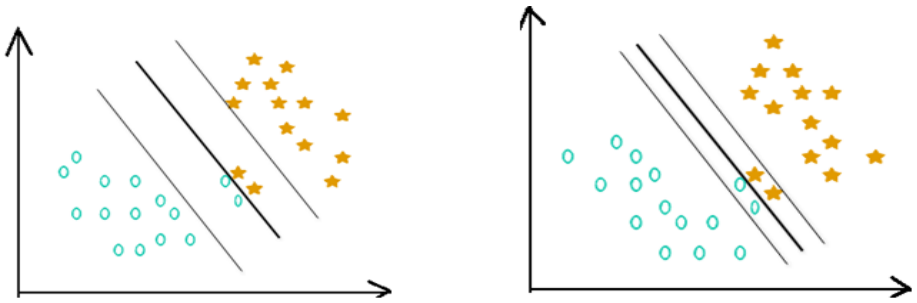
4.6 Parameter Optimization

There are multiple situations that will affect the performance of a method. One of these situations is selection of correct parameter.Hyperparameters are parameters that cannot be directly learned from the regular training process. Although the methods have default hyperparameter values, it is possible to improve performance of methods by using those hyperparameters with different values which can be more convenient. Parameter optimization plays an important role at this stage. It applies to methods to find best hyperparameter values to apply methods.

In our thesis we used 3 different classifiers for optimization. Firstly support vector machine with different C and gamma parameters, random forest by using different number of iterations and deep convolutional neural field with 3 different parameters that are node string, window string and regularization coefficient.

4.6.1 Parameter Optimization for Support Vector Machines

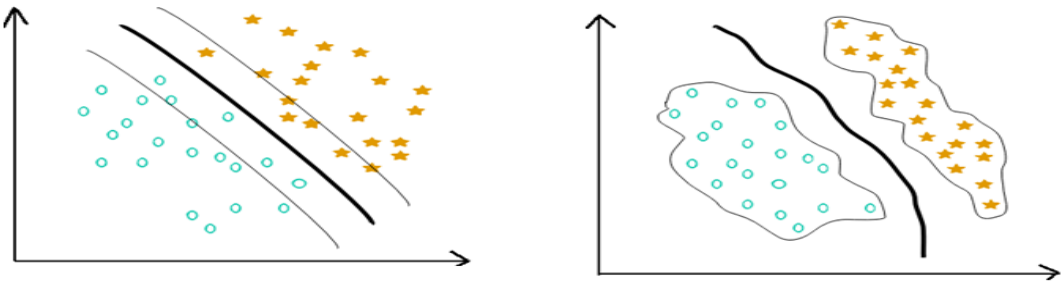
In SVM, we optimized the kernel function parameters which are C and Gamma (γ). C is a cost function parameter that checks the effect of each support vector. Selecting an appropriate value for C is important for tolerating the error. When the value of C is low, the decision surface is getting smooth and margin width becomes maximum. When the value of C is high, sensitivity at learning phase increases because of decreasing margin width.



a)C=10 **b)C=100**

Figure 4.6.1.1 The effect of C constant on the decision boundary a) The effect of C when equal to 10 and b) The effect of C when equal to 100.

Gamma parameter determines the amount of spreading influence. The decision boundary becomes more linear when value of gamma parameters is small. When gamma parameters take high value, the decision boundary becomes non-linear.



Gamma= 0.1 **Gamma=10**

Figure 4.6.1.2 The effect of kernel parameter on decision boundary

We took combination of C and gamma parameters for SVM optimization. For C and gamma, we have 10 values separately. Totally, we have 10*10=100 different combination results for each fold. At the end of execution, we found the combination that has best accuracy rate and use it for train-test datasets.

The aim is to test the SVM method with given parameters on validation datasets. The outputs we obtained from model will be the performance rate of the SVM method. The model parameters which have the highest performance ratio will be selected as the optimum parameters. Then, we applied SVM method on the test data with the optimum parameters we found and calculate correctness of the classification.

4.6.2 Parameter Optimization for Random Forest

We have one parameter that will be optimized for our randomforest model we used. Number of iteration is optimized for our model. Number of iteration means number of trees that will be used in randomforest. If the number of trees is larger, the result will be better. But, execution time will be longer. We select 24 different values to be optimized for each fold. So, for each fold we have 24 different results. After optimization, we found optimum parameters of number of iteration for each fold and applied them for train-test datasets.

4.6.3 Parameter Optimization for Deep Convolutional Neural Fields

Our deepCNF model has 3 hyperparameter to be optimized. These are node string, window string and regularization coefficient. Node string parameter defines hidden node combination of model. The number of node string gives us the number of neurons used for each hidden layer. Window string is used for kernel window size at each layer of deepCNF. For example, when window string is 5, window size will be 11 for each node (1 for it, 5 for next nodes and 5 for previous nodes). Regularization coefficient is a fine-tune regularizer. It is used for avoiding overfitting. L_2-norm of regularization is used. L_2-norm

provides to minimize the sum of the square of the differences between the target value and the estimated values.

We used node string, window string and regularization coefficient parameters in a combination for deep convolutional neural fields optimization. For node string, window string and regularization coefficient parameters, we have 3 values separately. Totally, we have 3*3*3=27 different combination results for each fold. At the end of execution, we found the combination that has best accuracy rate and use it for train-test datasets.

For an ensemble method, we also optimize the number of hidden layer. Each layer has a size in a multi layer neural network that can be set in different values and it controls the capacity. We added this parameter between the other hyper parameters and try to find optimum one that gives best accuracy rate.

Chapter 5

5.Ensemble Methods

The general approach for model selection is which one gives best result it is the best classifier. However, results of recent works in machine learning show that the performance of the final classifier can be improved by forming a method whose output will be a combination with outputs of classifiers that have different algorithms. The basic idea behind ensemble methods is to combine the predictions of several classifier for the purpose of improving estimation and robustness of models.

5.1 Combining Hidden Layers for Deep CNF

Hidden layer number is an important issue in neural network. Selecting hidden layers in different numbers and according to results combining them is a different ensemble approach.

In each fold, we took the optimum result set in the validation train sets and combined the results. According to results, we also take the number of hidden layers in the configuration. By applying hidden layers to configuration, we aim to take best accuracy rate from combination of all hidden layers and improve performance of model on this.

5.2 Model Averaging

Model averaging is a successful approach to improve the accuracy of secondary structure predictions by averaging over many methods to generate a consensus prediction. The main principle for model averaging methods is to form several classifiers individually and then averaging their prediction results. On model averaging, combined classifiers generally gets better results than any of the individual classifier because of its reducing variance.

Suppose we have N different classifiers n = 1 , … , N with probabilities of each class labels of secondary structure of protein.

$$p(X) = \frac{1}{N}\sum_{n=1}^{N} p(n) \tag{5.2.1}$$

P(X) is final probability of each class label at the end of model averaging. According to result of final probability, new class labels predictions are formed and compared with the correct class labels.

5.2.1 SVM+Randomforest Model Averaging

The first method we implement for model averaging was combination of Support Vector Machine and Randomforest methods. In this method, firstly each method is executed individually. At the end of execution, observed and predicted secondary structure class labels are obtained. Both results of methods are averaged and a new prediction is generated. This prediction is compared with the correct results and a new prediction rate is achieved.

5.2.2 SVM+DeepCNF Model Averaging

The second method we implement for model averaging was combination of Support Vector Machine and DeepCNF methods. The method applied is the same with the first method. Each method is executed individually and then, observed and predicted secondary structure class labels are obtained. According to results, new prediction is generated. A new rate is obtained with comparison of the prediction and the correct results.

5.2.3 Randomforest+DeepCNF Model Averaging

The third method we implement for model averaging was combination of Randomforest and DeepCNF methods. The method applied is the same with the first and second method. After execution of each method individually, observed and predicted secondary structure class labels are derived. Firstly, averaging is applied to methods and a new prediction is generated. New accuracy rate is obtained with comparison of the prediction and observed results.

5.2.4 SVM+Randomforest+DeepCNF Model Averaging

Last method for model averaging was consensus of support vector machine, randomforest and deepCNF methods. Our deepCNF method was executed for three different layer individually. So, this method is implemented for each layer separately and also applied for deepCNF with combination of hidden layers.

.

Chapter 6

6.Results

6.1 7-fold Cross Validation Experiment with CB513 Dataset

Firstly, we prepared our dataset to apply the methods. Our dataset for SVM and Random forest has 539 features, and 473 feature for Deep CNF method. Each amino acid has 49 features for SVM and Random forest, while for Deep CNF, it has 43 features. Because of taking window parameters as 5, there will be 11 windows for each amino acid which was 1 for itself, 5 for left neighbor and 5 for right neighbors and totally there will be 11*49=539 feature for SVM and Random forest and 11*43=473 for Deep CNF.

We train all models using the CB513 dataset and applied seven fold cross validation to determine the model hyper-parameters for each training method. . We divide our dataset into 7 equal size subsets for 7-fold cross validation. Each subset is used as test to validate and other 6 subsets are used as training. 10% of train test splits are taken as validation. These were randomly selected at the protein level. After selecting all each fold as test set and evaluate the model, 7 accuracy rates are obtained. Overall accuracy is calculated by taking average of 7 results obtained.

6.2 Optimization Results

We make parameter optimization for all methods we used with cross validation technique. For each method we applied optimization according to their hyper parameters.

After optimization, we re-train in the optimal configuration and get results on validation sets.

6.2.1 SVM Results

Method	**Fold**	**C parameter**	**Gamma parameter**	Q_3
SVM	1	32	0.00195313	84.0
SVM	2	32	0.00195313	81.3
SVM	3	2	0.03125	84.3
SVM	4	32	0.00195313	83.7
SVM		8192	0.000122070	83.7
SVM	5	2	0.0078125	84.5
SVM	6	2	0.03125	82.7
SVM	7	2048	0.000122070	83.2
Overall Accuracy:				**83.4**

Table 6.2.1.1 Q_3 accuracy of the SVM method for parameter optimization on CB513 validation dataset.

Table 6.2.1.1 shows the Q_3 accuracy for optimization results of SVM Method on the CB513 dataset.

For SVM, we have values as below:

C parameters = (2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}, 2^{11}, 2^{13})

Gamma parameters = (2^{-15}, 2^{-13}, 2^{-11}, 2^{-9}, 2^{-7}, 2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3})

After making parameter optimization on validation and train datasets, we get result which has shown in table 6.2.1.1. As shown in table, generally, approximate values were obtained for the parameters in each fold.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM	1	84.0	86.6	74.6	86.0
SVM	2	81.4	84.0	69.0	85.2

SVM	3	84.3	85.6	77.3	87.0
SVM	4	83.8	84.7	79.0	86.0
SVM		83.8	87.0	80.3	84.5
SVM	5	84.5	85.3	73.2	85.5
SVM	6	82.7	83.2	77.0	86.3
SVM	7	83.2	86.6	74.6	86.0
Overall Accuracy:		**83.4**	**85.2**	**75.7**	**85.8**

Table 6.2.1.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM method for training on CB513 validation datasets.

Table 6.2.1.2 shows the Q_3, Q_H, Q_E, Q_L accuracies of the SVM method for re-training optimal configuration on CB513 validation datasets. According to result, prediction for coil and helix is very close and more successful than strand. On validation datasets, accuracies nearly between 83 and 84%.

6.2.2 Random Forest Results

Method	**Fold Number**	**Number of trees**	Q_3
Random Forest	1	375	83.0
Random Forest	2	500	79.9
Random Forest	3	425	82.8
Random Forest	4	500	82.5
Random Forest	5	225	82.8
Random Forest	6	300	80.9
Random Forest	7	100	81.9
Overall Accuracy:			**81.9**

Table 6.2.2.1 Q_3 accuracy of the Random forest method for parameter optimization on CB513 validation dataset.

Table 6.2.2.1 shows the Q_3 accuracy for optimization results of Random forest method on the CB513 dataset. We use values as "5 10 15 20 25 50 75 100 125 150 175 200 225 250 275 300 325 350 375 400 425 450 475 500" for

number of trees. As shown in table, generally, the result is better, when we use larger value for iteration number. Our overall Q_3 accuracy is for random forest optimization is %82.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
Random Forest	1	83.0	84.5	70.8	87.2
Random Forest	2	79.9	82.1	65.7	85.0
Random Forest	3	82.8	83.1	73.4	88.0
Random Forest	4	82.5	82.6	75.1	86.7
Random Forest	5	82.8	85.0	75.8	84.4
Random Forest	6	80.9	81.7	69.4	86.2
Random Forest	7	81.9	82.7	74.3	85.4
Overall Accuracy:		**81.9**	**83.1**	**72.0**	**86.1**

Table 6.2.2.2 Q_3, Q_H, Q_E, Q_L accuracies of the Random Forest method for training on CB513 validation datasets.

Table 6.2.2.2 shows the Q_3, Q_H, Q_E, Q_L accuracies of the Random Forest method for re-training optimal configuration on CB513 validation datasets. According to result, prediction for coil and helix is more successful than prediction of strand.

6.2.3 Deep CNF Results

We applied 7 fold cross validation to all for each Deep CNF model we used using the CB513 dataset. For optimization part, we determined 3 factors with three different numbers for each hyper parameter. We repeat optimization part for 3 different number of hidden layer. We take hidden layer numbers as 3, 4 and 5. The values we used for optimization is as shown below:

Window size: “3, 4, 5”,

Number of nodes: “75, 100, 125”,

Regularization coefficient: “10, 50, 100”

6.2.3.1 Deep CNF Model with 3-Hidden Layers

Method	Number of Hidden Layer	Fold	Window string	Node string	Reg. Coefficient	Q_3
Deep CNF	3	1	3,3,3	100,100,100	50	90.5
Deep CNF	3	2	3,3,3	100,100,100	10	88.8
Deep CNF	3	3	3,3,3	125,125,125	50	92.0
Deep CNF	3	4	4,4,4	100,100,100	50	88.9
Deep CNF	3	5	4,4,4	125,125,125	50	89.7
Deep CNF	3	6	4,4,4	100,100,100	50	91.3
Deep CNF	3	7	4,4,4	100,100,100	100	89.6
Overall Accuracy:						**90.1**

Table 6.2.3.1.1 Q_3 accuracy of the Deep CNF method with 3 hidden layers for parameter optimization on CB513 validation datasets.

Table 6.2.3.1.1 shows the Q_3 accuracy for optimization results of Deep CNF method with 3 hidden layers on the CB513 dataset. We use 3 different hyper parameters to optimize. The best results are usually obtained, when regularization coefficient is set to 50. Also, when we set node number for each hidden layer as 100 and 125 mostly 100, it gives better result rather than node number with 75. Window size is also an important issue around hyper parameters. The best accuracy is achieved when window size is set to 3 or 4. These values are received for 3 hidden layers.

Method	Number of Hidden Layer	Fold Number	Q_3	Q_H	Q_E	Q_L
Deep CNF	3	1	90.5	92.2	89.4	89.7
Deep CNF	3	2	88.8	91.8	85.8	88.2
Deep CNF	3	3	92.0	95.2	91.2	89.6
Deep CNF	3	4	89.0	91.8	87.3	88.0
Deep CNF	3	5	89.7	91.1	87.0	89.9
Deep CNF	3	6	91.2	93.7	88.5	90.1
Deep CNF	3	7	89.6	90.3	83.3	90.1
Overall Accuracy:			**90.1**	**92.3**	**87.5**	**89.3**

Table 6.2.3.1.2 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 3 hidden layers for training on CB513 validation datasets.

Table 6.2.3.1.2 shows the Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF with 3 hidden layers method for re-training optimal configuration on CB513 validation datasets. Successful results are taken when Deep CNF is executed on validation datasets.

6.2.3.2 Deep CNF Model with 4-Hidden Layers

Method	Fold	Number of Hidden Layer	Window string	Node string	Reg. Coefficient	Q_3
Deep CNF	1	4	3,3,3,3	75,75,75,75	100	90.6
Deep CNF	2	4	3,3,3,3	100,100,100,100	10	88.8
Deep CNF	3	4	4,4,4,4	125,125,125,125	50	91.9
Deep CNF	4	4	3,3,3,3	125,125,125,125	50	88.9
Deep	5	4	3,3,3,3	75,75,75,75	10	89.8

CNF						
Deep CNF	6	4	3,3,3,3	125,125,125,125	50	91.3
Deep CNF	7	4	3,3,3,3	75,75,75,75	50	89.6
Overall Accuracy:						**90.2**

Table 6.2.3.2.1 Q_3 accuracy of the DeepCNF method with 4 hidden layers for parameter optimization on CB513 validation dataset.

Table 6.2.3.2.1 shows the Q_3 accuracy for optimization results of Deep CNF method with 4 hidden layers on the CB513 dataset. The best results are usually obtained, when regularization coefficient is set to 50 as 3 hidden layers. When node number is generally set to 75(differently from 3 hidden layers) and 125, the better results are taken. The best accuracy is achieved when window size is set to 3 for six fold and 4 for one fold. These values are obtained for 4 hidden layers.

Method	**Number of Hidden Layer**	**Fold Number**	Q_3	Q_H	Q_E	Q_L
Deep CNF	4	1	90.4	92.4	88.5	89.5
Deep CNF	4	2	88.5	90.9	85.2	88.8
Deep CNF	4	3	91.8	94.7	91.2	89.6
Deep CNF	4	4	89.0	92.2	87.6	87.8
Deep CNF	4	5	89.6	90.6	87.9	89.6
Deep CNF	4	6	91.4	93.8	89.5	90.4
Deep CNF	4	7	89.4	89.6	83.8	91.8
Overall Accuracy:			**90.0**	**91.9**	**87.7**	**89.6**

Table 6.2.3.2.2 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 4 hidden layers for training on CB513 validation datasets.

Table 6.2.3.2.2 shows the Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF with 4 hidden layers method for re-training optimal configuration on CB513 validation datasets. Successful results are taken when Deep CNF is executed on validation datasets the prediction of strands has least success when we compare with other class labels.

6.2.3.3 Deep CNF Model with 5-Hidden Layers

Method	**Fold**	**Number of hidden layer**	**Window string**	**Node string**	**Reg. Coefficient**	Q_3
Deep CNF	1	5	3,3,3,3,3	125,125,125,125,125	50	90.7
Deep CNF	2	5	4,4,4,4,4	125,125,125,125,125	50	88.7
Deep CNF	3	5	4,4,4,4,4	75,75,75,75,75	50	91.8
Deep CNF	4	5	4,4,4,4,4	125,125,125,125,125	50	89.1
Deep CNF	5	5	3,3,3,3,3	125,125,125,125,125	50	89.8
Deep CNF	6	5	3,3,3,3,3	75,75,75,75,75	50	91.4
Deep CNF	7	5	3,3,3,3,3	125,125,125,125,125	50 100	89.5
Overall Accuracy:						**90.1**

Table 6.2.3.3.1 Q_3 accuracy of the DeepCNF method with 5 hidden layers for parameter optimization on CB513 validation datasets.

The Q_3 accuracy for optimization results of Deep CNF method with 5 hidden layers on the CB513 dataset is shown in table 6.2.3.3.1. The best results are usually obtained, when regularization coefficient is set to 50 as in 3 and 4

hidden layer. When node number is generally set to 75(differently from 3 hidden layer) or 125, the better results are taken. The best accuracy is achieved when window size is set to 3 for six fold and 4 for one fold. These values are received for 4 hidden layers.

Method	**Number of Hidden Layer**	**Fold Number**	Q_3	Q_H	Q_E	Q_L
Deep CNF	5	1	90.4	92.2	89.4	89.0
Deep CNF	5	2	88.6	91.1	86.1	88.3
Deep CNF	5	3	91.6	94.8	91.7	89.0
Deep CNF	5	4	88.8	90.8	86.5	88.9
Deep CNF	5	5	89.6	91.5	89.0	88.4
Deep CNF	5	6	91.1	94.0	89.5	89.5
Deep CNF	5	7	89.4	**90.2**	**85.2**	**90.9**
			90.4			
Overall Accuracy:			**90.0**	**91.9**	**87.7**	**89.6**

Table 6.2.3.3.2 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 5 hidden layers for training on CB513 validation datasets.

Table 6.2.3.3.2 shows the Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF with 5 hidden layers method for re-training optimal configuration on CB513 validation datasets. Successful results are taken when Deep CNF is executed on validation datasets.

Overall, as shown from tables, when the number of hidden layers is different, there are not big differences at overall accuracies and good results are taken when Deep CNF is executed on validation datasets.

In literature, there are few studies about deep CNF. For optimization, there is only one study and it is only about optimization of regularization coefficient with 5 fold cross validation [58]. At the end of optimization, they get 73.2% Q_8 accuracy on average.

6.3 Train-Test Results

Optimum parameters that found after parameter optimization are separately applied to all methods for each fold for training. We differently make training on validation dataset as well as test set.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM	1	82.1	85.3	72.0	85.7
SVM	2	82.2	85.0	76.5	83.4
SVM	3	83.0	84.6	75.8	85.5
SVM	4	**82.7**	**84.1**	**75.7**	**85.0**
SVM	4	82.6	84.1	78.1	82.7
SVM	5	82.2	85.0	76.7	83.5
SVM	6	82.5	87.3	78.2	85.8
SVM	7	85.0	85.3	72.0	85.7
Overall Accuracy:		**82.8**	**85.2**	**75.3**	**84.9**

Table 6.3.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM method for training on CB513 test datasets.

Table 6.3.1 shows the Q_3, Q_H, Q_E, Q_L accuracies for training results of SVM Method on the CB513 dataset. As shown in table, generally, accuracies that obtained for each fold have close rates because of similarity values that found for optimum values. Accuracies are approximately 82% on test datasets, except seventh fold. It has nearly 85% accuracy rate at the end of training method. Accuracy of Q_E which is used for strand label of secondary structure has the lowest value compared to other labels for both of test and validation datasets.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
Random Forest	1	81.1	83.2	70.5	86.0
Random Forest	2	81.5	82.5	75.2	84.3

Random Forest	3	81.8	82.3	72.7	86.5
Random Forest	4	81.5	82.8	73.3	84.8
Random Forest	5	81.2	82.0	76.1	83.2
Random Forest	6	82.0	82.8	74.5	85.0
Random Forest	7	83.4	86.0	73.7	85.4
Overall Accuracy:		**81.8**	**83.2**	**73.7**	**85.0**

Table 6.3.2 Q_3, Q_H, Q_E, Q_L accuracies of the Random forest method for training on CB513 test datasets.

Table 6.3.2 shows the Q_3, Q_H, Q_E, Q_L accuracies for training results of Random forest Method on the CB513 dataset. As shown in table, generally, accuracies that obtained for each fold have close rates. Overall accuracy is 81.8% for test set.

Method	**Fold**	**Number of Hidden Layer**	Q_3	$. Q_H$	Q_E	Q_L
Deep CNF	1	3	81.7	86.7	73.6	83.1
Deep CNF	2	3	81.7	86.0	76.6	81.4
Deep CNF	3	3	83.2	84.7	75.4	85.4
Deep CNF	4	3	82.4	86.1	76.6	82.6
Deep CNF	5	3	81.7	85.4	78.3	81.2

Deep CNF	6	3	82.1	85.3	78.3	82.5
Deep CNF	7	3	84.2	87.7	77.1	84.9
Overall Accuracy:			**82.4**	**86.0**	**76.5**	**83.0**

Table 6.3.3 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 3 hidden layers for training on CB513 test dataset.

Table 6.3.3 shows the Q_3, Q_H, Q_E, Q_L accuracies for training results of Deep CNF Method with 3 hidden layers on the CB513 dataset. The overall accuracy is obtained as 82.4% for test set.

Method	**Fold**	**Number of Hidden Layer**	Q_3	$. Q_H$	Q_E	Q_L
Deep CNF	1	4	81.9	85.2	73.5	85
Deep CNF	2	4	81.9	85.4	76.5	82.8
Deep CNF	3	4	82.9	85.4	76.0	84.1
Deep CNF	4	4	82.3	85.9	76.5	82.8
Deep CNF	5	4	81.6	84.8	77.7	81.6
Deep CNF	6	4	82.3	84.6	77.1	83.4
Deep CNF	7	4	84.6	88.2	78.2	84.1
Overall Accuracy:			**82.5**	**85.6**	**76.5**	**83.4**

Table 6.3.4 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 4 hidden layers for training on CB513 test dataset.

Table 6.3.4 shows the Q_3, Q_H, Q_E, Q_L accuracies for training results of Deep CNF Method with 4 hidden layers on the CB513 dataset. The overall accuracy is obtained as 82.5% for test set.

Method	**Fold**	**Number of Hidden Layer**	Q_3	$. Q_H$	Q_E	Q_L
Deep CNF	1	5	82.0	86.2	73.1	84.4
Deep CNF	2	5	81.7	86.6	77.0	81.5
Deep CNF	3	5	83.0	85.7	75.6	84.5
Deep CNF	4	5	82.0	86.8	76.3	81.3
Deep CNF	5	5	81.7	85.5	77.7	81.3
Deep CNF	6	5	82.2	85.6	77.0	83.2
Deep CNF	7	5	84.5	88.3	77.4	84.3
Overall Accuracy:			**82.5**	**86.4**	**76.3**	**82.9**

Table 6.3.5 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF method with 5 hidden layers for training on CB513 validation and test dataset.

Table 6.3.5 shows the Q_3, Q_H, Q_E, Q_L accuracies for training results of Deep CNF Method with 5 hidden layers on the CB513 dataset. The overall accuracy is obtained as 82.5% for test set.

6.4 Ensemble Methods Results

6.4.1 Deep CNF Hidden Layer Combination

The first ensemble method we used is for deep CNF method. For this, we added also hidden layer number to configuration. For each fold, we find the configuration that gives optimum accuracy from the combination of 3, 4 and 5 hidden layer train validate results. We make separately a train test for each fold with the configuration found (including the hidden layer number). Then, we combined the results and make a new prediction rate.

The results for train-validate and train-test is as shown below.

Method	**Fold**	**Number of hidden layer**	**Window string**	**Node string**	**Reg. Coefficient**	Q_3	Q_H	Q_E	Q_L
Deep CNF	1	5	3,3,3,3,3	125,125,125, 125,125	50	90.7	90.2	89.4	89.0
Deep CNF	2	4	3,3,3,3	100,100,100, 100	10	88.8	91.0	85.0	88.8
		3	3,3,3	100,100,100	10	88.8	91.8	85.0	88.0
Deep CNF	3	3	3,3,3	125,125,125	50	92.0	95.0	91.2	89.0
Deep CNF	4	5	4,4,4,4,4	125,125,125, 125,125	50	89.0	90.0	86.5	89.0
Deep CNF	5	5	3,3,3,3,3	125,125,125, 125,125	50	89.8	91.5	89.0	88.4
Deep CNF	6	5	3,3,3,3,3	75,75,75,75,7 5	50	91.4	94.0	89.5	89.5
Deep CNF	7	3	4,4,4	100,100,100	100	89.6	90.3	83.3	91.2
Overall Accuracy:						**90.2**	**91.8**	**87.7**	**89.3**

Table 6.4.1.1 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF hidden layer ensemble method for train validation on CB513 dataset.

Table 6.4.1.1 shows the result for train-validate part. After applying hidden layer number to configuration, accuracy gets 90.2%. The accuracy for each hidden layer number without combining hidden layers was 90.1%. So result improved only 0.1% for train-validate result because of close accuracy

rate of each hidden layer number. If there were big differences in each hidden layer for accuracy rate, result would improve much more.

Method	**Fold**	**Number of hidden layer**	**Window string**	**Node string**	**Reg. coefficient**	Q_3	Q_H	Q_E	Q_L
Deep CNF	1	5	3,3,3,3,3	125,125,125, 125,125	50	82.0	86.2	73.0	84.5
Deep CNF	**2**	**4**	**3,3,3,3**	**100,100,100, 100**	**10**	**81.89**	**85.5**	**76.6**	**82.8**
		3	3,3,3	100,100,100	10	81.7	86.0	76.0	81.4
Deep CNF	3	3	3,3,3	125,125,125	50	83.2	84.7	75.4	85.4
Deep CNF	4	5	4,4,4,4,4	125,125,125, 125,125	50	82.0	86.8	76.3	81.4
Deep CNF	5	5	3,3,3,3,3	125,125,125, 125,125	50	81.7	85.5	77.7	81.3
Deep CNF	6	5	3,3,3,3,3	75,75,75,75,7 5	50	82.2	85.7	77.0	83.2
Deep CNF	7	3	4,4,4	100,100,100	100	84.2	87.7	77.2	85.0
Overall Accuracy:						**82.5**	**86.1**	**76.2**	**83.3**

Table 6.4.1.2 Q_3, Q_H, Q_E, Q_L accuracies of the Deep CNF hidden layer ensemble method for train test on CB513 dataset.

Table 6.4.1.2 shows the result for train-test part. After the best configuration that gives best accuracy rate is found, the optimum parameters configuration is applied to train test for each fold. The overall accuracy for Q_3 we get was 82.5%. It is the same with the results for without applying hidden layer combination.

6.4.2 Model Averaging Results on Validation and Test Sets

6.4.2.1 Support Vector Machines and Random Forest

Firstly, we applied model averaging method to Support vector machines and random forest methods. For this, we prepared predicted probability scores for 3 classes for each amino acid in each protein. Then, we took the average of these scores and then got the class label that gave the maximum score. At the end of averaging, we calculate the new accuracy scores.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
SVM+RF	1	84.0	85.8	73.8	87.5
SVM+RF	2	83.0	83.8	67.8	85.6
SVM+RF	3	83.0	85.0	76.0	87.8
SVM+RF	4	83.0	83.7	77.5	86.7
SVM+RF	5	83.0	86.2	78.4	85.0
SVM+RF	6	83.0	84.2	71.4	86.0
SVM+RF	7	83.0	83.4	76.9	86.7
Overall Accuracy:		**83.2**	**84.6**	**74.5**	**86.5**

Table 6.4.2.1.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Random forest model averaging ensemble method for validation on CB513 dataset.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
SVM+RF	1	82.0	84.4	71.0	86.0
SVM+RF	2	82.2	84.5	76.0	84.0
SVM+RF	3	82.4	84.0	75.0	86.0
SVM+RF	4	82.5	83.7	75.0	85.0
SVM+RF	5	82.4	83.3	78.0	83.0
SVM+RF	6	82.4	84.3	76.0	85.0
SVM+RF	7	83.0	86.9	76.0	86.0
Overall Accuracy:		**82.6**	**84.5**	**75.5**	**85.0**

Table 6.4.2.1.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Random forest model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.1.1 and 6.4.2.1.2 shows the result of combination of support vector machines and random forests. After making model averaging on SVM and RF, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.2% for validation dataset and 82.6% for test dataset.

6.4.2.2 Support Vector Machines and Deep CNF

Secondly, model averaging is applied to support vector machines and deep convolutional neural field method. Here, we used deep CNF method that has a combination with hidden layer numbers. The same procedure is implemented. The predictive probability score of 3 classes for each amino acid in each protein is prepared according to result of individual classifier execution. Then, we obtained the class label that gave the maximum value by taking average of these scores. According to results, overall accuracy score for this ensemble method is calculated.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF	1	84.0	87.0	75.2	86.6
SVM+DeepCNF	2	83.0	85.3	70.8	84.3
SVM+DeepCNF	3	83.0	85.3	77.6	87.0
SVM+DeepCNF	4	84.0	85.5	79.8	85.5
SVM+DeepCNF	5	84.0	87.5	80.3	84.0
SVM+DeepCNF	6	83.0	85.4	73.2	85.0
SVM+DeepCNF	7	83.0	83.8	77.5	85.6
Overall Accuracy:		**83.5**	**85.7**	**76.3**	**85.4**

Table 6.4.2.2.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF	1	81.9	86.0	73.2	85.5
SVM+DeepCNF	2	82.2	85.4	76.8	83.5
SVM+DeepCNF	3	82.4	85.0	76.0	86.0
SVM+DeepCNF	4	82.5	85.7	76.5	83.3
SVM+DeepCNF	5	82.4	84.6	78.0	82.4
SVM+DeepCNF	6	82.4	85.4	77.2	84.0
SVM+DeepCNF	7	82.8	87.5	77.8	85.6
Overall Accuracy:		**83.0**	**85.7**	**76.5**	**84.3**

Table 6.4.2.2.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.2.1 and 6.4.2.2.2 shows the result of ensemble of support vector machines and deep CNF with hidden layer combination. After making model averaging on SVM and Deep CNF with hidden layer combination, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.5% for validation dataset and 83.0% for test dataset.

6.4.2.3 Support Vector Machines and Deep CNF with 3 Hidden Layers

Third method is for support vector machine and deep convolutional neural field with 3 hidden layers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(3)	1	84.0	87.0	75.7	86.0

SVM+DeepCNF(3)	2	83.0	85.0	70.8	84.3
SVM+DeepCNF(3)	3	83.0	85.7	77.0	86.8
SVM+DeepCNF(3)	4	83.0	84.6	79.8	85.5
SVM+DeepCNF(3)	5	84.0	87.5	80.3	84.0
SVM+DeepCNF(3)	6	83.0	85.3	71.7	85.6
SVM+DeepCNF(3)	7	83.0	84.0	77.5	85.7
Overall Accuracy:		**83.4**	**85.6**	**76.0**	**85.4**

Table 6.4.2.3.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 3 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(3)	1	82.0	86.0	73.3	84.8
SVM+DeepCNF(3)	2	82.0	85.6	77.0	83.2
SVM+DeepCNF(3)	3	83.0	85.0	76.0	86.0
SVM+DeepCNF(3)	4	83.0	85.6	76.5	84.2
SVM+DeepCNF(3)	5	83.0	85.0	78.3	82.3
SVM+DeepCNF(3)	6	83.0	85.3	77.8	83.5
SVM+DeepCNF(3)	7	83.0	87.6	77.8	85.6
Overall Accuracy:		**83.0**	**85.8**	**76.6**	**84.2**

Table 6.4.2.3.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 3 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.3.1 and 6.4.2.3.2 shows the result of combination of support vector machines and deep CNF with 3 hidden layers. After making model averaging on SVM and Deep CNF with 3 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.4% for validation dataset and 83.0% for test dataset.

6.4.2.4 Support Vector Machines and Deep CNF with 4 Hidden Layers

Fourth method is for support vector machine and deep convolutional neural field with 4 hidden layers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(4)	1	84.0	87.0	75.2	86.6
SVM+DeepCNF(4)	2	83.0	84.5	69.4	85.3
SVM+DeepCNF(4)	3	83.0	85.3	77.6	87.0
SVM+DeepCNF(4)	4	83.0	85.5	79.8	85.5
SVM+DeepCNF(4)	5	84.0	87.0	80.7	84.0
SVM+DeepCNF(4)	6	83.0	85.0	73.0	85.0
SVM+DeepCNF(4)	7	83.0	83.7	77.7	85.7
Overall Accuracy:		**83.3**	**85.4**	**76.2**	**85.6**

Table 6.4.2.4.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 4 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(4)	1	82.0	85.4	73.0	85.5
SVM+DeepCNF(4)	2	82.0	85.4	76.8	83.5
SVM+DeepCNF(4)	3	83.0	85.0	76.2	85.5
SVM+DeepCNF(4)	4	83.0	85.3	76.2	84.5
SVM+DeepCNF(4)	5	82.0	84.5	77.8	82.7
SVM+DeepCNF(4)	6	83.0	85.0	77.4	84.0
SVM+DeepCNF(4)	7	83.0	88.0	78.4	85.4
Overall Accuracy:		**82.6**	**85.6**	**76.5**	**84.4**

Table 6.4.2.4.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 4 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.4.1 and 6.4.2.4.2 shows the result of combination of support vector machines and deep CNF with 4 hidden layers. After making model averaging on SVM and Deep CNF with 4 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.3% for validation dataset and 82.6% for test dataset.

6.4.2.5 Support Vector Machines and Deep CNF with 5 Hidden Layer

Fifth method is for support vector machine and deep convolutional neural field with 5 hidden layers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(5)	1	84.0	87.0	75.3	85.8
SVM+DeepCNF(5)	2	83.0	84.5	69.8	85.3
SVM+DeepCNF(5)	3	83.0	85.4	78.0	87.0
SVM+DeepCNF(5)	4	83.0	85.2	79.8	85.0
SVM+DeepCNF(5)	5	84.0	88.0	80.3	83.5
SVM+DeepCNF(5)	6	83.0	85.5	73.2	85.0
SVM+DeepCNF(5)	7	83.0	83.8	77.5	85.6
Overall Accuracy:		**83.3**	**85.6**	**76.3**	**85.3**

Table 6.4.2.5.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 5 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+DeepCNF(5)	1	82.0	86.0	73.2	85.5
SVM+DeepCNF(5)	2	82.0	85.8	77.2	83.0
SVM+DeepCNF(5)	3	83.0	85.4	76.3	85.7
SVM+DeepCNF(5)	4	83.0	85.7	76.5	83.3

SVM+DeepCNF(5)	5	82.0	84.7	78.0	82.4
SVM+DeepCNF(5)	6	83.0	85.4	77.2	84.0
SVM+DeepCNF(5)	7	83.0	87.8	78.0	85.0
Overall Accuracy:		**82.6**	**85.8**	**76.6**	**84.1**

Table 6.4.2.5.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM and Deep CNF with 5 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.5.1 and 6.4.2.5.2 shows the result of combination of support vector machines and deep CNF with 5 hidden layers. After making model averaging on SVM and Deep CNF with 5 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.3% for validation dataset and 82.6% for test dataset.

6.4.2.6 Random Forest and Deep Convolutional Neural Field

Sixth method is for combination of random forest and deep convolutional neural field. Average scores of predictive probability scores for 3 classes for each amino acid in each protein are obtained from random forest and deep CNF classifiers results. Then, according to class labels that have maximum score the overall accuracy is calculated.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF	1	84.0	86.0	73.0	87.0
RF+DeepCNF	2	83.0	85.7	69.5	84.4
RF+DeepCNF	3	83.0	84.7	75.6	87.8
RF+DeepCNF	4	83.0	84.7	78.2	86.0
RF+DeepCNF	5	83.0	86.5	78.8	84.4
RF+DeepCNF	6	83.0	84.0	71.5	85.3
RF+DeepCNF	7	83.0	83.7	76.0	83.0
Overall Accuracy:		**83.1**	**85.0**	**74.6**	**85.8**

Table 6.4.2.6.1 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF	1	82.0	85.3	72.5	85.7
RF+DeepCNF	2	82.0	84.4	76.3	84.0
RF+DeepCNF	3	82.0	84.5	75.0	86.0
RF+DeepCNF	4	82.0	85.5	75.0	83.6
RF+DeepCNF	5	82.0	84.0	77.4	82.6
RF+DeepCNF	6	82.0	85.0	75.8	84.6
RF+DeepCNF	7	87.4	75.8	85.8	83.0
Overall Accuracy:		**82.8**	**85.2**	**75.5**	**84.6**

Table 6.4.2.6.2 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.6.1 and 6.4.2.6.2 shows the result of ensemble of random forests and deep CNF with hidden layer combination. After making model averaging on RF and Deep CNF with hidden layer combination, new accuracies rates are found. The overall accuracy for Q_3 we get was 83.1% for validation dataset and 82.8% for test dataset.

6.4.2.7 Random Forest and Deep CNF with 3 Hidden Layers

Seventh method is for random forest and deep convolutional neural field with 3 hidden layers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	Fold Number	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(3)	1	82.0	86.3	73.0	86.0
RF+DeepCNF(3)	2	82.0	85.7	69.5	84.0
RF+DeepCNF(3)	3	82.0	85.0	75.5	87.6

RF+DeepCNF(3)	4	82.0	84.0	78.2	86.0
RF+DeepCNF(3)	5	82.0	86.5	78.8	84.4
RF+DeepCNF(3)	6	82.0	83.6	70.0	85.7
RF+DeepCNF(3)	7	83.0	84.0	76.0	85.4
Overall Accuracy:		**82.1**	**85.0**	**74.4**	**85.6**

Table 6.4.2.7.1 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 3 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(3)	1	82.0	86.0	73.3	84.8
RF+DeepCNF(3)	2	82.0	85.6	76.8	83.2
RF+DeepCNF(3)	3	82.0	85.0	76.0	86.0
RF+DeepCNF(3)	4	82.0	85.6	76.5	84.2
RF+DeepCNF(3)	5	82.0	85.0	78.3	82.3
RF+DeepCNF(3)	6	82.0	85.3	77.8	83.5
RF+DeepCNF(3)	7	83.0	87.5	77.8	85.6
Overall Accuracy:		**82.1**	**85.8**	**76.6**	**84.2**

Table 6.4.2.7.2 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 3 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.7.1 and 6.4.2.7.2 shows the result of combination of random forests and deep CNF with 3 hidden layers. After making model averaging on RF and Deep CNF with 3 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.1% for validation and test dataset.

6.4.2.8 Random Forest and Deep CNF with 4 Hidden Layers

Eighth method is for combination of Radom forest and deep convolutional neural field with 4 hidden layers classifiers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(4)	1	82.0	86.0	73.0	87.0
RF+DeepCNF(4)	2	82.0	84.4	67.7	85.2
RF+DeepCNF(4)	3	82.0	84.7	75.6	87.8
RF+DeepCNF(4)	4	82.0	84.7	78.2	86.0
RF+DeepCNF(4)	5	82.0	86.2	79.3	84.4
RF+DeepCNF(4)	6	82.0	83.8	72.0	85.2
RF+DeepCNF(4)	7	83.0	83.7	76.0	85.8
Overall Accuracy:		**82.1**	**84.8**	**74.5**	**86.0**

Table 6.4.2.8.1 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 4 hidden layers combination model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(4)	1	82.0	85.4	73.0	85.5
RF+DeepCNF(4)	2	82.0	85.4	76.8	83.5
RF+DeepCNF(4)	3	82.0	85.0	76.2	85.5
RF+DeepCNF(4)	4	82.0	85.3	76.2	84.5
RF+DeepCNF(4)	5	84.5	77.8	82.7	84.5
RF+DeepCNF(4)	6	85.0	77.4	84.0	85.0
RF+DeepCNF(4)	7	88.0	78.4	85.3	88.0
Overall Accuracy:		**82.8**	**85.6**	**76.5**	**84.4**

Table 6.4.2.8.2 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 4 hidden layers combination model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.8.1 and 6.4.2.8.2 shows the result of combination of random forests and deep CNF with 4 hidden layers. After making model averaging on RF and Deep CNF with 4 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.1% for validation dataset and 82.8% for test dataset.

6.4.2.9 Random Forest and Deep CNF with 5 Hidden Layers

Ninth method is for combination random forest and deep convolutional neural field with 5 hidden layers classifiers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(5)	1	82.2	86.0	74.0	86.3
RF+DeepCNF(5)	2	82.3	84.2	68.0	85.3
RF+DeepCNF(5)	3	82.2	84.8	76.2	87.7
RF+DeepCNF(5)	4	82.2	84.8	77.5	85.0
RF+DeepCNF(5)	5	82.3	87.0	79.2	84.2
RF+DeepCNF(5)	6	82.3	84.0	71.5	85.3
RF+DeepCNF(5)	7	83.3	83.7	760	85.8
Overall Accuracy:		**82.5**	**85.0**	**74.6**	**85.6**

Table 6.4.2.9.1 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 5 hidden layers combination model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
RF+DeepCNF(5)	1	82.2	86.0	73.2	85.5
RF+DeepCNF(5)	2	82.2	85.8	77.2	83.0
RF+DeepCNF(5)	3	82.3	85.4	76.3	85.7
RF+DeepCNF(5)	4	82.3	85.7	76.5	83.3
RF+DeepCNF(5)	5	82.1	84.6	78.0	82.4
RF+DeepCNF(5)	6	82.4	85.4	77.2	84.0
RF+DeepCNF(5)	7	83.0	87.8	78.0	85.0
Overall Accuracy:		**82.5**	**85.9**	**76.6**	**84.1**

Table 6.4.2.9.2 Q_3, Q_H, Q_E, Q_L accuracies of the RF and Deep CNF with 5 hidden layers combination model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.9.1 and 6.4.2.9.2 shows the result of combination of random forests and deep CNF with 5 hidden layers. After making model averaging on RF and Deep CNF with 5 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.5% for validation test dataset.

6.4.2.10 Support Vector Machines, Random Forest and Deep CNF

Tenth method is for combination of support vector machines, random forest and deepCNF classifiers. Here, we used three classifiers as an ensemble method. The predictive probability score of 3 classes for each amino acid in each protein is prepared according to result of individual classifier execution. Then, we obtained the class label that gave the maximum value by taking average of these scores. According to results, overall accuracy score for this ensemble method is calculated.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF	1	82.0	86.3	74.2	87.0

SVM+RF+DeepCNF	2	82.0	85.0	69.7	85.4
SVM+RF+DeepCNF	3	83.0	85.2	76.6	87.7
SVM+RF+DeepCNF	4	83.0	84.6	78.2	86.0
SVM+RF+DeepCNF	5	82.0	86.8	79.3	84.6
SVM+RF+DeepCNF	6	83.0	84.2	72.2	85.5
SVM+RF+DeepCNF	7	83.0	83.7	77.0	86.2
Overall Accuracy:		**82.6**	**85.1**	**75.3**	**86.0**

Table 6.4.2.10.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with hidden layer combination model averaging ensemble method for validation on CB513 dataset.

Method	**Fold Number**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF	1	82.0	85.5	72.5	86.0
SVM+RF+DeepCNF	2	82.0	85.0	76.6	84.0
SVM+RF+DeepCNF	3	83.0	84.5	75.0	86.4
SVM+RF+DeepCNF	4	83.0	85.0	76.0	84.4
SVM+RF+DeepCNF	5	82.0	84.0	77.7	82.5
SVM+RF+DeepCNF	6	83.0	85.0	76.7	84.5
SVM+RF+DeepCNF	7	83.0	87.4	77.0	86.0
Overall Accuracy:		**82.6**	**85.3**	**76.0**	**84.8**

Table 6.4.2.10.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with hidden layer combination model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.10.1 and 6.4.2.10.2 shows the result of ensemble of SVM, random forests and deep CNF with hidden layer combination. After making

model averaging on SVM, RF and Deep CNF with hidden layer combination, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.6% for validation and test dataset.

6.4.2.11 Support Vector Machine, Random Forest and Deep CNF with 3 Hidden Layers

Eleventh method is for combination of support vector machines, random forest and deep convolutional neural field with 3 hidden layers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(3)	1	82.0	86.3	74.0	86.6
SVM+RF+DeepCNF(3)	2	82.0	85.0	69.7	85.4
SVM+RF+DeepCNF(3)	3	83.0	85.2	76.4	87.4
SVM+RF+DeepCNF(3)	4	83.0	84.3	78.3	86.0
SVM+RF+DeepCNF(3)	5	82.0	86.8	79.3	84.6
SVM+RF+DeepCNF(3)	6	82.0	84.6	71.2	86.2
SVM+RF+DeepCNF(3)	7	83.0	84.0	77.0	86.2
Overall Accuracy:		**82.4**	**85.2**	**75.1**	**86.0**

Table 6.4.2.11.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 3 hidden layers model averaging ensemble method for test and validation on CB513 dataset.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(3)	1	82.0	85.5	72.7	85.6
SVM+RF+DeepCNF(3)	2	82.0	85.0	76.5	84.0

SVM+RF+DeepCNF(3)	3	83.0	84.5	75.0	86.4
SVM+RF+DeepCNF(3)	4	83.0	85.0	75.6	85.0
SVM+RF+DeepCNF(3)	5	82.0	84.0	77.8	83.0
SVM+RF+DeepCNF(3)	6	82.0	85.0	77.0	84.3
SVM+RF+DeepCNF(3)	7	83.0	87.4	77.0	86.0
Overall Accuracy:		**82.4**	**85.3**	**76.0**	**84.8**

Table 6.4.2.11.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 3 hidden layers model averaging ensemble method for test and validation on CB513 dataset.

Table 6.4.2.11.1 and 6.4.2.11.2 shows the result of combination of SVM, random forests and deep CNF with 3 hidden layers. After making model averaging on SVM, RF and Deep CNF with 3 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.4% for validation and test dataset.

6.4.2.12 Support Vector Machine, Random Forest and Deep CNF with 4 Hidden Layers

Twelfth method is for combination of Support Vector Machines, Radom forest and deep convolutional neural field with 4 hidden layers classifiers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(4)	1	82.0	86.3	74.2	87.0
SVM+RF+DeepCNF(4)	2	82.0	84.4	68.7	85.7
SVM+RF+DeepCNF(4)	3	83.0	85.2	76.6	87.7
SVM+RF+DeepCNF(4)	4	83.0	84.6	78.2	86.0

SVM+RF+DeepCNF(4)	5	82.0	86.7	79.7	84.6
SVM+RF+DeepCNF(4)	6	83.0	84.5	72.4	85.6
SVM+RF+DeepCNF(4)	7	83.0	83.7	77.3	86.0
Overall Accuracy:		**82.6**	**85.0**	**75.3**	**86.0**

Table 6.4.2.12.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 4 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(4)	1	82.0	85.0	72.6	86.2
SVM+RF+DeepCNF(4)	2	82.0	85.0	76.6	84.0
SVM+RF+DeepCNF(4)	3	83.0	84.5	75.7	86.0
SVM+RF+DeepCNF(4)	4	83.0	85.0	75.5	85.0
SVM+RF+DeepCNF(4)	5	82.0	83.7	78.0	83.0
SVM+RF+DeepCNF(4)	6	83.0	84.6	76.7	84.8
SVM+RF+DeepCNF(4)	7	83.0	87.7	77.4	85.6
Overall Accuracy:		**82.6**	**85.2**	**76.0**	**85.0**

Table 6.4.2.12.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 4 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.12.1 and 6.4.2.12.2 shows the result of combination of SVM, random forests and deep CNF with 4 hidden layers. After making model averaging on RF and Deep CNF with 4 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.6% for validation and test dataset.

6.4.2.13 Support Vector Machine, Random Forest and Deep CNF with 5 Hidden Layers

Thirteenth method is for combination of Support Vector Machines, random forest and deep convolutional neural field with 5 hidden layers classifiers. According to results of classifiers individually, the probability scores are arranged and the average of these scores is taken. Then class label which gave maximum score is obtained and prediction results are determined.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(5)	1	82.0	86.4	74.0	86.8
SVM+RF+DeepCNF(5)	2	82.0	84.2	69.0	85.6
SVM+RF+DeepCNF(5)	3	83.0	85.0	76.8	87.7
SVM+RF+DeepCNF(5)	4	83.0	84.7	78.2	86.0
SVM+RF+DeepCNF(5)	5	82.0	87.1	79.8	84.6
SVM+RF+DeepCNF(5)	6	83.0	84.2	72.1	85.5
SVM+RF+DeepCNF(5)	7	83.0	83.7	76.9	86.2
Overall Accuracy:		**82.6**	**85.0**	**75.3**	**86.0**

Table 6.4.2.13.1 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 5 hidden layers model averaging ensemble method for validation on CB513 dataset.

Method	**Fold**	Q_3	Q_H	Q_E	Q_L
SVM+RF+DeepCNF(5)	1	82.0	85.5	72.5	86.0
SVM+RF+DeepCNF(5)	2	82.0	85.5	76.8	83.8
SVM+RF+DeepCNF(5)	3	83.0	85.0	75.6	86.4
SVM+RF+DeepCNF(5)	4	83.0	85.0	76.0	84.4
SVM+RF+DeepCNF(5)	5	82.0	84.0	77.7	82.5
SVM+RF+DeepCNF(5)	6	83.0	85.0	76.7	84.5
SVM+RF+DeepCNF(5)	7	83.0	87.6	77.2	85.6
Overall Accuracy:		**82.6**	**85.4**	**76.0**	**84.7**

Table 6.4.2.13.2 Q_3, Q_H, Q_E, Q_L accuracies of the SVM, RF and Deep CNF with 5 hidden layers model averaging ensemble method for test on CB513 dataset.

Table 6.4.2.13.1 and 6.4.2.13.2 shows the result of combination of SVM, random forests and deep CNF with 5 hidden layers. After making model averaging on SVM, RF and Deep CNF with 5 hidden layers, new accuracies rates are found. The overall accuracy for Q_3 we get was 82.6% for validation and test dataset.

Method	Q_3	Q_H	Q_E	Q_L
SVM	**83.4**	85.2	75.7	85.8
RF	**82.0**	83.0	72.0	86.0
$\mathbf{DeepCNF_3}$	**89.8**	92.3	87.5	89.3
$\mathbf{DeepCNF_4}$	**90.0**	91.9	87.7	89.6
$\mathbf{DeepCNF_5}$	**90.1**	92.1	88.2	89.1
$\mathbf{DeepCNF_{3,4,5}}$	**90.2**	91.8	87.7	89.3
SVM+RF	**83.2**	84.6	74.5	86.5
SVM+$\mathbf{DeepCNF_{3,4,5}}$	**83.5**	85.7	76.3	85.4
SVM+$\mathbf{DeepCNF_3}$	**82.9**	85.0	74.4	85.6
SVM+$\mathbf{DeepCNF_4}$	**83.5**	85.4	76.2	85.6
SVM+$\mathbf{DeepCNF_5}$	**83.4**	85.6	76.3	85.3
RF+$\mathbf{DeepCNF_{3,4,5}}$	**83.0**	85.0	74.6	85.8
RF+$\mathbf{DeepCNF_3}$	**82.9**	85.0	74.4	85.6
RF+$\mathbf{DeepCNF_4}$	**83.0**	84.8	74.5	86.0
RF+$\mathbf{DeepCNF_5}$	**82.9**	85.0	74.6	85.6
SVM+RF+$\mathbf{DeepCNF_{3,4,5}}$	**83.4**	85.0	75.3	86.0
SVM+RF+$\mathbf{DeepCNF_3}$	**83.3**	85.2	75.0	86.0

SVM+RF+DeepCNF$_4$	**83.3**	85.0	75.3	86.0
SVM+RF+DeepCNF$_5$	**83.3**	85.0	75.3	86.0

Table 6.4.2.14.1 Q_3, Q_H, Q_E, Q_L accuracies of the all methods used for test and validation on CB513 dataset.

Method	Q_3	Q_H	Q_E	Q_L
SVM	**82.8**	85.0	76.0	84.5
RF	**81.8**	83.2	73.7	85.0
DeepCNF$_3$	**82.4**	86.0	76.5	83.0
DeepCNF$_4$	**82.5**	85.6	76.5	83.4
DeepCNF$_5$	**82.5**	86.4	76.3	82.9
DeepCNF$_{3,4,5}$	**82.5**	86.1	76.2	83.3
SVM+RF	**82.6**	84.5	75.5	85.0
SVM+DeepCNF$_{3,4,5}$	**83.0**	85.7	76.5	84.3
SVM+DeepCNF$_3$	**82.8**	85.8	76.6	84.2
SVM+DeepCNF$_4$	**83.0**	85.6	76.5	84.4
SVM+DeepCNF$_5$	**83.0**	86.0	76.6	84.0
RF+DeepCNF$_{3,4,5}$	**82.8**	85.2	75.5	84.6
RF+DeepCNF$_3$	**82.8**	85.8	76.6	84.2
RF+DeepCNF$_4$	**82.8**	85.6	76.5	84.4
RF+DeepCNF$_5$	**82.7**	85.9	76.6	84.0
SVM+RF+DeepCNF$_{3,4,5}$	**83.0**	85.3	76.0	84.8
SVM+RF+DeepCNF$_3$	**83.0**	85.3	76.0	84.8
SVM+RF+DeepCNF$_4$	**83.0**	85.2	76.0	85.0
SVM+RF+DeepCNF$_5$	**83.0**	85.4	76.0	84.7

Table 6.4.2.14.2 Q_3, Q_H, Q_E, Q_L accuracies of the all methods used for test and validation on CB513 dataset.

Table 6.4.2.14.1 and 6.4.2.14.2 shows the accuracies we get at the end of all classifiers we used for test and validation dataset.

In our thesis, our aim was to improve classifiers performance by applying ensemble methods. We have a little improvement by combining classifiers but not much. Also, there was no increase on validation datasets when ensemble methods are applied.

CHAPTER 7

CONCLUSION

Bioinformatics is analyzing and processing of biological information with computer systems. It develops techniques of biological storage and depositing. Then, it organizes and analyzes them. Bioinformatics has lots of issues waiting to be solved. One of these problems is protein structure prediction. Protein structure prediction is one of the most important goals of bioinformatics and theoretical chemistry.

Protein structure prediction can be defined as the calculation of the three-dimensional structure of a given amino acid sequence. For this, it is necessary to determine the coordinates of the three dimensional ligament of all the atoms in the protein molecule. It is an effective and efficient approach to estimate protein structure by methods of calculation when experimental methods are inadequate. In addition, since there is a relationship between the protein's structure and its function, the correct prediction of protein yields important clues for protein function. By understanding protein function, diseases can be diagnosed and new drugs can be designed.

It is difficult to directly estimate the 3-D structure of a protein. The possibility of estimating the three-dimensional structure is directly related to the improvements of the one-dimensional structure estimate. So, improvements in one-dimensional structure estimates will allow direct estimation of three-dimensional structure estimates more accurately. In this way, the amount of success in methods of estimating protein function will increase and successful drug and enzyme design will be possible.

Although lots of researches on protein structure prediction have been handled, the structure prediction problem has not been solved yet. For this problem, lots of methods have been used. In addition individual methods, ensemble methods have begun to be used. According to results of researches in recent years, ensemble methods are more successful than individual classifier. Ensemble methods are formed from combination of classifiers. It benefits from the results of all classifiers used.

In our thesis, we have aimed to optimize classifiers for protein secondary structure prediction and show importance and improvement of ensemble methods when applied to datasets. We used distant templates in constructing structural profiles matrix setting the percentage of sequence identity threshold to 20% (i.e. removing templates having greater similarity than this threshold to query). Therefore our results are obtained in the most difficult setting. In previous studies that has similar with our terms approximately 80% - 83% success has been observed. For our study, we take nearly 83% accuracy rate for our ensemble models execution. There is no great success when compared with individual methods. The rate of increase in success is small. It is nearly between 1-2%. But, these results show that our general results are better when compared to most of previous studies in the literature and our models achieved accuracy comparable state of the art. As a future work we are planning to repeat the optimization experiments for other difficulty levels and for dihedral angle class and solvent accessibility class predictions as well.

BIBLIOGRAPHY

[1] Protein structure prediction: https://en.wikipedia.org/wiki/Protein_structure_prediction

[2] D. T. Jones, "Protein structure prediction in genomics," *Briefings in Bioinformatics*, vol. 2, no. 2, pp. 111–125, 2001.

[3] J. Zeng, S. Zhu, H. Yan, "Towards accurate human promoter recognition: a review of currently used sequence features and classification methods," *Briefings in Bioinformatics*, vol. 10, no. 5, pp. 498–508, 2009.

[4] G. Bologna, and R. D. Appel, "A comparison study on protein fold recognition," *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, vol. 5, pp. 2492-2496, IEEE, Nov. 2002.

[5] A. Chinnasamy, W. K. Sung, and A. Mittal, "Protein structure and fold prediction using tree-augmented naive Bayesian classifier," *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 04, pp. 803-819, 2005.

[6] C. H. Ding, and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," Bioinformatics, vol. 17, no. 4, pp. 349-358, 2001.

[7] J. J. Ward, L. J. McGuffin, B. F. Buxton, and D. T. Jones, "Secondary structure prediction with support vector machines," *Bioinformatics*, vol. 19, no. 13, pp. 1650-1655, 2003.

[8] S. Hua, and Z. Sun, "A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach," *J. Mol. Biol.*, vol. 308, pp. 397–407, 2001.

[9] H. Kim, H. Park, "Protein secondary structure prediction based on an improved support vector machines approach," *Protein Engineering*, vol. 16, no. 8, pp. 553-560, 2003.

[10] K. E. Chen, L. A. Kurgan, and J. Ruan, “Prediction of protein structural class using novel evolutionary collocation-based sequence representation,” *Journal of computational chemistry*, vol. 29, no. 10, pp. 1596-1604, 2008.

[11] T. G. Dietterich, “Ensemble methods in machine learning,” *International workshop on multiple classifier systems*, Springer Berlin Heidelberg, vol. 1857, pp. 1–15, June 2000.

[12] R. King, M. Ouali, A. Strong, A. Aly, A. Elmaghraby, M. Kantardzic, and D. Page, “Is it better to combine predictions?,” *Protein Engineering*, vol. 13, pp. 15-19, 2000.

[13] P. Kountouris, M. Agathocleous, V. J. Promponas, G. Christodoulou, S. Hadjicostas, V. Vassiliades, and C. Christodoulou, “A comparative study on filtering protein secondary structure prediction,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 3, pp. 731-739, 2012.

[14] M. Alirezaee, A. Dehzangi, and E. Mansoori, “Ensemble of neural networks to solve class imbalance problem of protein secondary structure prediction,” *International Journal of Artificial Intelligence & Applications*, vol. 3, no. 6, pp. 9, 2012.

[15] A. Bhola, S. K. Yadav, and A. K. Tiwari, “Machine Learning based Approach for Protein function prediction using sequence derived properties,” *International journal of computer applications*, vol. 105, no. 12, 2014.

[16] I. Melvin, J. Weston, C. S. Leslie, and W. S. Noble, (2008). “Combining classifiers for improved classification of proteins from sequence or structure,” *Bmc Bioinformatics*, vol. 9, no. 1, pp. 389, 2008.

[17] G. Pollastri, P. Baldi, P. Fariselli, and R. Casadio, “Prediction of coordination number and relative solvent accessibility in proteins,” *Proteins: Structure, Function, and Bioinformatics*, vol. 47, no. 2, pp. 142-153, 2002.

[18] W. Li, Y. Chen, and Y. Zhao, “Multi-layer ensemble classifiers on protein secondary structure prediction,” *In International Conference on Intelligent Computing*, Springer Berlin Heidelberg, pp. 79-85, Sep. 2008.

[19] 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, 2014.

[20] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi, “Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles,” *Proteins: Structure, Function, and Bioinformatics*, vol. 47, no. 2, pp. 228-235, 2002.

[21] H. Bouziane, B. Messabih, and A. Chouarfia, “Effect of simple ensemble methods on protein secondary structure prediction,” *Soft Computing*, vol. 19, no 6, pp. 1663-1678, 2015.

[22] A. Dehzangi, S. Phon-Amnuaisuk, and O. Dehzangi, “Enhancing protein fold prediction accuracy by using ensemble of different classifiers,” *Australian Journal of Intelligent Information Processing Systems*, vol. 26, no. 4, pp. 32-40, 2010.

[23] A. Dehzangi, K. Paliwal, A. Sharma, O. Dehzangi, and A. Sattar, “A combination of feature extraction methods with an ensemble of different classifiers for protein structural class prediction problem,” *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 10, no. 30, pp. 564-575, 2013.

[24] J. A. Cuff, G. J. Barton, “Evaluation and improvement of multiple sequence methods for protein secondary structure prediction,” *Proteins: Structure, Function, and Bioinformatics*, vol. 34, no. 4, pp. 508-519, 1999.

[25] Protein: https://en.wikipedia.org/wiki/Protein

[26] Protein yapısı: http://tr.wikipedia.org/wiki/Protein_yapısı

[27] Protein structure: https://en.wikipedia.org/wiki/Protein_structure

[28] Torsion Angles and the Ramachandran Plot: http://www.proteinstructures.com/Structure/Structure/Ramachandran-plot.html

[29] Picture of a protein dihedral angles phi,psi and omega, Quora user Jeffrey Brender, https://www.quora.com/How-common-is-it-for-omega-dihedral-angles-to-have-a-value-other-than-+180-180-or-0%C2%B0

[30] Comparison of SURFACE and AREAIMOL for accessible surface area calculations, user Peter Briggs: http://www.ccp4.ac.uk/newsletters/newsletter38/03_surfarea.html

[31] 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.

[32] F. M. Richards, and C. E. Kundrot, "Identification of structural motifs from protein coordinate data: secondary structure and first-level supersecondary structure," *Proteins: Structure, Function, and Bioinformatics*, vol. 3, no. 2, pp. 71-84, 1988.

[33] D. Frishman, and P. Argos, "Knowledge-based protein secondary structure assignment," *Proteins: Structure, Function, and Bioinformatics*, vol. 23, no. 4, pp. 566-579, 1995.

[34] A. Zemla, C. Venclovas, K. Fidelis, and B. Rost, "A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment," *Proteins: Structure, Function, and Bioinformatics*, vol. 34, no. 2, pp. 220-223, 1999.

[35] M. Remmert, A. Biegert, A. Hauser, J. Söding, "HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment," *Nat. Methods.*, vol. 9, no. 2, pp. 173-175, 2011.

[36] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," *In Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 4277-4280, IEEE, March 2012.

[37] 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, pp. 49, 2008.

[38] Z. Aydin, D. Baker, and W. S. Noble, “Constructing Structural Profiles for Protein Torsion Angle Prediction,” *In BIOINFORMATICS*, pp. 26-35, 2015.

[39] 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,” *Proceedings of the International Conference on Bioinformatics & Computational Biology (BIOCOMP)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), pp.473, Jan. 2012.

[40] S. M. Reynolds, L. Käll, M. E. Riffle, J. A. Bilmes, and W. S. Noble, “Transmembrane topology and signal peptide prediction using dynamic bayesian networks,” *PLoS Comput Biol*, vol. 4, no. 11, pp. e1000213, 2008.

[41] Graphical Models Toolkit (GMTK): http://melodi.ee.washington.edu/~bilmes/gmtk/.

[42] J. A. Cuff, and G. J. Barton, *Proteins: Struct. Funct. Genet.* , vol. 34, pp. 508–519,1999.

[43] B. Rost, and C. Sander, *J. Mol. Biol.* , vol. 232, pp. 584–599, 1993.

[44] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, “Choosing multiple parameters for support vector machines,” *Machine learning*, vol. 46, no. 1, pp. 131-159, 2002.

[45] S. Busuttil, “Support vector machines,” *In Proceedings of CSAW’03*, pp. 34, Nov. 2003.

[46] Picture of non-linear SVM, Researchgate user atarina Moreira, https://www.researchgate.net/figure/260283043_fig13_Figure-A15-The-non-linear-SVM-classifier-with-the-kernel-trick

[47] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[48] H. M. Karakoyun, and M. Hacıbeoğlu, “Biyomedikal Veri Kümeleri İle Makine Öğrenmesi Sınıflandırma Algoritmalarının İstatistiksel Olarak Karşılaştırılması,” *DEÜ Mühendislik Fakültesi Mühendislik Bilimleri Dergisi*, vol. 16, pp. 30-41, 2014.

[49] A. Statnikov, L. Wang, and C. F. Aliferis, “A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification,” *BMC bioinformatics*, vol. 9, no. 1, pp. 319, 2008.

[50] B. J. Lee, M. S. Shin, Y. J. Oh, H. S. Oh, and K. H. Ryu, “Identification of protein functions using a machine-learning approach based on sequence-derived properties,” *Proteome science*, vol. 7, no. 1, pp. 27, 2009.

[51] Y. Qi, M. Oja, J. Weston, and W. S. Noble, “A unified multitask architecture for predicting local protein properties,” *PloS one*, vol. 7, no. 3, pp. e32235, 2012.

[52] P. Di Lena, K. Nagata, and P. Baldi, “Deep architectures for protein contact map prediction,” *Bioinformatics*, vol. 28, no. 19, pp. 2449-2457, 2012.

[53] R. Collobert, and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” *In Proceedings of the 25th international conference on Machine learning*, pp. 160-167, ACM, July 2008.

[54] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *Journal of Machine Learning Research*, pp. 2493-2537, 12 Aug. 2011.

[55] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798-1828, 2013.

[56] J. Peng, L. Bo, and J. Xu, “Conditional neural fields,” *In Advances in neural information processing systems*, pp. 1419-1427, 2009.

[57] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.

[58] S. Wang, J. Peng, J. Ma, and J. Xu, “Protein secondary structure prediction using deep convolutional neural fields,” *Scientific reports* 6, 2016.

[59] AUC-MAXIMIZED DEEP CONVOLUTIONAL NEURAL FIELDS FOR SEQUENCE LABELING - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/286263785_fig1_Figure-1-Illustration-

of-a-DeepCNF-Here-i-is-the-position-index-and-X-i-the-associated
[accessed 17 May, 2017]