In-Silico Drug Discovery using Protein-Small Molecule Interaction

Project Report Submitted in Partial Fulfillment of the Requirements for the

Degree

Bachelor of Technology

in

Electronics and Communication Engineering

Submitted by

Gaurav Mishra : 1510110480

Under the Supervision of

Prof. Gajendra P.S. Raghava

Head of Department,

Center for Computational Biology,

IIIT Delhi, Okhla Phase 3, New Delhi

and

Dr. Madhur Deo Upadhayay

Assistant Professor,

Deptt. of Electrical Engineering,

Shiv Nadar University,

Gautam Buddha Nagar, UP

Department of Electrical Engineering

May 2019

To my family

Candidate Declaration

May, 2019

I hereby declare that the thesis entitled "In-Silico Drug Discovery using Protein-Small Molecule Interaction" submitted for the B.Tech Degree program has been written in my own words. I have adequately cited and referenced the original sources.

Gaurav Mishra
Roll Number: 1510110480
May 11, 2019

Certificate

May, 2019

I hereby state that the thesis entitled "In-Silico Drug Discovery using Protein-Small Molecule Interaction" submitted for the B.Tech Degree program by Gaurav Mishra was completed under my on-campus supervision. This thesis bears my approval.

Dr. Madhur Deo Upadhayay
Assistant Professor
Department of Electrical Engineering,
Shiv Nadar University,
Gautam Buddha Nagar, UP, India.
13th May 2019

ACKNOWLEDGEMENT

Firstly, I would like to express my sincere gratitude to my advisor Prof. Gajendra P.S. Raghava for his continuous support towards my project and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my final year thesis. I would also like to thank him for providing me an opportunity to join his team as research intern, and for giving me access to the laboratory and research facilities. Without his precious support it would not have been possible to conduct this project.

Besides my advisor, I would like to thank the rest of Prof. Raghava's lab, specifically Piyush Agarwal, for their insightful comments and encouragement, but also for their valuable inputs which helped me widen my research from various perspectives. I would also like to thank them for sharing their domain expertise with me, as and when required. My sincere thanks goes to them.

I would also like to thank my local supervisor, Dr. Madhur Deo Upadhayay, who helped me and kept me informed of all departmental proceedings and requirements during my internship. I would also like to thank him on his valuable suggestions at various stages.

I thank my friends Akash, Akshay, Ishmeet, Karan, Karthik, Satvik and Simran who kept my morale up throughout the span of the project.

Last but not the least, I would like to thank my family: my parents and my sisters for supporting me at every stage of my life.

Gaurav Mishra

May 11, 2019

ABSTRACT

Proteins are the fundamental players of all living cells and play a vital role in various cellular functions. Protein function is primarily determined by its structure. Interaction of protein with other molecules for example, ligand or other small molecules imparts its specific function. Protein-small molecule interaction is one such interaction which has been studied in detail in past in the field of drug designing. These small molecules carry out various processes such as signal transmission, post-translational modification, etc. therefore it is important to understand the protein-small molecule interaction in order to design novel drugs. The project aims at creating an in-silico model drug discovery model by predicting if a particular amino acid in a given protein would bind to a ligand. We have chosen the ligand as Uridine-5'-diphosphate. The ligand imparts many important functionalities to the proteins it interacts with. Various machine learning algorithms have been used to generate predictive models which successfully do the required task. In this study, we will make an attempt to annotate protein at residue level. We will select biologically important ligand to find out interaction with given protein sequences by implementing machine learning techniques including. The application of this work will be in drug designing where we will target the proteins which are involved in various diseases like cancer, tuberculosis, etc.

Keywords: Uridine-5'-Diphosphate, UDP, UDP Binding Site Prediction, Machine Learning Techniques, Therapeutic Drug Target, PSSM Profile, Drug Discovery

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Form
Acc.	Accuracy
AUROC	Area under ROC Curve
DNA	Deoxyribonucleic Acid
DSSP	Defined Secondary Structures of Proteins
FN	False Negative
FP	False Positive
FPR	False Positive Rate
kNN	K-Nearest Neighbours
LPC	Ligand Protein Contact
MCC	Matthew's Correlation Coefficient
ML	Machine Learning
MLP	Multi-Layer Perceptron
PCA	Prinicipal Component Analysis
PDB	Protein Data Bank
PSI-BLAST	Position-Specific Iterative Basic Local Alignment Search Tool
PSSM	Position Specific Scoring Matrix
RF	Random Forest
RFE	Recursive Feature Elimination
RNA	Ribonucleic Acid
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate
UDP	Uridine-5'-Diphosphate

1 Introduction

In the post genomics era, there is a huge gap in number of protein sequences and solved protein 3D structures. Therefore, it is important to annotate the protein function at sequence level. Previously, methods have been developed at protein level as a whole, for example, whether the protein is DNA interacting or not. Annotation at residue level provides more comprehensive information for example, it tells us that whether each residue in the protein will interact with a small molecule or not.

Nucleotides are organic molecules which act as monomer units to two most important bio-molecules within all life-forms of earth, DNA and RNA[1]. Nucleotides play a central role in providing metabolism at cellular level. They carry energy to cells to facilitate cell division, synthesis of amino acids etc. The nucleotides also play a very vital role in cell-signaling and in multiple signaling pathways. The nucleotides bind to proteins to achieve these very fundamental functionalities which become an important precursor to all the biological processes.

All these vital tasks take place because these nucleotides bind to protein enabling it to function as it should. However, there are only certain parts of a protein where these nucleotides can bind and mediate the protein's functionality. Predictive analyses have been used to predict these binding sites by annotating protein chains at structural and functional level. In the past, several methods have been used to predict these binding sites in a protein. Methods first developed were generalised in nature meaning that the predictive analyses done were not ligand-specific but were developed on the premise that once binding sites or pockets are known any ligand could bind to that site. Further studies and discoveries in structural biology led to the findings which suggested that ligands can be of different sizes, shape and nature and having a generalised method which treats all ligands the same way would not perform better. This led to development of ligand-specific

predictive analyses [2, 3, 4, 5, 6] wherein a ligand is chosen and as per the existing data on its interaction, binding sites in proteins are predicted. It was observed that these ligand specific predictive analyses gave better results than the generalised ones. [7, 8, 9]

2 Uridine-5'-Diphosphate

The ligand which has been chosen for the analyses to follow is Uridine-5-Diphosphate(UDP). It is a uracil nucleotide containing a pyrosulphate group esterified to C5 of the sugar moiety. It belongs to the class of organic compounds known as pyrimidine ribonucleoside diphosphates. It is the most common nucleotide carrier for sugars and is the most abundant of all nucleotide-sugars. UDP-glucose was the first nucleotide sugar to be discovered [10].

UDP plays an important role in glycogenesis, the process by which glucose molecules are added to chains of chains of glycogen for storage. It plays a significant role in synthesis of glycogen by forming intermediate compounds. Enzyme UDP-glucose pyrophosphorylase converts glucose-1-phosphate to UDP-glucose. This UDP-glucose unit are then acted upon by enzyme glycogen synthase which combines these UDP-glucose units to form a glycogen chain. Role of UDP is vital as it enables glucose to be stored as glycogen in the liver and muscles.[11] UDP mediates various biological effects via P1 and P2 purinergic receptors which makes it an important extracellular pyrimidine signaling molecule. [12, 13]

Apart from these, UDP, as a metabolite, is involved in multiple biochemical pathways such as amino sugar metabolism, Lactose synthesis, Nicotine Action pathway etc. UDP also hydrolyses ATP and other nucleotides to regulate purinergic neurotransmision(Uniprot ID P49961). It also provides important precursors for DNA synthesis.(Uniprot ID P31350)

The UGT1A1 carries out the metabolism and subsequent excretion of various molecules which comprises toxic xenobiotics, carcinogens, estrogens, bilirubin and therapeutic drugs. This enzyme covalently links glycosyl groups to lipophilic substrates, making them more water soluble and therefore more readily excreted.[14]

After researching on the roles of UDP in multiple biological processes, we decided to choose it as our ligand of choice and developed predictive methods.

3 Data

3.1 Tools and Databases Used

3.1.1 Protein Data Bank (PDB)

Protein Data Bank[15] is a database for three-dimensional structural data of large molecules such as proteins. These data are submitted by biologists and biochemists from all around the world. The PDB is a key database source for studies in structural biology and serves as a starting point for multiple research problem statements concerning structures.

3.1.2 Ligand-Protein Contact (LPC)

Ligand-Protein Contact[16] is a tool which performs analysis of inter-atomic contacts in ligand-protein complexes and in protein entries. This helps us in analysing which residues of a given chain are in contact with UDP. LPC also gives what is the vicinity of ligand with the particular residue.

3.1.3 Defined Secondary Structure of Proteins (DSSP)

DSSP[17, 18] is used to assign secondary structure to the all the residues (amino acids) present in a protein. It is a database of secondary structure assignments for all protein entries in the PDB.

3.1.4 CD-HIT

CD-HIT[19] is a tool which is used to cluster sequences. It is a recommended tool when the sequences obtained are redundant. CD-HIT first sorts the sequences in decreasing order of chain length. It takes the longest chain as the representative sequence of the first cluster. If the next chain is similar to this representative sequence above a given threshold (*known as similarity threshold*), it is added to

that cluster otherwise a new cluster is formed with this particular sequence as the representative sequence of next cluster.

CD-HIT is a fast and greedy approach to cluster sequences. It avoids multiple pairwise sequence alignments by implementing short-word based heuristics. The short word based heuristics suggests that for two sequences to be similar up to an extent they must share a certain number of identical k-mers (k-mers is k residues taken at a time). So, it is possible to simply count number of k-mers and deduce that two sequences are not similar (based on the similarity threshold). This saves a lot of computational power and as a result, clustering is done very fast. The short word size is normally taken between 2-5 (depending on the similarity threshold). So maximum possible values of a k-mer(5-mer) is 21^5 (4 million possibilities) which is well scaled for processors these days.

This makes CD-HIT a fast clustering algorithm and is widely employed in analyses like this.

3.2 Protein Data Collection

The PDB was searched to find instances of UDP in the database. 406 entries were found which had UDP occurring as a free ligand in them. After this, two in-built filters present in PDB were applied:

- Protein length should be between 50 and 50000.
- X-ray resolution should be between 0-3 Å.

Following this filter, out of 406 entries 381 were left and 25 entries were filtered out. From these 381 entries, 864 individual protein chains were obtained. These chains contained the following attributes which made them inappropriate for further processing and required further filtering:

- The chains contained invalid non-natural residues viz. B, J, O, U, Z, X, or

Similarity Threshold	Number of Clusters
40%	68
45%	70
50%	72
55%	75
60%	76
95%	89

Table 3.1: CD-HIT's output for different thresholds

- The chains had length less than 50, or
- The chains had residues whose contact with UDP was farther than 4 Å.

The 864 chains were further filtered on the basis of aforementioned points, this left 445 sequences which were compatible for next set of processing methods. These 445 sequences contain redundant sequences as well. To assess this redundancy, we use CD-HIT (As mentioned in 3.1.4).

Table 3.1 shows the results obtained after running CD-HIT at different thresholds on 445 sequences:

The clusters formed at 95% threshold were chosen so that we have diverse clusters. The standard procedure after this operation is to choose only the representative sequence from each cluster ad then form data set using those representative sequences. But due to the paucity of sequences available with us, we deviate from the normal course.

3.2.1 Fold formation from clusters

The clusters received from CD-HIT were divided into folds. Rather than using only the representative sequences from each clusters, we take entire cluster and put it into folds. The sequences in each cluster share at least 40% with each of the other sequences. Putting entire cluster into folds retains the variability of data as the similarity between folds would be less. Each of the folds would express

sufficient variance so as to train and validate models on those. The following methodology has been adopted to achieve the same:

- Individual files were formed for each cluster which have the sequence ID of all the chains in them.
- These clusters were sorted in increasing order of their size (size is number of chains in each cluster).
- The clusters were divided into folds in the following way:
  - Cluster 1 (after sorting) goes into 1st fold, Cluster 2 goes into 2nd fold, Cluster 6 goes into the validation set, Cluster 7 goes into 1st fold and so on.

These description of the folds and the test set are given in Table 3.2

Detail	**Number of Sequences**
fold1	71
fold2	71
fold3	71
fold4	71
fold5	71
test	90
total	445

Table 3.2: Description of Data Set formed

3.3 Positive and Negative Residues

An example protein sequence in fasta format would look like:

```
>1f7rA
MIIEGDGILDKRSEDAGYDLLAAKEIHLLPGEVKVIPTGVKLMLPKGYWGLIIGKSSIGSKGL
```

DVLGGVIDEGYRGEIGVIMINVSRKSITLMERQKIAQLIILPCKHEVLEQGKVVMDSERGDNG
YGSTGVF

This sequence has UDP as a free ligand as mentioned in its PDB entry. Only a few residues in this seuquence would be bound to UDP. The residues which have UDP bound to them are called as *positive residues* and the ones which are not bound to UDP are called *negative residues*. This data can be tallied with tables obtained from LPC and DSSP(sections 3.1.2 and 3.1.3). LPC would provide with residue numbers which are in contact with UDP, same can be tallied with DSSP generated. For example, the sequence given above will have DSSP and LPC as shown in figures 3.1 and 3.2.

As evident in figure 3.1 residue numbers 68, 69, 70, 71, 74, 78, 79, 81 are in direct contact with UDP and are within the vicinity of 4 Å. These residues are *positive*. These residue numbers can be referenced in the DSSP data to know which amino acid interact with UDP. Such residues have been marked in lower case and the others, which are non-interacting and hence, negative, have been marked in capital letters.

A sequence once processed to find positive and negative binding site would look like the following:

>1f7rA
MIIEGDGILDKRSEDAGYDLLAAKEIHLLPGEVKVIPTGVKLMLPKGYWGLIIGKSSIGSKGL
DVLGgvidEGyRGEigViMINVSRKSITLMERQKIAQLIILPCKHEVLEQGKVVMDSERGDNG
YGSTGVF

1f7rA	UDP	138A	68	A	GLY	3.2	31.7	+	-	-	-
1f7rA	UDP	138A	69	A	VAL	3.4	37.4	+	-	+	+
1f7rA	UDP	138A	70	A	ILE	3.5	26.0	-	-	+	+
1f7rA	UDP	138A	71	A	ASP	3.1	31.7	+	-	-	+
1f7rA	UDP	138A	74	A	TYR	3.4	57.8	+	-	+	+
1f7rA	UDP	138A	78	A	ILE	3.8	5.2	-	-	-	-
1f7rA	UDP	138A	79	A	GLY	2.7	45.7	+	-	-	+
1f7rA	UDP	138A	81	A	ILE	3.7	29.4	+	-	+	+

Figure 3.1: Ligand Protein Contact of Sequence ID 1f7rA

60	A	S	T	107	-71.1	8.5
61	A	K	T	125	-105.0	0.8
62	A	G	T	27	94.6	15.9
63	A	L	E	7	-112.3	139.0
64	A	D	E	87	-127.4	158.3
65	A	V	E	20	-105.2	129.4
66	A	L	E	69	-107.2	160.0
67	A	G	C	69	62.6	-123.4
68	A	G	C	10	107.2	10.8
69	A	V	E	85	-106.1	112.3
70	A	I	E	15	-98.6	118.8
71	A	D	C	107	-79.2	144.9
72	A	E	T	59	-48.1	-39.4
73	A	G	T	53	-82.2	-12.3
74	A	Y	C	100	-52.4	138.9
75	A	R	C	121	-133.5	2.2
76	A	G	S	25	-89.0	-161.5
77	A	E	C	82	-87.7	139.6
78	A	I	C	1	-79.4	126.0

Figure 3.2: DSSP data of Sequence ID 1f7rA

It can be seen that the residues (G, V, I, D, Y,I, G) which interact with UDP (refer to figures 3.1 and 3.2) have been changed to lower case.

3.4 Pattern Generation

After the sequences were marked differently for positive and negative residues, they need to be made compatible for any typical machine learning method. The sequences may come in variable length so it is important that we make the features which are of fixed length. To further generate features based on these sequences, we generate fixed-length patterns from each sequence. The pattern sizes vary from a window size of 5 till 23, all in odd numbers only. Consider an example sequence:

```
>exampleSequence
ACDDefigIVEF
```

Referring to the sequence above, if patterns for a window size of 5 were to be generated it would be in the following format:

```
XXACD
XACDD
ACDDe
CDDef
DDefi
Defig
efigI
figIV
igIVE
gIVEF
IVEFX
VEFXX
```

The patterns generated above for the given example sequence has 12 patters. In each of the patters, letter in the centre is the residue which pertains to that pattern. The pattern of 5 conveys information about the sequence considering each residue and it's two neighbours on either side. The first residue and the last residue are padded with a placeholder 'X' to convey that the sequence ends on either side. Using this method sequences of fixed lengths are obtained. These patterns bring out important information about neighbours for each residues. Smaller segments are considered which bring in more information about the sequences.

Further to this, negative and positive data sets are formed by combining all positive patterns and negative pattern. Positive patterns are such patterns in which the residue at the centre are positive residues. For example, the positive examples would be:

```
Positive Patterns:
                              DDefi
                              Defig
                              efigI
                              figIV
```

Similarly, the negative patterns would be:

```
Negative Patterns
                              XXACD
                              XACDD
                              ACDDe
                              CDDef
                              igIVE
                              gIVEF
                              IVEFX
                              VEFXX
```

X is appended to left of first residue and right side of last residue so as to attribute the beginning and end of the sequence. The number of 'X' to be appended on either side is given by the formula

$$n = \frac{w-1}{2} \tag{3.1}$$

where n is number of Xs to be appended and w is pattern length.

Positive and negative patterns from all the chains within a fold are joined together to form the pattern dataset. All negative patterns are put into negative dataset and positive into positive dataset.

Sequence based data is ready now and next steps are to form feature sets from these, as covered in next section.

3.5 Imbalanced Dataset

In the original dataset, total number of positive and negative examples are enumerated in Table 3.3

Dataset	Positive	Negative
Fold 1	881	21713
Fold 2	885	23939
Fold 3	919	21992
Fold 4	890	23180
Fold 5	914	24984
Test Set	1139	31014
Total	5628	146822

Table 3.3: Count of realistic data points

As evident from Table 3.3, the dataset is highly imbalanced with only 3.8 % positive samples. This creates a bias in the model we will try to fit as the model would learn majorly from the negative samples and would not classify the positive samples very efficiently.

3.5.1 Undersampling

To curb the problem of imbalanced dataset, we tried **undersampling**. Undersampling is an approach wherein we cut short the majority class dataset so as to match the number of positive data points. The count of new dataset has been mentioned in Table 3.4.

Dataset	Positive	Negative
Fold 1	881	881
Fold 2	885	885
Fold 3	919	919
Fold 4	890	890
Fold 5	914	914
Test Set	1139	1139
Total	5628	5628

Table 3.4: Count of balanced data points

All further analyses have been done on the balanced dataset.

4 Methodology

Two approaches have been used to create data set on which predictive analyses would be done.

4.1 Binary Profiling

In order to generate machine learning ready data, we generate binary profiles for the positive and negative patterns. Each amino acid is encoded as a vector of size 21 containing 0s and 1s. The encoding scheme is mentioned in Table 4.1:

Pertaining to each amino acid in a given pattern, binary profiles are made and concatenated one after another. The length of feature vector created would be given by the formula:

$$l = w * 21, \tag{4.1}$$

where l is length of vector and w is length of pattern generated
For example, patterns generated of length 5 would make a feature set of length 5*21 = 105.

4.2 Position Specific Scoring Matrix

A PSSM, or Position-Specific Scoring Matrix, is a type of scoring matrix that provides evolutionary information in the form of substitution scores given separately for each amino acid at a particular position in multiple sequence alignment of proteins. The PSSM profile is generated by hitting the query sequence against non-redundant protein sequence database. The similarity search takes place in between the query sequence and the sequences present in the database. The score is provided in the form of positive and negative integers where positive score suggests that the given amino acid substitution occurs very frequently than expected by chance in the given alignment whereas a negative score indicates that the sub-

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

Table 4.1: Binary Encoding of Sequences

stitution is less frequent than expected. Large positive scores provide useful insight such as information of critical functional residues which could be an active site or these residues are involved in various interactions. PSSM are often derived from a set of aligned sequences that are thought to be functionally related and have become an important part of many software tools for computational motif discovery.

The PSSM profiles were generated using Position-Specific Iterative Basic Local Alignment Search Tool (PSI-BLAST) and searching against the swiss prot database. Three iterations were performed with e-value cut-off 0.01 against each

sequence. The profiles thus obtained are normalised to get a value between 0 and 1, followed by calculation of position specific score for each residue. The final matrix consists of 21 x N elements, where N is the length of target protein sequence (i.e. the pattern size from 5-23). Each element represents the probability of occurrence of each amino acid residue at a particular position in the obtained alignment. This is to say that the evolutionary information for each amino acid corresponding to the given sequence is encapsulated in the vector of dimension 21.

5 Algorithms Used

Following ML algorithms have been applied to predict the binding sites:

5.1 Support Vector Machine(SVM)

SVM was introduced in 1992 by Boser, Guyon and Vapnik. SVM uses the concept of hyperplanes which divide the feature space into two subspaces (for binary classification). Mathematically, the optimisation of SVM is a problem of non-linear optimisation. The objective is to reduce the loss function while maximising the margin. The dual formula formed is:
Minimise

$$L(\alpha) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) + \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i^* - \alpha_i) \quad (5.1)$$

such that

$$\sum(\alpha_n - \alpha_n^*) = 0,\ \forall n : 0 \leq \alpha_n, \alpha_n^* \leq C$$

where ε is the 'soft-margin' margin and C is the penalty imposed on observations that lie beyond the ε-margin. It prevents overfitting. This value determines the trade-off between the flatness of f(x) and the amount upto which deviations beyond ε are tolerated.

5.1.1 Non-linearity in Data

In real-life data sets, it is highly improbable that the data points are linearly separable. The principle of SVM posits the linear separability of data points. In such cases, kernel method is used to transform data points into higher dimensional space such that they become linearly separable.

Three types of kernels (represented as $\phi(x_j, x_k)$)are widely used:

- Linear Kernel $\phi(x_j, x_k) = x_j' x_k$

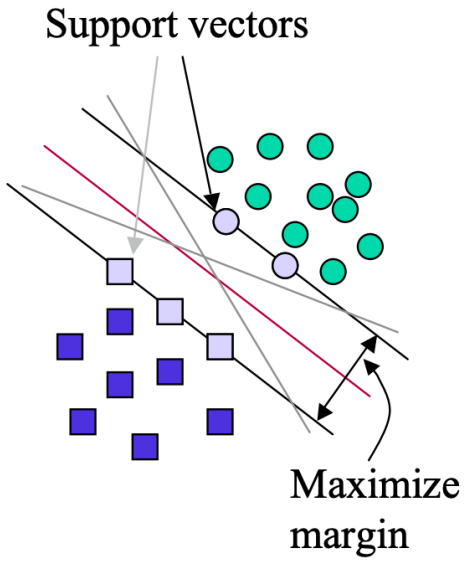

Figure 5.1: Support Vector Classification

- Gaussian Kernel $\phi(x_j, x_k) = e^{-\gamma||x_j - x_k||}$, also known as radial basis function (rbf)
- Polynomial Kernel $\phi(x_j, x_k) = (1 + x_j' x_k)^p$ where p is the degree of polynomial

5.2 k-Nearest Neighbours(kNN)

Parametric methods used so far have a disadvantage, they assume a model for f(x) and then tune the parameters on that assumed framework. If the assumed model is not the same as the real model, we end up getting a poor performance. In contrast, non-parametric approaches such as k-NN do not explicitly assume a parametric form for the target function f(X) and in turn provide a more flexible approach.

General approach of a kNN classifier is as follows:

- For a given value of k and a prediction data point x_0

- kNN identifies the k nearest training data points N_0 closest* to the prediction data point x_0
- The estimation of class is done by identifying class of maximum points N_0 near the input data point x_0

*To come to a decision about how close a point is to other data point (in higher dimension), a number of distance metrics have been taken into account. Some of them are:

- Minkowski Distance $\left(\sum |x_1 - x_2|^p\right)^{\frac{1}{p}}$ where x_1 and x_2 are two data points
  - If p=1, it is called Manhattan Distance.
  - If p=2, it is called Euclidean Distance.
- There are other types of distance metrics as well like Chebyshev, Mahalanobis, etc.

5.3 Multi-Layer Perceptron

A Multi Layer Perceptron (MLP) is a type of feed-forward artificial neural networks. Except for the input nodes, each node is a neuron that uses a nonlinear activation function and takes an input and passes an output. MLP utilizes a supervised learning technique called back-propagation for training. The Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

As shown in figure 5.2, MLP is a finite acyclic graph. The nodes are called neuron and they have an activation function which transforms the input from previous layer to output a value, which in turns becomes input for the nodes in next layer. The first layer is called the **input layer** and the last one is called an **Output Layer**. All the layers in between are called **hidden layers**. An MLP has a minimum of one hidden layer.

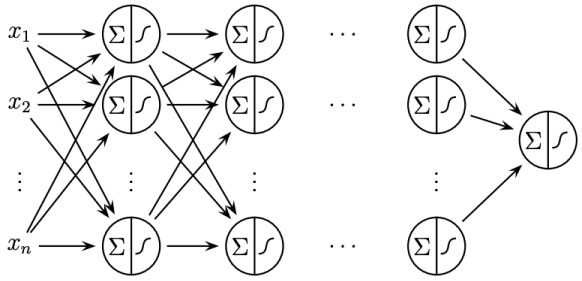

Figure 5.2: Architecture of an MLP

5.3.1 Activation Functions

In neural networks, the output to the next node is provided by an activation function of the given node. This output is further used as an input for the next node present in the next layer. This process continues till the last layer. Activation functions are used in every layer except the input layer.

Following types of activation functions(Represented as $\sigma(x)$) are more commonly used in Neural Networks:

- Sigmoid or Logistic: $\sigma(x) = \frac{1}{1+e^{-x}}$
- Tan Sigmoid: $\sigma(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
- Rectified Linear Unit (ReLU) $\sigma(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases}$

Several other types of activation functions depending on the dataset are used. We have employed these activation functions in our analyses.

5.4 Random Forest

Random Forest is an important machine learning technique which comes under the type ensemble learning. This technique has been used in number of problem

involving both classification and regression. RF constructs multiple decision trees at the time of training and takes vote of all the learners and choses the majority vote (or mode of all the observations)

In Random Forest we do not use all the predictors while deciding for a split but instead we consider only a random subset of all the features.

5.5 Extratrees Classfier

Adding one extra randomization to RFs yields extremely randomized trees, or ExtraTrees. Extremely Randomised Trees are a variant of random forest that select splits, both attribute and cut-point, totally or partially at random while in random forests, only attributes are selected at random. This adds another layer of randomness and helps the tree to generalise better and prevent overfit. It is still an ensemble method which takes a vote of each weak learners and then outputs the mode of these predictions.

While similar to ordinary random forests in that they are an ensemble of individual trees, there are two main differences: first, each tree is trained using the whole learning sample (rather than a bootstrap sample), and second, the top-down splitting in the tree learner is randomized. Instead of computing the locally optimal cut-point for each feature under consideration (based on, e.g., information gain or the Gini impurity), a random cut-point is selected.

5.6 Ridge Classifier

Ridge Classifier is a classifer which implements L2 regularization. L2 regularization adds an L2 penalty to the weights of the features. This helps in preventing overfitting and is an important regularisation method. A tuning parameter (λ) controls the strength of the penalty term. When λ= 0, ridge regression equals least squares regression. If $\lambda = \inf$, all coefficients are shrunk to zero. The ideal penalty is therefore somewhere in between 0 and inf.

Ridge regression places a particular form of constraint on the parameters(βs): $\hat{\beta}_{ridge}$ is chosen to minimize the penalized sum of squares:

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{5.2}$$

which is equivalent to minimization of $\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2$ for some $c > 0$, $\sum_{j=1}^{p} \beta_j^2 < c$, i.e. constraining the sum of the squared coefficients.

6 Machine Learning on Data Set

6.1 Evaluation Metrics

The following evaluation metrics have been chosen to assess the performance of the models:

$$Sensitivity = \frac{TP}{TP+FN} * 100 \tag{6.1}$$

$$Specificity = \frac{TN}{TN+FP} * 100 \tag{6.2}$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} * 100 \tag{6.3}$$

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{6.4}$$

Where TP, FP, TN, FN are True Positive, False Positive, True Negative and False Negative respectively.

Sensitivity (also called the true positive rate) measures the proportion of actual positives that are correctly identified as such.

Specificity (also called the true negative rate) measures the proportion of actual negatives that are correctly identified as such

Matthew's Correlation Coefficient (MCC) is in essence a correlation coefficient between the observed and predicted binary classifications; it returns a value between 1 and +1. A coefficient of +1 represents a perfect prediction which is in line with the actual output, 0 no better than random prediction and 1 indicates total de-correlation between prediction and observation. It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes. It is a better metric than accuracy which only considers if correct classification takes place whereas

MCC does not give an overly optimistic outcome and considers the distribution of classes(positive or negative) as well.

6.2 K-fold Cross Validation

In k-fold cross-validation, the original sample is randomly partitioned into k equal sized data sets. Each of these k datasets are referred to as folds. Out of these k folds, one fold is retained as the validation data for testing the model, and the remaining k - 1 folds are used for training the model. The cross-validation process is then repeated k times, where each of the folds is used once for validation. Average result is reported after performing k-fold cross validation.

For our analyses, we have chosen k as 5 and have performed 5-fold cross validation.

6.3 Grid Search

In order to tune the hyper-parameters, Grid Search method has been used. It does an exhaustive search over given parameters for an estimator. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid. The estimators and parameters which were tuned have been mentioned in table 6.1:

6.4 Pattern Generation

AS mentioned in section 3.4, patterns of sizes 5, 7, 9, 11, 13, 15, 17, 19, 21 were developed for both Binary and PSSM dataset. Grid search was carried out for each of the pattern sizes and for both binary and PSSM datasets.

Estimator	Parameter	Specified Values
SVM	Kernel	rbf, linear,
	C	1,2,3,4,5,6,7,8,9,10
	Gamma(for rbf)	1,0.5,0.01,0.001, 0.0001
Extra Trees	Number of estimators	10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 600, 700, 800, 1000
Random Forest	Number of estimators	10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 600, 700, 800, 1000
MLP Classifier	Activation Function	identity, logistic, tanh, relu
	Solver	lbfgs, sgd, adam
	Hidden layer size	1-21
	Max iteration	1000
k- Nearest Neighbours	Number of neighbours	5,6,7,8,9,10
	Weights	uniform, distance
	Algorithm	auto, ball tree, kd tree, brute
Ridge Classifier	Alpha	1,0.1,0.01,0.001,0.0001,0

Table 6.1: Parameters for Grid Search

6.5 Feature Selection

The feature size which we have is large starting from 105 (for patterns of size 5) to 483 (for patterns of size 23). We tried few approaches to reduce the feature vector size, the algorithms for those are mentioned under:

6.5.1 Principal Component Analysis (PCA)

PCA aims at orthogonalising the covariance matrix such that there is least or zero covariance between two features. The idea of PCA is to find a linear combination of these two predictors which contains most of the information conveyed by these two original predictors. In other words, PCA distributes the variance explained by each predictors into the new predictor (which is a linear combination of the those features).

Problem Definition

For a given input **X**, we need to find a feature matrix **W**

Such that their weights de-correlate i.e.

$$(WX)(WX)^T = NI$$

Upon solving:

$$WXX^TW^T = NI$$

$$WCov(X)W^T = NI \tag{6.5}$$

We need to diagonalise the covariance matrix so as to decrease the redundancy. Covariance matrices are positive definite which means they are factorisable by

$$U^TAU = \Lambda \tag{6.6}$$

Where U is eigenvectors of A in its columns, and $\Lambda = diag(\lambda_i)$ where λ_i are the eigenvalues of A.

Number of features to retain depends on the variance retained by the new transformed features. This threshold is subject to change as per dataset.

$$V_m = \frac{\lambda_1 + \lambda_2 + ... + \lambda_m}{\sum_{alleigenvalues} \lambda} \tag{6.7}$$

where V is the fraction of variance retained by first m principal components (sorted in descending order).

Using the value of m obtained, the original features are projected onto a lower subspace with m basis.

6.5.2 Recursive Feature Elimination (RFE)

Recursive Feature Elimination (RFE) as its title suggests recursively removes features, builds a model using the remaining attributes and calculates model accuracy. RFE is able to work out the combination of attributes that contribute to the prediction on the target variable (or class)

First, the algorithm fits the model to all predictors. Each predictor is ranked using it's importance to the model. Let S be a sequence of ordered numbers which are candidate values for the number of predictors to retain (S1 $>$ S2, ...). At each iteration of feature selection, the S_i top ranked predictors are retained, the model is refit and performance is assessed. The value of S_i with the best performance is determined and the top S_i predictors are used to fit the final model.

7 Results

7.1 Compositional Analysis

Analysis of interacting and non-interacting sites on basis of amino acid has been done in figure 7.1.

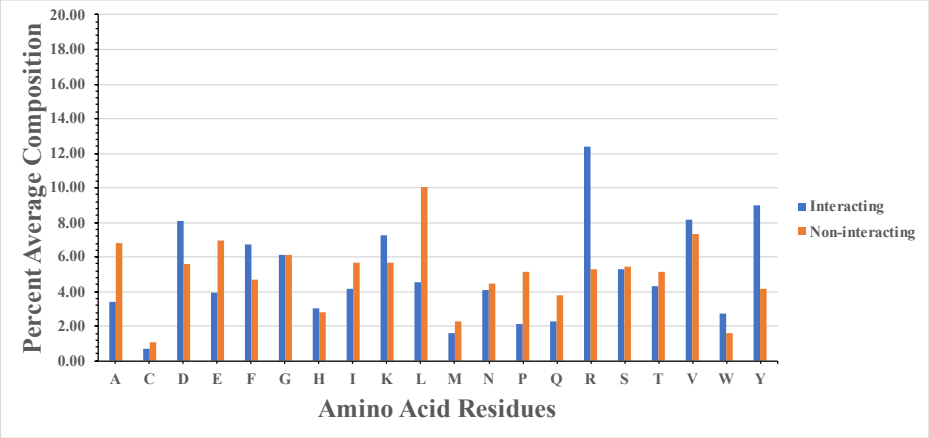

Figure 7.1: Compositional Analysis (Amino Acid)

It can be seen that some amino acids favour binding to UDP more than others. Composition of D, F, G, H, K, R, V, Y is higher in residues which interact with UDP whereas composition of A, C, E, I, L, M, N, P, Q, S, T is higher in non-interacting residues.

7.2 Propensity Analysis

Analysis of interacting and non-interacting sites on basis of propensity of amino acids has also been done in figure 7.2.

It can be observed that residues like D, F, R, W, Y have greater propensity score for interacting residues whereas residues like A, C, E, G, H, I, K, L, M, N, P, Q, S, T, Y have greater propensity score for non-interacting residues. It can also

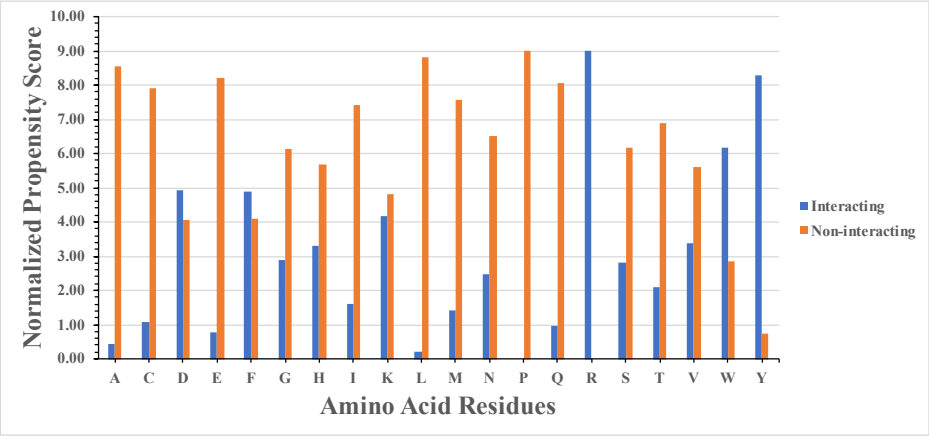

Figure 7.2: Propensity Analysis

be deduced that amino acids like A, L and P have very little normalised propensity score if they interact with UDP.

7.3 Physico-chemical Property Analysis

Further analysis, on the basis of physico-chemical properties of interacting and non-interacting residues was done as shown in figure 7.3. The residues which interact with UDP are more charged, basic, polar and aromatic whereas the ones which do not interact with UDP are more acidic, smaller in size, non-polar and aliphatic in nature.

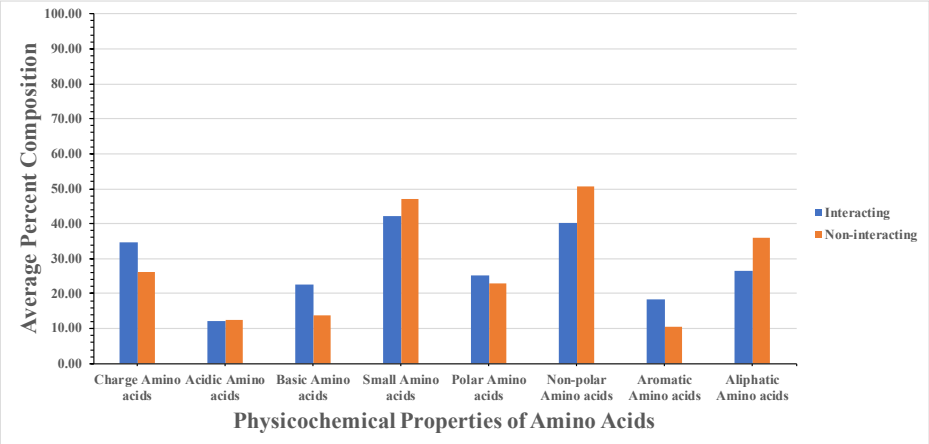

Figure 7.3: Physico-chemical Property Analysis

7.4 Results from ML algorithms

After applying Grid Search, we got a set of parameters against each algorithm for each pattern. The result has been summarised in tables below. The two cases which have been taken into account are the one with balanced sensitivity and specificity and the other with highest MCC. The former case means that the model is equally good at classifying both the positive and negative data points. The latter means that the predicted labels is highly correlated with the the actual labels and not necessarily have balanced specificity and sensitivity.

The training scores have been obtained by applying 5-fold cross validation. (Refer to tables 7.1, 7.2, 7.3, 7.4) The results in PSSM profile are greater than that of binary. Pattern 13 in PSSM profile shows similar AUROC but higher MCC as compared to other patterns of PSSM Profile. Having shorter patterns is better because this way the features generated would have more details about each segment of the sequence so we choose pattern 13 as the best case with extra trees giving the best results.

Pattern Size	Algorithm	Sensitivity	Specificity	MCC	AUROC	Acc.
5	Random Forest	62.58	58.55	0.21	0.67	60.57
7	Extra Tree	65.72	59.51	0.25	0.67	62.62
9	Random Forest	65.21	61.57	0.27	0.71	63.39
11	Random Forest	68.53	60.92	0.3	0.73	64.73
13	Random Forest	66.66	66.47	0.33	0.74	66.57
15	Extra Tree	68.46	62.75	0.31	0.74	65.60
17	Random Forest	68.30	64.97	0.33	0.74	66.64
19	Random Forest	70.69	64.69	0.35	0.74	67.69
21	Random Forest	69.14	63.87	0.33	0.74	66.51
23	Random Forest	70.2	62.98	0.33	0.74	66.59

Table 7.1: Training Statistics of Binary Profile

Pattern Size	Algorithm	Sensitivity	Specificity	MCC	AUROC	Acc.
5	Random Forest	60.16	61.7	0.22	0.66	60.93
7	Extra Tree	62.78	64.05	0.27	0.70	63.41
9	Random Forest	66.12	65.4	0.32	0.74	65.76
11	Random Forest	71.18	63.41	0.35	0.78	67.30
13	Random Forest	69.74	70.82	0.41	0.78	70.28
15	Extra Tree	76.51	66.94	0.44	0.79	71.73
17	Random Forest	71.54	70.46	0.42	0.79	71.00
19	Random Forest	66.58	69.74	0.36	0.79	68.16
21	Random Forest	68.93	70.28	0.39	0.78	69.60
23	Random Forest	71.09	70.37	0.41	0.78	70.73

Table 7.2: Testing Statistics of Binary Profile

Pattern Size	Algorithm	Sensitivity	Specificity	MCC	AUROC	Acc.
5	Random Forest	75.61	66.1	0.42	0.80	70.75
7	Extra Trees	77.18	67.92	0.45	0.81	72.55
9	Extra Trees	74.27	69.85	0.44	0.81	72.06
11	Extra Trees	76.87	69.75	0.47	0.82	73.31
13	**Extra Trees**	**78.05**	**68.56**	**0.47**	**0.83**	**73.30**
15	Extra Trees	78.09	68.06	0.46	0.83	73.08
17	Extra Trees	77.72	66.66	0.45	0.83	72.19
19	Extra Trees	78.37	66.61	0.45	0.83	72.49
21	Extra Trees	72.8	76.64	0.49	0.83	72.72

Table 7.3: Training Statistics for PSSM Profile

Pattern Size	Algorithm	Sensitivity	Specificity	MCC	AUROC	Acc.
5	Random Forest	84.01	75.97	0.6	0.88	79.99
7	Extra Trees	82.57	74.35	0.57	0.88	78.46
9	Extra Trees	79.04	77.87	0.57	0.88	78.46
11	Extra Trees	80.13	78.95	0.59	0.82	79.54
13	Extra Trees	78.95	78.86	0.58	0.87	78.91
15	Extra Trees	78.59	77.6	0.56	0.87	78.09
17	Extra Trees	78.41	77.78	0.56	0.87	78.09
19	Extra Trees	83.74	77.87	0.62	0.88	80.80
21	Extra Trees	79.86	84.64	0.65	0.87	82.25

Table 7.4: Testing Statistics for PSSM Profile

Dataset	Algorithm	Sensitivity	Specificity	MCC	AUROC	Acc.
Training	Extra Tree	72.52	73.59	0.19	0.82	73.55
Testing	Extra Tree	76.96	83.67	0.30	0.88	83.42
Training	KNN	44.19	97.08	0.38	0.74	95.08
Testing	KNN	66.85	97.48	0.57	0.86	96.38
Training	MLP	41.52	93.50	0.25	0.77	91.57
Testing	MLP	55.01	94.74	0.37	0.85	93.24
Training	RF	78.26	67.49	0.18	0.81	67.88
Testing	RF	76.96	79.31	0.26	0.87	79.23
Training	Ridge	64.22	60.67	0.10	0.68	60.80
Testing	Ridge	78.41	60.53	0.1	0.76	61.20
Training	SVC	73.45	69.54	0.17	0.80	69.69
Testing	SVC	79.77	69.39	0.20	0.86	69.78

Table 7.5: Statistics of Realistic Data (Pattern 13, PSSM)

7.4.1 Running the best parameters on realistic data

After obtaining the best pattern length, algorithm and its pertinent parameters, we ran the algorithm on the realistic data (details of which are mentioned in table 3.3). The statistics obtained from there are mentioned below in table 7.5

7.5 Receiver Operating Characteristic(ROC) Curve

In Machine Learning, performance measurement is a vital task. So when it comes to a classification problem, we can count on an AUC - ROC Curve (if we have a balanced dataset). When we need to check or visualize the performance of the multi - class classification problem, we use AUC (Area Under The Curve) ROC (Receiver Operating Characteristics) curve. It is one of the most important evaluation metrics for checking any classification model's performance. It is also written as AUROC (Area Under the Receiver Operating Characteristics).

The ROC curve is created by plotting the true positive rate (TPR)(Sensitivity) against the false positive rate (FPR)(1-Specificity) at various threshold settings.

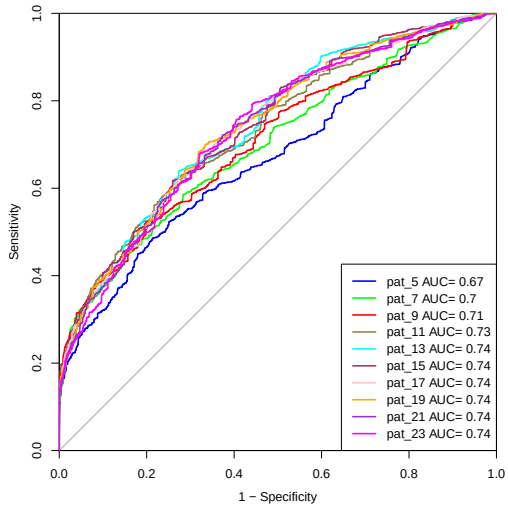

Figure 7.4: ROC Curve for Binary Profile (Training Data)

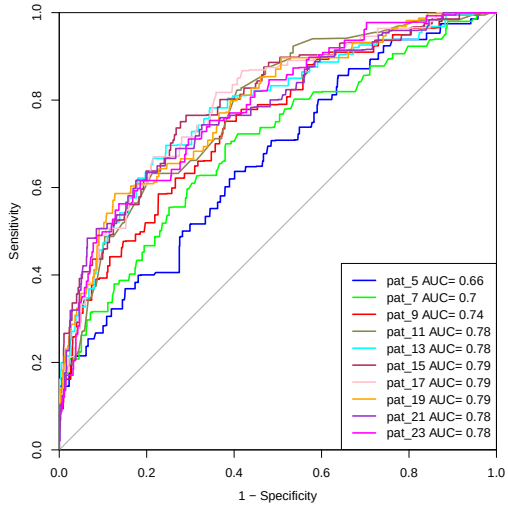

Figure 7.5: ROC Curve for Binary Profile (Test Data)

The false-positive rate is also known as the fall-out rate.

Figures 7.4 and 7.5 represent the ROC curves for training and testing data respectively corresponding to binary profiles for different pattern sizes.

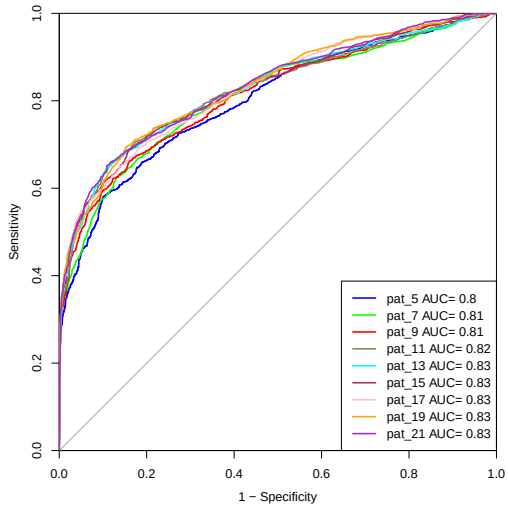

Figure 7.6: ROC Curve for PSSM Profile (Training Data)

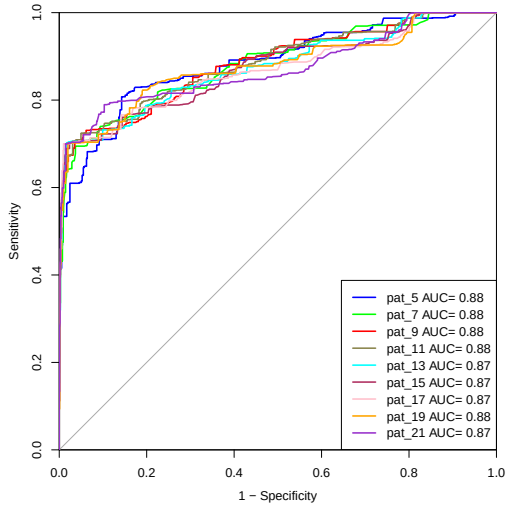

Figure 7.7: ROC Curve for PSSM Profile (Test Data)

Figures 7.6 and 7.7 represent the ROC curves for training and testing data respectively corresponding to PSSM profiles for different pattern sizes.

7.6 Feature Reduction and Selection Methods on the pat13

PCA and RFE were applied on the dataset. Since pattern 13 in PSSM profile gave the best results, we would apply feature selection methods on the same parameters. Any number of prinicipal components above 224 retained more than 96% variance. So the original dataset of 273 features was decomposed to 224 features using PCA. The results obtained are enumerated in table 7.6.

Dataset	**Algorithm**	**Sensitivity**	**Specificity**	**MCC**	**AUROC**
Training	Extra Tree	80.68	78.02	0.59	0.87
Testing	Extra Tree	66.41	83.81	0.51	0.83

Table 7.6: PCA with 224 features

RFE was applied by limiting the feature size to 224 as well. The results obtained are enumerated in tabel 7.7.

Dataset	**Algorithm**	**Sensitivity**	**Specificity**	**MCC**	**AUROC**
Training	Extra Tree	77.09	69.78	0.47	0.83
Testing	Extra Tree	80.13	78.32	0.58	0.88

Table 7.7: RFE with 224 features

8 Discussion and Conclusion

As evident from the tables appended before, pattern of size 13 for PSSM profiles works the best and gives AUROC of 0.83 with 78.05 sensitivity and 68.56 specificity. The accuracy in this case is 73.30%. The statistics quoted above are for balanced dataset.

For realistic data, i.e. without under-sampling, Extra Tree performs the best with 0.82 AUROC and 73.55% accuracy. The difference in performance is not very much but running the gridsearch on realistic data would have been computationally very expensive. Undersampling led to fast optimisation and tuning of all the algorithms. The best algorithm and parameters from the undersampled dataset was then used on realistic data. This saved the computational time.

As discussed, having more features can increase the computational time exponentially. So feature reduction methods were tried. Upon trying PCA, which reduces the dimensionality, the AUROC improved to 0.87. This shows that initially there was some redundancy in the dataset which led to the model learning a few of noisy data points. PCA deals with it as when we choose the first m highest eigenvalues we go on to reduce the noise of data, as higher eigenvalue corresponds to less noise in that principal component.

UDP is a relevant ligand as dilineated in section 2. A predictive analysis for this important ligand might open a gateway to important drug discoveries.

References

[1] Jie Sun and Ke Chen. "NSiteMatch: Prediction of Binding Sites of Nucleotides by Identifying the Structure Similarity of Local Surface Patches". In: *Computational and mathematical methods in medicine* 2017 (2017).

[2] Nitish K Mishra and Gajendra PS Raghava. "Prediction of FAD interacting residues in a protein from its primary sequence using evolutionary information". In: *BMC bioinformatics* 11.1 (2010), S48.

[3] Hifzur R Ansari and Gajendra PS Raghava. "Identification of NAD interacting residues in proteins". In: *BMC bioinformatics* 11.1 (2010), p. 160.

[4] Jagat S Chauhan, Nitish K Mishra, and Gajendra PS Raghava. "Prediction of GTP interacting residues, dipeptides and tripeptides in a protein from its evolutionary information". In: *BMC bioinformatics* 11.1 (2010), p. 301.

[5] Dong-Jun Yu et al. "TargetATPsite: A template-free method for ATP-binding sites prediction with residue evolution image sparse representation and classifier ensemble". In: *Journal of computational chemistry* 34.11 (2013), pp. 974–985.

[6] Mariana Babor et al. "Prediction of transition metal-binding sites from apo protein structures". In: *Proteins: Structure, Function, and Bioinformatics* 70.1 (2008), pp. 208–217.

[7] Xiuzhen Hu et al. "Recognizing metal and acid radical ion-binding sites by integrating ab initio modeling with template-based transferals". In: *Bioinformatics* 32.21 (2016), pp. 3260–3269.

[8] Ke Chen, Marcin J Mizianty, and Lukasz Kurgan. "Prediction and analysis of nucleotide-binding residues using sequence and sequence-derived structural descriptors". In: *Bioinformatics* 28.3 (2011), pp. 331–341.

[9] Dong-Jun Yu et al. “Designing template-free predictor for targeting protein-ligand binding sites with classifier ensemble and spatial clustering”. In: *IEEE/ACM transactions on computational biology and bioinformatics* 10.4 (2013), pp. 994–1008.

[10] Nadhipuram V Bhagavan. *Medical biochemistry*. Academic press, 2002.

[11] John W. Pelley. “8 - Gluconeogenesis and Glycogen Metabolism”. In: *Elsevier’s Integrated Review Biochemistry (Second Edition)*. Ed. by John W. Pelley. Second Edition. Philadelphia: W.B. Saunders, 2012, pp. 67–73. ISBN: 978-0-323-07446-9. DOI: https://doi.org/10.1016/B978-0-323-07446-9.00008-8. URL: http://www.sciencedirect.com/science/article/pii/B9780323074469000088.

[12] Ayako Kataoka et al. “Involvement of vasodilator-stimulated phosphoprotein in UDP-induced microglial actin aggregation via PKC-and Rho-dependent pathways”. In: *Purinergic signalling* 7.4 (2011), pp. 403–411.

[13] Shota Kobayashi et al. “UDP-induced relaxation is enhanced in aorta from female obese Otsuka Long–Evans Tokushima Fatty rats”. In: *Purinergic signalling* 14.1 (2018), pp. 91–96.

[14] Michael Oellerich and Amitava Dasgupta. *Personalized immunosuppression in transplantation: Role of biomarker monitoring and therapeutic drug monitoring*. Elsevier, 2015. Chap. 5, pp. 109–124.

[15] Helen M Berman et al. “The protein data bank”. In: *Nucleic acids research* 28.1 (2000), pp. 235–242.

[16] Vladimir Sobolev et al. “Automated analysis of interatomic contacts in proteins.” In: *Bioinformatics (Oxford, England)* 15.4 (1999), pp. 327–332.

[17] Robbie P Joosten et al. “A series of PDB related databases for everyday needs”. In: *Nucleic acids research* 39.suppl_1 (2010), pp. D411–D419.

[18] Wolfgang Kabsch and Christian Sander. “Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features”. In: *Biopolymers* 22.12 (1983), pp. 2577–2637.

[19] Weizhong Li and Adam Godzik. “Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences”. In: *Bioinformatics* 22.13 (2006), pp. 1658–1659.

A Appendix: Other Projects

During my internship at IIIT, apart from my main project which has been explained in-detail in this thesis, I worked on the following two projects:

A.0.1 Pfeature

In order to facilitate scientific community, Raghava's Lab has developed a software package that computes more than 50,000 features, important for predicting function of a protein and its residues. It has five major modules for computing; composition-based features, binary profiles, evolutionary information, structure-based features and patterns. The composition-based module allows user to compute; i) simple compositions like amino acid, dipeptide, tripeptide; ii) Properties based compositions; iii) Repeats and distribution of amino acids; iv) Shannon entropy to measure the low complexity regions; iv) Miscellaneous compositions like pseudo amino acid, autocorrelation, conjoint triad, quasi-sequence order. Binary profile of amino acid sequences provides complete information including order of residues or type of residues; specifically, suitable to predict function of a protein at residue level. Pfeature allows one to compute evolutionary information-based features in form of PSSM profile generated using PSIBLAST. Structure based module allows computing structure-based features, specifically suitable to annotate chemically modified peptides/proteins. Pfeature also allows generating overlapping patterns and feature from whole protein or its parts (e.g., N-terminal, C-terminal). In summary, Pfeature comprises of almost all features used till now, for predicting function of a protein/peptide including its residues. Availability: It is available in form of a web server, named as Pfeature (https://webs.iiitd.edu.in/raghava/pfeature/), as well as python library and standalone package (https://github.com/raghavagps/Pfeature) suitable for Windows, Ubuntu, Fedora, MacOS and Centos based operating system. The features generated by Pfeature have been shown in figure A.1. In addition to the webserver

executables and standalone versions are also available online for the scientific community:

A.0.2 Pfeature Scripts

More than 120 python scripts have been developed for computing wide range of features, visit GitHub website for detail. These scripts can be downloaded as a zip file from here User needs to unzip this file to create a folder of python scripts. . User can run these scripts by typing following command ”python script.py –h”, which will provide help on usage of python script.

A.0.3 Python Libraries

In order to develop method for annotating proteins in python user may wish to call python functions for speed optimization. We have also developed python library of Pfeature, which can be downloaded from here. The prerequisite to run the python library is pandas, numpy and python version 3.6 and above. The zipped file can be uncompressed. There is a file inside Pfeature folder named “setup.py”, which helps in installation of library using command “python setup.py install”. This Pfeature library consists of 120 functions for descriptor generation. These functions can easily be imported and compute thedesired features by passing the required arguments.

A.0.4 Pfeature Executables

One of the challenges in computing features of protein is that there are huge number of features. In case, we wish to compute all type of features we need to run more than 100 python script, which is time consuming. We are inspired with the field of chemoinformatics where software compute wide range of chemical descriptors in a single command, for example PaDEL compute more than 10000 chemical descriptors in a single command. Thus, we

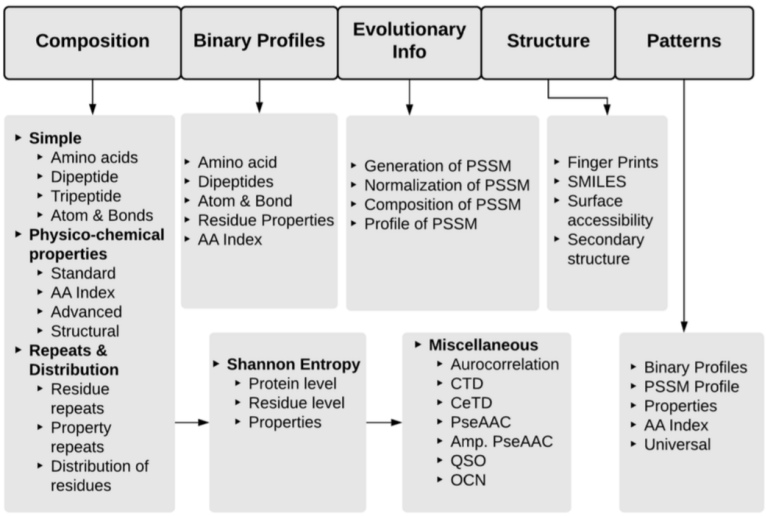

Figure A.1: Summary of Features that can be computed using Pfeature

have developed a standalone version of Pfeature which can compute thousands of protein descriptors in a single command. To cover maximum user, we have developed executables for Windows, Ubuntu, Fedora, MacOS and Centos based operating system. These executables can be downloaded from https://github.com/raghavagps/Pfeature/tree/master/exec/.

A.1 SAMBinder

S-adenosyl-L-methionine (SAM) is one of the important cofactor present in the biological system and play a key role in many diseases. There is a need to develop a method for predicting SAM binding sites in a protein for designing drugs against SAM associated disease. Best of our knowledge, there is no method that can predict the binding site of SAM in a given protein sequence. Result: This manuscript describes a method SAMbinder, developed for predicting SAM binding sites in a protein from its primary sequence. All models were trained, tested and evaluated on 145 SAM binding protein chains where no two chains have more than 40% sequence similarity. Firstly, models were developed using different machine learning techniques on a balanced dataset contain 2188 SAM interacting and an equal number of non-interacting residues. Our Random Forest based model developed using binary profile feature got maximum MCC 0.42 with AUROC 0.79 on the validation dataset. The performance of our models improved significantly from MCC 0.42 to 0.61, when evolutionary information in the form of PSSM profile is used as a feature. We also developed models on realistic dataset contains 2188 SAM interacting and 40029 non-interacting residues and got maximum MCC 0.61 with AUROC of 0.89. In order to evaluate the performance of our models, we used internal as well as external cross-validation technique. Availability and implementation: https://webs.iiitd.edu.in/raghava/sambinder/

A.1.1 Standalone

Standalone of SAMbinder is Python-based and is available at the Github site. The user can download it from the site https://github.com/raghavagps/sambinder/. SAMbinder standalone version is also implemented in the docker technology. Complete usage of downloading the image and its implementation is provided in the docker manual “GPSRdocker” which can be downloaded for the website https://webs.iiitd.edu.in/gpsrdocker/.

B Appendix: Publications

Papers in Communication as on May 11, 2019

- SAMbinder: A web server for predicting SAM binding residues of a protein from its amino acid sequence
  - **Authors:** Piyush Agrawal, **Gaurav Mishra**, Gajendra P.S. Raghava
  - Paper in communication with Bioinformatics (Oxford Academic)
- Computing wide range of protein/peptide features from their sequence and structure
  - **Authors:** Akshara Pandey, Sumeet Patiyal, Anjali Lathwal, Chakit Arora, Dilraj Kaur, Anjali Dhall, **Gaurav Mishra**, Harpreet Kaur, Neelam Sharma, Shipra Jain, Salman Sadullah Usmani, Piyush Agrawal, Rajesh Kumar, Vinod Kumar, Gajendra P.S. Raghava.
  - Paper in communication with Plos One

  N.B.: The paper for the work enumerated in this thesis shall be communicated shortly.