Prediction of Membrane Proteins Using Machine Learning Approaches

Maqsood Hayat

PhD Thesis

Pakistan Institute of Engineering and Applied Sciences
Department of Computer and Information Sciences,
Islamabad, Pakistan
Prediction of Membrane Proteins Using Machine Learning Approaches

By

Maqsood Hayat

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer and Information Sciences

To
The Department of Computer and Information Sciences,
Pakistan Institute of Engineering and Applied Sciences,
Islamabad, Pakistan
2012
Abstract

Membrane proteins are the basic constituent of a cell that manage intra and extracellular processes of a cell. About 20-30% of genes of eukaryotic organisms are encoded from membrane proteins. In addition, almost 50% of drugs are directly targeted against membrane proteins. Owing to the significant role of membrane proteins in living organisms, the identification of membrane proteins with substantial accuracy is essential. However, the annotation of membrane proteins through conventional methods is difficult, sometimes even impossible. Therefore, membrane proteins are predicted from topogenic sequences using computational intelligence techniques. In this study, we conducted our research in two phases regarding the prediction of membrane protein types and structures. In Phase-I, regarding the prediction of membrane protein types, four different ways are explored in order to enhance true prediction.

In the first part of phase-I, membrane protein types are predicted using Composite protein sequence representation followed by the application of principal component analysis in conjunction with individual classifiers. In the second part, the notion of ensemble classification is utilized. In part three, an error correction code is incorporated with Support Vector Machine using evolutionary profiles (Position Specific Scoring Matrix) and SAAC based features. Finally, in part four, a two-layer web predictor Mem-PHybrid is developed. Mem-PHybrid accomplishes the prediction in two steps. First, a protein query is identified as a membrane or a non-membrane protein. In case of membrane protein, then its type is predicted.

In the second phase of this research, the structure of membrane protein is recognized as alpha-helix transmembrane or outer membrane proteins. In case of alpha-helix transmembrane proteins, features are explored from protein sequences by two feature extraction schemes of distinct natures; including physicochemical properties and compositional index of amino acids. Singular value decomposition is employed to extract high variation features. A hybrid feature vector is formed by combining the different types of features. Weighted Random Forest is then used as a classification algorithm. On the other hand, in case of outer membrane proteins, protein sequences are represented by Amino acid composition, PseAA composition, and SAAC along with their hybrid
models. Genetic programming, K-nearest neighbor, and fuzzy K-nearest neighbor are adopted as classification algorithms.

Through the simulation study, we observed that the prediction performance of our proposed approaches in case of both types and structures prediction is better compared to existing state of the arts/approaches. Finally, we conclude that our proposed approach for membrane proteins might play a significant role in Computational Biology, Molecular Biology, Bioinformatics, and thus might help in applications related to drug discovery. In addition, the related web predictors provide sufficient information to researchers and academicians in future research.
This thesis is carried out under the supervision of

Dr. Asifullah khan

Associate Professor
Department of Computer and Information Sciences,
Pakistan Institute of Engineering and Applied Sciences,
Islamabad, Pakistan

This work is financially supported by Higher Education Commission (HEC) of Pakistan
Under the indigenous 5000 Ph.D. fellowship program
17-5-3 (Eg3-045)/HEC/Sch/2006
DECLARATION

I declare that all material in this thesis which is not my own work has been identified and that no material has previously been submitted and approved for the award of a degree by this or any other university.

Signature: ____________________

Author’s Name: Maqsood Hayat

It is certified that the work in this thesis is carried out and completed under my supervision.

Supervisor:

Dr. Asifullah Khan
Associate Professor
DCIS, PIEAS, Islamabad.
Acknowledgements

Firstly, I would like to thank God Almighty for the blessings bestowed upon me, during this research work. He provided me the strength, determination and knowledge to accomplish my PhD research work. It is the special blessing of God Almighty to have coterie of talented teachers, praying parents and loving friends which all help to make my endeavors a success.

I would heartedly thank my supervisor Dr. Asifullah Khan, whose sincere guidance and efforts were always source of inspiration and strength for me. In addition, it was his close assistance that helped me to maneuver the technical hurdles I faced during my research. Further, his encouraging and kindling attitude during down times were real motivation to achieve the set milestone. Thus, I feel myself the most fortunate to work with Dr. Asifullah Khan.

I would also like to express my deepest gratitude for my loving parents and siblings whose prayers and support made this work a success. I am also thankful to my friends who have been very helping during my stay at PIEAS.

I am very thankful to Dr. Abdul Jalil and Dr. Abdul Majid for his kind support, gratitude, guidance and precious comments from time to time during this research.

Finally, I would like to thank the Higher Education Commission (HEC) of Pakistan, for the financial support provided through Indigenous 5000 PhD program with reference to the award letter number 17-5-3 (Eg3-045)/HEC/Sch/2006.

Maqsood Hayat
List of Publications

**JOURNAL PUBLICATIONS**

2012  

2012  
Maqsood Hayat and Asifullah Khan, Prediction of Membrane Protein Types by Using Dipeptide and Pseudo Amino Acid Composition based Composite Features, accepted in *IET Communications* (impact factor: 0.96).

2011  

2011  

2011  

2011  

2011  
Maqsood Hayat, and Asifullah Khan, Predicting Membrane Protein Types by Fusing Composite Protein Sequence Features into Pseudo Amino Acid Composition, *Journal of Theoretical Biology*, 271 10–17 (impact factor: 2.37).
CONFERENCE PUBLICATIONS

2010

Maqsood Hayat, and Asifullah Khan, Prediction of Membrane Protein Types using Pseudo-Amino Acid Composition and Ensemble Classification, *IEEE International Conference on Intelligence and Information Technology (ICIIT)*, Oct 28-30, Lahore.

2010

# Abstract

1. Introduction .................................................................................................................. 21
   1.1. Biological Membranes ........................................................................................... 21
   1.2. Membrane Protein ............................................................................................... 23
      1.2.1. Peripheral membrane proteins ....................................................................... 24
   1.2.2. Integral membrane proteins ............................................................................. 24
      1.2.2.1. Transmembrane proteins ............................................................................ 25
      1.2.2.2. Anchored Membrane Proteins ................................................................. 27
   1.3. Research Objectives and Contributions ................................................................. 27
   1.4. Structure of the Thesis .......................................................................................... 29

# Literature Survey

2.1. Membrane Protein Types .......................................................................................... 32

# Machine Learning

3.1. Database ..................................................................................................................... 41

4.1. Implemented Feature extraction Schemes ................................................................. 41
   3.2.1. Biochemical properties .................................................................................... 41
      3.2.1.1. Amino Acid (AA) Composition ................................................................. 41
      3.2.1.2. Pseudo Amino Acid (PseAA) Composition ............................................. 42
3.2.1.3. Split Amino Acid Composition .................................................. 44
3.2.1.4. Discrete Wavelet Transform .................................................. 44
3.2.2. Evolutionary Information ......................................................... 46
  3.2.2.1. Position Specific Scoring Matrix ........................................... 46
3.2.3. Physiochemical properties ....................................................... 48
  3.2.3.1. Composition and Translation .............................................. 48
  3.2.3.2. Composite Protein Sequence Representation ....................... 49
3.3. Feature Selection Techniques ..................................................... 51
  3.3.1. Principal Component Analysis ................................................ 52
  3.3.2. Neighborhood Preserving Embedding .................................... 52
  3.3.3. Minimum Redundancy Maximum Relevance ......................... 54
  3.3.4. Singular Value Decomposition .............................................. 55
3.4. Data Partition (Validation) .......................................................... 56
  3.4.1. Self-consistency Test .............................................................. 57
  3.4.2. Independent Dataset Test ....................................................... 57
  3.4.3. Jackknife Test ...................................................................... 57
  3.4.4. Sub-sampling Test ................................................................. 58
3.5. Classification Algorithms ............................................................ 58
  3.5.1. AdaBoost ........................................................................... 58
  3.5.2. Random Forest ..................................................................... 60
  3.5.3. Support Vector Machine ....................................................... 61
  3.5.4. Probabilistic Neural Network ............................................... 63
  3.5.5. K-Nearest Neighbor ............................................................. 65
  3.5.6. Fuzzy K-nearest Neighbor .................................................... 66
  3.5.7. Evidence theoretic K- Nearest Neighbor Classifier .................. 67
  3.5.8. Generalized Regression Neural Network ............................... 68
  3.5.9. Genetic Programming .......................................................... 69
3.6. Evaluation criteria ............................................................................................................. 73
3.6.1. Accuracy ..................................................................................................................... 74
3.6.2. Sensitivity and Specificity .......................................................................................... 74
3.6.3. Mathew’s Correlation Coefficient .............................................................................. 75
3.6.4. F-measure ................................................................................................................... 76
3.6.5. Q-statistics .................................................................................................................. 76
3.6.6. Receiver Operating Characteristic Curve ................................................................. 77

4. Prediction of Membrane Protein Types Using Composite Protein Sequence
   Representation ......................................................................................................................... 79
4.1. Introduction ....................................................................................................................... 79
4.2. Materials and Methods ...................................................................................................... 80
   4.2.1. Dataset ...................................................................................................................... 80
   4.2.2. Composite Protein Sequence Representation ............................................................ 80
   4.2.3. Principal Component Analysis ................................................................................ 81
4.3. Experimental Results and Discussion ............................................................................... 82
   4.3.1. Self-consistency Test ................................................................................................ 82
   4.3.2. Jackknife Test ........................................................................................................... 82
   4.3.3. Independent Dataset Test ........................................................................................ 84
   4.3.4. Individual Membrane Protein Types ...................................................................... 85
   4.3.5. Comparison with the Existing Approaches ............................................................. 87

5. Prediction of Membrane Protein Types Using Ensemble Classification ............................. 89
5.1. Introduction ....................................................................................................................... 89
5.2. Materials and Methods ...................................................................................................... 90
   5.2.1. Datasets .................................................................................................................... 90
   5.2.2. Feature Extraction Schemes ..................................................................................... 92
   5.2.3. Ensemble Classification ........................................................................................... 93
   5.2.3.1. Simple Majority Voting Based Ensemble ............................................................ 93
   5.2.3.2. GA Majority Voting Based Ensemble ................................................................. 95
5.3. Experimental Results and Discussion ................................................................. 95
  5.3.1. Dataset2 ............................................................................................................. 96
    5.3.1.1. Self-consistency Test ......................................................... 96
    5.3.1.2. Jackknife Test ................................................................. 96
    5.3.1.3. Independent Dataset Test .............................................. 97
    5.3.1.4. Hybrid Features vs. NPE based Selected Features .......... 98
    5.3.1.5. Ensemble through GA ...................................................... 101
    5.3.1.6. Individual Membrane Protein Types ......................... 103
    5.3.1.7. Comparison with the Existing Approaches ..................... 105
  5.3.2. Dataset3 ......................................................................................................... 106

6. Prediction of Membrane Protein Types Using Evolutionary Information and BCH Coding ................................................................. 108

  6.1. Introduction ....................................................................................................... 108
  6.2. Materials and Methods .................................................................................... 109
    6.2.1. Datasets ................................................................................................. 109
    6.2.2. Feature Extraction Schemes .............................................................. 109
      6.2.2.1. Position Specific Scoring Matrix ........................................ 109
    6.2.3. BCH Coding ......................................................................................... 110
    6.2.4. Proposed Method .................................................................................... 111
  6.3. Experimental Results and Discussion .............................................................. 114
    6.3.1. PSSM vs. Hybrid (PSSM+ SAAC) .............................................. 114
    6.3.2. Hybrid (PSSM+SAAC) ....................................................................... 116
      6.3.2.1. Lin-SVM vs. Poly-SVM ..................................................... 116
      6.3.2.2. Lin-SVM vs. RBF-SVM ............................................... 117
      6.3.2.3. Poly-SVM vs. RBF-SVM ............................................. 118
  7. Online web Predictor: Mem-PHybrid Predictor .................................................. 120

  7.1. Introduction ....................................................................................................... 120
  7.2. Materials and Methods ................................................................................... 121
7.2.1. Datasets .................................................................................................................... 121
7.2.2. Feature Extraction Schemes ..................................................................................... 122
  7.2.2.1. Physicochemical Properties ............................................................................... 122
7.3.3. Hybrid Model ........................................................................................................... 122
7.3. Web Server (Mem-PHybrid Predictor) ........................................................................... 123
7.4. Experimental Results and Discussion ............................................................................. 126
  7.4.1. Jackknife Test ........................................................................................................... 126
    7.4.1.1. Membrane vs. Non-membrane Proteins ............................................................ 126
    7.4.1.2. Membrane Protein Types ................................................................................... 127
  7.4.2. Independent Dataset Test ......................................................................................... 130
8. RF-TMH: Prediction of Transmembrane Helix Using Hybrid Features ....................... 132
  8.1. Introduction ..................................................................................................................... 132
  8.2. Materials and Methods .................................................................................................... 133
    8.2.1. Datasets .................................................................................................................... 133
    8.2.2. Feature Extraction schemes ...................................................................................... 133
      8.2.2.1. Physicochemical properties ............................................................................... 133
      8.2.2.2. Compositional Index .......................................................................................... 135
    8.2.3. Singular Value Decomposition ................................................................................ 136
    8.2.4. RF-TMH Proposed System ...................................................................................... 136
  8.3. Performance Measures .................................................................................................... 138
  8.4. Experimental Results and Discussion ............................................................................. 139
    8.4.1. Performance analysis at Protein level ...................................................................... 139
    8.4.2. Performance analysis at Segment level .................................................................... 140
    8.4.3. Performance analysis at residue level ....................................................................... 141
9. Prediction of Outer Membrane Proteins Using Fuzzy KNN ............................................ 143
  9.1. Introduction ..................................................................................................................... 143
  9.2. Materials and Methods .................................................................................................... 144
    9.2.1. Datasets .................................................................................................................... 144
9.2.2. Feature Extraction Schemes ................................................................. 144
9.2.3. Proposed System .................................................................................. 145
9.3. Experimental Results and Discussion ......................................................... 146
  9.3.1. OMPs vs. non-OMPs .......................................................................... 147
    9.3.1.1. GP ................................................................................................. 147
    9.3.1.2. KNN ............................................................................................. 148
    9.3.1.3. Fuzzy KNN ................................................................................... 149
  9.3.2. Discrimination of OMPs from Alpha-helix membrane and Globular proteins .... 150
  9.3.3. Comparison with the Existing approaches on DS1 .............................. 152
  9.3.4. Comparison with the Existing approaches on DS2 .............................. 154

10. Conclusions and Future Directions ............................................................. 155
  10.1. Membrane Protein Types .................................................................... 155
  10.2. Membrane Protein Structures .............................................................. 156
  10.3. Future Directions ................................................................................. 157
References ......................................................................................................... 158
List of Figures

Figure 1.1. Fluid mosaic model [1] ................................................................................................ 23

Figure 1.2. Classification of Membrane Proteins ........................................................................ 24

Figure 1.3. Graphical illustration shows the eight types of membrane proteins: (1) Type-I transmembrane; (2) Type-II transmembrane; (3) Type-III transmembrane; (4) Type-IV transmembrane; (5) Multipass transmembrane; (6) lipid chain-anchored membrane; (7) GPI-anchored membrane; and (8) peripheral membrane proteins. Reproduced from Chou and Shen (2007) with permission [9]. ............................................................................................................ 25

Figure 1.4. (a) shows the alpha helix and (b) represents beta-barrel transmembrane protein...... 27

Figure 3.1. Overall framework of Machine learning processes and sub-processes. ................. 40

Figure 3.2. DWT decomposition-tree at level four ...................................................................... 45

Figure 3.3. Graphical representation of SVD .............................................................................. 56

Figure 3.4. Classification of SVM ................................................................................................ 62

Figure 3.5. Framework of PNN .................................................................................................. 64

Figure 3.6. Crossover operations on trees ................................................................................... 72

Figure 3.7. Mutation operations on trees .................................................................................... 72

Figure 3.8. Shows the number of healthy and diseased patients ................................................ 77

Figure 3.9. Depicts the ROC curve ............................................................................................ 78

Figure 4.1. First principal component versus second principal component ............................. 81

Figure 4.2. Performance of classifiers using Jackknife test ......................................................... 83

Figure 4.3. MCC and F-measure of classifiers using Jackknife test ........................................... 84

Figure 4.4. Performance of classifiers for each type of membrane proteins using Jackknife test. 86

Figure 4.5. Performance of classifiers for each type of membrane proteins using Independent dataset test .............................................................................................................. 87

Figure 5.1. Framework of Ensemble Classification ..................................................................... 94

Figure 5.2. Performance of DWT-EnsC using GA for jackknife test ....................................... 102

Figure 5.3. Performance of Hybrid-EnsC using GA for jackknife test ....................................... 102
Figure 5.4. Accuracy of various ensemble classifiers for each type of membrane proteins using Jackknife test................................................................. 105

Figure 6.1. Architecture of merging BCH coding with SVM................................................. 112

Figure 7.1. Screenshot (a) Display illustrates the Main page of Mem-PHybrid predictor (b) Display showing the input query of protein sequence (c) Display depicting the input protein query as a membrane or non-membrane (d) Display showing the predicted type of membrane protein. .............................................................................................................................. 125

Figure 7.2. Performance of various classifiers for membrane and non-membrane proteins using Jackknife test on dataset1 and dataset2.............................................................................................................................. 127

Figure 7.3. ROC curve for SVM, RF, and ET-KNN on dataset1 ........................................... 129

Figure 7.4. ROC curve for SVM, RF, and ET-KNN on dataset2 ........................................... 130

Figure 8.1. Framework of the proposed approach ................................................................. 137

Figure 9.1. Framework of the proposed approach ................................................................. 146

Figure 9.2. ROC Curve for KNN and Fuzzy KNN using SAAC on DS1 ................................. 150

Figure 9.3. ROC Curve for KNN and Fuzzy KNN using SAAC on DS2 ................................. 151

Figure 9.4. Performance comparison of our proposed approach with already existing models on DS1 ................................................................................................................................. 153
List of Tables

Table 3.1 Division of amino acids according to their physicochemical properties ....................... 49
Table 3.2 Deriving features based division of amino acids [140] ........................................................ 50
Table 3.3 Eisenberg hydrophobicity index value of amino acids [142] .............................................. 51
Table 3.4. Confusion Matrix .......................................................................................................... 74
Table 3.5. Relation between classifier C_i and classifier C_j ............................................................ 76
Table 4.1. Performance of classifiers using Jackknife test ............................................................... 83
Table 4.2. Performance of classifiers using Independent dataset test ............................................. 83
Table 4.3. Performance of classifiers for each membrane protein type using Jackknife test ......... 85
Table 4.4. Performance of classifiers for each membrane protein type using Independent dataset test ................................................................. 85
Table 4.5. Comparative analysis between the proposed approach and existing approaches ......... 88
Table 5.1. Success rates of classifiers on Jackknife test using individual feature extraction strategies. ........................................................................................................ 97
Table 5.2. Success rates of classifiers on Independent dataset test using individual feature extraction strategies. ........................................................................................................ 98
Table 5.3. Classification outcomes of the Jackknife test using hybrid entire and reduced features. ........................................................................................................ 99
Table 5.4. Classification outcomes of the Independent dataset test using hybrid entire and reduced features. ........................................................................................................ 100
Table 5.5a-b. Success rates of ensemble classifiers for each type of membrane proteins using Jackknife test ........................................................................................................ 104
Table 5.6. Comparative analysis between the proposed approach and existing approaches on dataset2 ........................................................................................................ 106
Table 5.7. Comparative analysis between the proposed approach and existing approaches on dataset3 ........................................................................................................ 107
Table 6.1. A unique binary code for each class ............................................................................ 112
Table 6.2. Prediction performance of SVMs with and without BCH coding on dataset1 .......... 115
Table 6.3. Performance of SVMs with and without BCH coding on dataset 1. ........................... 115
Table 6.4. Prediction performance of SVMs with and without BCH coding on dataset 2 .......... 118
Table 6.5. Performance of SVMs with and without BCH coding on dataset 2. ......................... 119
Table 7.1. Performance of classifiers for membrane and non-membrane protein using Jackknife test on dataset 1 and dataset 2. ........................................................................................................... 127
Table 7.2. Success rates of classifiers using Jackknife test on dataset 1 and dataset 2 .......... 128
Table 7.3. Success rates of classifiers for each type of membrane proteins using Jackknife test on both datasets. ......................................................................................................................... 129
Table 7.4. Success rates of classifiers using Independent dataset test on both datasets. .......... 131
Table 7.5. Success rates of classifiers for each type of membrane proteins using Independent dataset test on both datasets. .............................................................................................................. 131
Table 8.1. Physicochemical properties of amino acids ............................................................... 134
Table 8.2. Performance comparison of proposed method with the existing methods .......... 141
Table 9.1. Success rates of GP, KNN, and Fuzzy KNN using 5-fold cross-validation .......... 148
Table 9.2. Prediction performance of Fuzzy KNN ........................................................................... 151
Table 9.3. Comparative analysis between the proposed approach and existing approaches on DS1 ......................................................................................................................................................... 153
Table 9.4. Comparative analysis between the proposed approach and existing approaches on DS2 ......................................................................................................................................................... 154
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>Amino Acid</td>
</tr>
<tr>
<td>BBA</td>
<td>Basic Belief Assignment</td>
</tr>
<tr>
<td>BCH</td>
<td>Bose, Ray-Chaudhuri, Hocquenghem</td>
</tr>
<tr>
<td>CDA</td>
<td>Covariant Discriminant Algorithm</td>
</tr>
<tr>
<td>CPSR</td>
<td>Composite Protein Sequence Representation</td>
</tr>
<tr>
<td>DWT</td>
<td>Discrete Wavelet Transform</td>
</tr>
<tr>
<td>ET-KNN</td>
<td>Evidence Theoretic K-Nearest Neighbor</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>GP</td>
<td>Genetic Programming</td>
</tr>
<tr>
<td>GO</td>
<td>Gene Ontology</td>
</tr>
<tr>
<td>GRNN</td>
<td>Generalized Regression Neural Network</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>KNN</td>
<td>K-Nearest Neighbor</td>
</tr>
<tr>
<td>MCC</td>
<td>Mathew’s Correlation Coefficient</td>
</tr>
<tr>
<td>mRMR</td>
<td>Minimum Redundancy Maximum Relevance</td>
</tr>
<tr>
<td>NPE</td>
<td>Neighborhood Preserving Embedding</td>
</tr>
<tr>
<td>OET-KNN</td>
<td>Optimized Evidence Theoretic K-Nearest Neighbor</td>
</tr>
<tr>
<td>OMPs</td>
<td>Outer Membrane Proteins</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>PNN</td>
<td>Probabilistic Neural Network</td>
</tr>
<tr>
<td>PseAA</td>
<td>Pseudo Amino Acid</td>
</tr>
<tr>
<td>PSSM</td>
<td>Position Specific Scoring Matrix</td>
</tr>
<tr>
<td>PSI-BLAST</td>
<td>Position Specific Iterated BLAST</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Base Function</td>
</tr>
<tr>
<td>RF</td>
<td>Random Forest</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristic</td>
</tr>
<tr>
<td>SAAC</td>
<td>Split Amino Acid Composition</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
</tbody>
</table>

20
1. Introduction

Cell is the primary unit of life, all living things are created from one to trillions of cells, which can be categorized as prokaryotic or eukaryotic. Cell performs various functions such as molecule transportation, reproduction, and energy conversion. Cell may have varying shapes and sizes based on the organisms, however, majority of their functions are similar in all organisms.

Proteins are responsible for various cellular processes and are composed of 20 different amino acids. Protein functions depend on the particular chain of constituent amino acids and their shape. The fundamental component of a cell is membrane protein, which manages the intra and extracellular processes of a cell. Membranes that surround the cell in all the three domains of life such as bacteria, archea, and eukaryote are called plasma membrane. In addition, eukaryotic cell consists of an internal membrane called organelles. Membrane is a physical barrier that simultaneously protects the cell from external molecules and prevents leakage of internal molecules. Membrane proteins perform various biological functions according to their locations in a cell. Membrane can be categorized into nine distinct classes according to their subcellular localization including (a) chloroplast; (b) nucleus; (c) plasma membrane; (d) endoplasmic reticulum; (e) mitochondria; (f) lysosome; (g) vacuole; (h) Golgi apparatus; and (i) peroxisome.

1.1. Biological Membranes

Cell membrane is an essential component of a cell that works like a protector and covers the outer surface of a cell. It is a biological membrane that is known as plasma membrane or phospholipids bilayer. Cell membrane is mostly composed of lipids, proteins, and carbohydrates, which help to preserve the interior of a cell from the external
Cell membrane is selectively permeable lipid bilayer, which contains the outer layer of a cell, and has specific channels and pumps to control the composition of ions and molecules of intracellular medium. There are different kinds of lipids among which glycol and phospholipids are the most popular ones. These lipids can be neutral, zwitterionic or negatively charged. The most common attribute in all the lipids is the amphipathic molecules with hydrophilic head groups and hydrophobic hydrocarbon tails or fatty acids.

The essential materials can be transported across the membrane through passive or active transport. In passive transport, the substances moving across the membrane can be either chemical or electrical equilibrium, which is occurring without cellular energy. In case of active transport, the substances move against their electrical or chemical gradient. In fact, the substances are moving away from equilibrium, therefore, they require energy. There are two types of active transport utilized by cells; direct or indirect. Both types require integral membrane proteins. The cell membrane plays an important role in anchoring the cytoskeleton to give shape to a cell. Further, cell membrane has responsibility to attach the cell to the extracellular matrix and other cells, as well as group cells together to form tissues. In addition, cell membrane is responsible for the transportation of small molecules, from dense molecules region to light molecules region, across cell membrane. Plants, fungi, and some bacteria have a cell wall surrounding the cell membrane. So, the structure of cell membrane is dynamic with an equal combination of lipids and proteins, which can move laterally as represented by the fluid mosaic model in Figure 1.1[1].
1.2. Membrane Protein

Membrane proteins are responsible for most of the dynamic processes of a cell. Their preliminary task is signaling between cells, transportation across cell membranes, energy transduction process, and cell adhesion. Researchers are convinced that membrane proteins encode almost 20-30% genes of eukaryotic organisms [2; 3; 4]. In addition, human genomes are encoded from 8,000 membrane proteins, approximately [5]. Therefore, information regarding the structure and function of membrane proteins has massive significance in biological and pharmaceutical research. In addition, membrane proteins are the prime drug targets in various pharmaceutical industries, because drugs are primarily utilized by human and veterinarian [6]. Besides, Membrane proteins constitute approximately 50% of potential targets for novel drugs [5; 7; 8]. Additionally, more than half of proteins interact with membrane proteins. Membrane proteins can be classified into two main categories: integral (intrinsic) and peripheral (extrinsic) membrane proteins as shown in Figure 1.2. However, some biomembranes may have qualities of both.
1.2.1. **Peripheral membrane proteins**

Peripheral membrane proteins are loosely attached to the outer surface of a biological membrane. It indirectly communicates with the integral membrane through the combination of hydrophobic, electrostatic, and other non-covalent. Direct communication is performed with the lipid polar head group. On the other hand, peripheral membrane is not in contact with the hydrophobic core of membrane. Peripheral membrane can be released without disrupting membrane by modification in its salt contents or elevated pH.

1.2.2. **Integral membrane proteins**

Integral membrane proteins are strongly associated with biological membranes and require a detergent or an organic solvent to be released. Integral membrane protein has one or more polypeptide segments embedded in the lipid bilayer. The segments traverse the entire membrane so that the protein must have both the regions of polar end and non-polar end. Therefore, in integral membrane proteins some residues have hydrophobic side chains interacting with the fatty acyl groups of the membrane phospholipids. Integral membrane proteins can further be categorized into two types; transmembrane proteins and anchored membrane proteins.
1.2.2.1. Transmembrane proteins

Transmembrane proteins cover the entire surface of biological membrane. The transmembrane proteins contain one or more hydrophobic segments, and hence are relatively easily differentiable from non-membrane proteins. The segment of the transmembrane protein, embedded in the lipid bilayer, has non-polar residues. Usually, these residues, hydrophobic and stable inside the bilayer, build a coil or helix. Transmembrane proteins have three regions; the region in the bilayer, the region inside the cell (intracellular region), and the region outside the cell (extracellular region).

Like other proteins, transmembrane proteins also perform various biological functions namely, energy generation, signal transduction, transportation of solutes across the membrane, maintenance of ionic, proton gradients, metabolism, and the production of the fatty acids. [3]. There are two different types of transmembrane membrane proteins: Alpha-helix single pass and multipass alpha or beta.

![Figure 1.3. Graphical illustration shows the eight types of membrane proteins: (1) Type-I transmembrane; (2) Type-II transmembrane; (3) Type-III transmembrane; (4) Type-IV transmembrane; (5) Multipass transmembrane; (6) lipid chain-anchored membrane; (7) GPI-anchored membrane; and (8) peripheral membrane proteins. Reproduced from Chou and Shen (2007) with permission [9].](image)
.Alpha-helical transmembrane proteins

Alpha-helix transmembrane proteins exist in every cell membrane. Alpha-helix membrane proteins are usually made of hydrophobic residues and their side chains can organize the van der Waals interactions with the fatty acids in membrane core. Alpha-helix single pass can further be divided into subtypes namely, type-I single-pass transmembrane, type-II single-pass transmembrane, type-III single-pass transmembrane and type-IV single-pass transmembrane proteins are shown in Figure 1.3. Type-I single-pass transmembrane protein has an extracellular on N-terminus and cytoplasmic on C-terminus, while type-II transmembrane protein has lack of cleavable endoplasmic reticulum signal sequence and presents its hydrophilic C-terminus on exoplasmic side and hydrophilic N-terminus on cytoplasmic side. In both types, the polypeptides cross the lipid bilayer only once and contain 20-25 hydrophobic amino acids. Similarly, multipass alpha or beta is further classified into two parts; alpha-helix multipass and beta-barrels multipass are illustrated in Figure 1.4. In alpha-helix multipass membrane protein, the polypeptide passes the lipid bilayer several times. It is also called tetraspanins. Tetraspanins are found most frequently in those proteins, which have spanned the membrane four or seven times. If the multipass transmembrane protein has an even number of alpha helices, then its C- and N- termini are on the same side of the membrane.

Beta-barrel Transmembrane Proteins

Beta-barrel transmembrane proteins typically constitute the outer membranes of Gram-negative bacteria, chloroplasts, and mitochondria [10]. Beta-barrel transmembrane proteins are constructed from an even number of anti-parallel strands where each strand has hydrogen bonds facing towards its neighbor strand in the primary sequence [11]. In transmembrane region, the hydrophilic residues are facing to the interior of the barrel whereas the hydrophobic residues are oriented towards the exterior of the barrel or surrounding the lipids. This alternative pattern reveals that one amino acid is non-polar and hydrophobic whereas the second is polar and hydrophilic in topogenic sequence. In addition, approximately 2-3% of genomes are encoded from beta-barrel membrane proteins [12; 13], however, the identification of beta-barrel membrane protein is difficult.
because of uncertain number and unclear pattern. The other reason is, beta-strands consist a smaller number of residues and have an extensive range of interactions. In addition, few structure patterns of beta-barrel membrane proteins are available compared to alpha-helical transmembrane proteins.

![Alpha Helix and Beta-Strand](image)

(a) (b)

**Figure 1.4.** (a) shows the alpha helix and (b) represents beta-barrel transmembrane protein

### 1.2.2.2. Anchored Membrane Proteins

Anchored membrane proteins have two types: lipid chain-anchored membrane proteins and glycoposphatidylinositol (GPI) anchored membrane proteins are depicted in Figure 1.3. Lipid chain-anchored membrane protein is attached to the bilayer only whereas GPI-anchored membrane protein is adhered to the membrane by a GPI-anchor. GPI-anchored membrane proteins have lack of transmembrane region, and have no cytoplasmic tail; therefore, they are located on the outer surface of the plasma membrane.

### 1.3. Research Objectives and Contributions

The number of protein sequences in databanks is consistently increasing due to the rapid technological advancement in molecular biology. In 1986, the total number of protein sequences in Swiss-Prot was 3,939 [9]. According to the version 52.4 released on May 1, 2007 at http://www.ebi.ac.uk/swissprot/, the number of protein sequences approached
265,950. It reveals that more than 67 times increase in the number of protein sequences reported in 1986. So, the identification and annotation of this unprocessed data are the prime challenges in the field of Bioinformatics and Computational Biology. Conventional experimental methods usually offer accurate results; however, for some of the species, proteomics and microscopic detection are almost impossible to be conducted. This is mainly due to inherent complications; experimental methods are restricted to a limited number of proteins. Therefore, the demand for developing an automatic, fast, and reliable computational model for discriminating uncharacterized proteins is increasing.

The objective of this research is to predict membrane proteins from their primary amino acid sequences by using various machine learning approaches. These sequences contain motifs (patterns) and useful information (features), which may help in the identification of membrane proteins. However, the challenging task is to extract informative descriptors from topogenic sequences and to exploit the discrimination power of these features to correctly identify the membrane protein. For this purpose, we have introduced and conducted a comparative study of various methods for prediction of membrane proteins. Our research work span over two phases. In the first phase, we predict the types of membrane proteins and their structure in the second phase.

Phase-I: In the first phase, membrane protein types are predicted.

- The membrane protein sequences are expressed using various feature extraction schemes including biochemical, physicochemical, and evolutionary profiles methods.
- To avoid the classification algorithm from unnecessary and irrelevant features, features selection methods are applied, this helps in reducing the computational cost along with improvement in performance of the classifiers.
- Several classification algorithms are employed in the form of individual and ensemble classifications.
- The performance of the classifiers is evaluated using widely recognized statistical cross validation tests namely, self-consistency, independent dataset, and jackknife tests.
Various performance measures including accuracy, sensitivity, specificity, Mathew’s Correlation Coefficient (MCC), F-measure, Q-statistics, and Receiver operating characteristic (ROC) curve are adopted to evaluate the performance of the classifiers.

A two-Layer web predictor “Mem-PHybrid” is developed.

Phase-II: In this phase, membrane protein structures are identified as alpha helix transmembrane and outer membrane proteins.

A. Transmembrane alpha-helix
   - Compositional indices and physicochemical properties of amino acids are used to represent protein sequences.
   - Feature space is reduced through single value decomposition (SVD).
   - Weighted Random Forest (RF) is used as a classification algorithm.
   - The performance of the classifiers is estimated by 10-fold cross validation.
   - Performance is measured in terms of per protein, per segment, and per residue levels.

B. Outer membrane proteins
   - Three biochemical methods including Amino acid (AA) composition, Pseudo amino acid (PseAA) composition, and Split amino acid composition (SAAC) as well as hybrid versions of these methods are used for feature representation.
   - Three different classification algorithms namely, Genetic Programming (GP), K nearest neighbor (KNN), and Fuzzy K nearest neighbor are applied.
   - 5-fold cross validation is used for the evaluation of the classifier’s performance.

1.4. Structure of the Thesis

In Chapter 2, some related work about membrane proteins is presented. Literature survey reveals that membrane proteins prediction is performed by considering its types
and structures. The literature further identifies membrane protein structure as alpha-helix transmembrane and outer membrane proteins.

In Chapter 3, general background regarding machine learning processes is discussed that we have utilized in this research. The processes consist of database, feature extraction and feature selection techniques, data partitioning, classification algorithms, and evaluation criteria.

Chapter 4 commences with the first phase of this research, which presents the contribution regarding membrane protein types. In this chapter, we suggest a prediction model for membrane protein types. In this model, Composite protein sequence representation (CPSR) is used as feature extraction scheme whereas principal component analysis is applied as feature selection technique. Support vector machine (SVM), probabilistic neural network (PNN), and generalized regression neural network (GRNN) are adopted as classification algorithms. Three cross validation tests are performed to assess the performance of the classification algorithms. The performance of the classifiers is measured in terms of accuracy, sensitivity, specificity, F-measure, and MCC.

In Chapter 5, we predict membrane protein types by adopting the notion of ensemble classification. The features are extracted using biochemical methods based on discrete wavelet transform (DWT), PseAA composition, and SAAC. Combinations of these methods are also investigated. In addition, hybrid features are reduced using neighborhood preserving embedding and comparisons are shown between regular features and reduced features. AdaBoost, RF, SVM, PNN, and KNN are utilized as classification algorithms. Two kinds of ensemble classification are considered: simple majority voting based ensemble and Genetic algorithm (GA) based ensemble. In order to assess the performance of the individual as well as ensemble classification algorithms, we performed self-consistency, jackknife, and independent dataset tests.

In Chapter 6, membrane protein sequences are expressed using evolutionary profiles and biochemical properties. A hybrid model is developed by fusing PSSM and SAAC based features. The behavior of the three kernels of SVM is investigated. In addition, error correction code such as Bose, Ray-Chaudhuri, and Hocquenghem (BCH) code is incorporated with SVM.
In Chapter 7, a two-layer predictor (Mem-PHybrid) is developed. The model is developed on the basis of hybrid feature space. The hybrid features are the fusion of physicochemical properties (composition and translation) and SAAC based features. Minimum redundancy and maximum relevance (mRMR) feature selection technique is employed to reduce the dimension of a feature space. SVM, ET-KNN, and RF are adopted as classification algorithms. Jackknife and independent dataset tests are used for evaluating the performance of the classification algorithms. The performance of the classifiers is assessed using accuracy, sensitivity, specificity, F-measure, and MCC. In addition, ROC curve is used to show the tradeoff between different thresholds.

Chapter 8 starts with the discussion of the Phase-II research, membrane protein structure. In this chapter, transmembrane alpha-helices are predicted. Physicochemical properties and compositional index of amino acids are used as feature extraction schemes. SVD is employed as features selection technique, whereas weighted RF is used as a classifier. 10-fold cross validation is applied to evaluate the performance of the classifier. The performance is measured in terms of per protein, per segment, and per residue levels.

In Chapter 9, outer membrane proteins are predicted. Three biochemical methods AA composition, PseAA composition, SAAC, and combination of these methods are used as feature extraction schemes. GP, KNN, and Fuzzy KNN are adopted as classification algorithms. 5-fold cross validation test is used for evaluating the performance of the classifiers.

Chapter 10 concludes this study having focused on the major achievements. It also highlights some aspects that still need consideration in future research.
In order to assess all the gene functions and expression motifs as well as to comprehend protein biological functions, one needs to carry out studies regarding the protein structure. The identification of membrane proteins is considered one of the most important problems in Structural, Functional Genomics, and Computational Biology [14]. In the last few decades, prediction of membrane proteins has been performed by considering its type and structure. Literature survey regarding membrane protein types has been provided followed by a brief review of membrane protein structures.

2. Literature Survey

2.1. Membrane Protein Types

Conventional experimental methods usually offer accurate results; however, for some of the species and proteomics the microscopic detection are almost impossible to be conducted. This is mainly due to the inherent complications, and restraint of conventional methods to a limited number of proteins. Therefore, the demand for developing automatic, efficient, and reliable computational model is rising for identifying novel proteins. Consequently, these proteins are using for basic research and drug discovery. The classification procedure is accomplished in two steps. In the first step, the protein sequence is converted into fixed length feature vector. In the second step, the feature vector is provided as an input to the classification model to predict the desired result. However, the challenging task is to extract informative descriptors from amino acid sequences and to exploit the discrimination power of these features to predict the protein correctly. In this regard, a series of efforts have been performed to predict the types of membrane protein from their primary sequence of amino acids. Initial endeavors about the types of membrane protein were highlighted in 1999. Most of the developed methods
for the prediction of membrane protein types are based on two discrete models; AA composition [15; 16; 17; 18; 19; 20] and PseAA composition [21; 22; 23; 24; 25].

A pioneer work regarding the types of membrane protein has been introduced by Chou and Elrod [15]. They have utilized AA composition in conjunction with the covariant discriminant algorithm (CDA) where AA composition represents the occurring frequency of twenty amino acids [16; 17; 18; 19; 20]. The main pitfall in AA composition is the loss of sequence order information. In order to improve the quality of prediction model, the sequence order information is indispensable. Chou has proposed PseAA composition to overcome this shortcoming [26]. By employing PseAA composition, a protein sequence can be expressed without losing the sequence order and sequence length information. Chou has applied PseAA composition coupled with least hamming distance, least Euclidean distance, ProtLock, and CDA to predict membrane protein types and their locations, consequently, significant improvement has been reported [26]. Cai et al. have applied SVM [23], whereas, Wang et al. have introduced a variant of SVM, weighted-SVM and PseAA composition for membrane protein types prediction [24]. The problem arises when simple SVM operating uneven or imbalanced datasets, which make the predicted results of SVM undesirably biased towards the majority class. To some extent, this problem has been resolved through the weighted-SVM by assigning different weights to each class [27]. Furthermore, Liu et al. have utilized low frequency Fourier spectrum coupled with SVM for the same problem [28]. Shen and Chou have used optimized evidence-theoretic K-nearest neighbor (OET-KNN) and PseAA composition [29]. Further, Chou and Cai have proposed Gene Ontology (GO) in combination with PseAA composition [30]. However, the previous studies have conducted only for the types of membrane protein whereas the protein query already belonged to membrane proteins. In contrast, Chou and Cai have developed two phase predictor, which not only predict the types of membrane protein but also discriminating between membrane and non membrane proteins. Moreover, Chou and Cai have used the amphipathic PseAA composition and CDA [31]. In addition, Shen et al. have utilized the discrimination power of Fuzzy KNN to predict the types of membrane protein [32]. Moreover, AA composition and dipeptide composition have been used by Yang et al. [33], while Wang et al. have applied dipeptide composition and feature selection approach i.e. NPE [34].
Similarly, Rezaei et al. and Qiu et al. have introduced the notion of DWT to predict the types of membrane protein [35; 36]. The importance of wavelet transforms is the ability to extract both spectral and temporal information simultaneously while Fourier transform extracts spectral information only. In DWT, the protein sequence is transformed into Kyte and Doolittle hydrophobicity scale free of energies [37].

In addition, Jia et al. have introduced a hybrid feature by fusing protein domain profiles and their physicochemical properties [38], while Golmohammadi et al. have used a combination of AA composition and physicochemical properties [39]. In this model, a powerful feature selection technique mRMR is employed, which remarkably improves the discriminating capability of membrane protein types. Mahdavi and Jahandideh have endeavored to handle problems related to the complexity in membrane proteins [40].

In the last few years, the concept of ensemble classification has incurred reasonable consideration due to its superiority over individual classifiers. Individual classifiers may produce diverse classification results for the same problem. However, when the prediction of individual classifiers is combined through any combination schemes, the classification result may be maximized. In ensemble classification, the deficiency of one classifier is substituted by the advantages of other classifiers. Therefore, researchers have introduced various ensemble methods in the area of Computational Biology and Bioinformatics [41]. In view of this, ensemble classification is rapidly emerging on account of its superiority over the individual classifiers to enhance the discrimination capability of a classification system [42; 43]. The first ensemble approach for the prediction of membrane proteins has been proposed by Wang et al. [44]. They have developed stacked generalization approach in conjunction with PseAA composition. Similarly, PseAA composition and ensemble classifiers have also been adopted by several researchers to predict the types of membrane protein [42; 45; 46; 47]. However, Chou and Shen have used an ensemble of OET-KNN classifier in conjunction with PSSM to extract evolutionary information from protein sequences [9; 48]. Then PseAA composition has been applied to PSSM profile for feature extraction. The proposed predictor of Chou and Shen is called MemType-2L, which performs classification into two layers. Likewise, Pu et al. have used PSSM, whereas Nanni and Lumini have
employed AA composition and ensemble of SVM [45; 46] to predict the type of membrane proteins.

2.2. Membrane Protein Structures

The primary structure of a protein is a polymer of amino acids, where amino acids interact with each other to pattern a polypeptide chain. The secondary structure is configured by the formation of hydrogen bonds between amino acids in a polypeptide chain. According to the structural architectures, integral membrane proteins are classified into two distinct groups namely, alpha-helical membrane proteins and beta-barrel membrane proteins (outer membrane proteins). Both have their own significance and play a key role in structuring membrane proteins. In the last few decades, the prediction of membrane protein secondary structure remains an active research area and achieves some success in the field of Structural, Functional, and Computational Biology [14]. A series of efforts have been made in this regard, as mentioned below.

2.2.1. Transmembrane Proteins (Alpha-helices)

In order to comprehend the function of transmembrane proteins, it is essential to determine its spatial organization in the cell. Information about transmembrane proteins provides some useful clue in determining their structure. For instance, if some information about transmembrane proteins is known, then segments, penetrating within membrane and forming loops on either side of membrane can easily be determined. In early, a few visualization methods were developed for the prediction of transmembrane proteins such as “helical wheel” [49] and “helical net” [50]. In addition, a quantitative method “hydrophobic moment” [51] has also been used for the prediction of transmembrane proteins. In all these methods, the regions of alpha-helix are already identified and only a particular property of amino acids is detected.

Few researchers have applied biochemical and spectroscopic experiments for instance, infrared spectroscopy [52; 53], solid state Nuclear Magnetic Resonance (NMR) [54; 55; 56], and electron microscopy [57; 58], etc. The major problems in experimental approaches are the lack of raw materials for crystallization, toxicity, inclusion bodies, etc. Besides these, it is solely used for low-resolution structural information on membrane
proteins; in addition, it is time consuming and laborious. Due to difficulties in conventional experimental works, very limited number of membrane protein structures is available. In order to overcome this issue, topology is considered the alternative, where membrane protein structure is not available. Topology is based on two basic observations of alpha-helical transmembrane proteins. The first observation illustrates that where the transmembrane segments are embedded within membrane whereas the second determines which side of membrane connects the C-terminus with N-terminus by loop. This information is vital for both structural and functional classification of proteins [59]. The segments of transmembrane proteins are about 25 hydrophobic residues long, but sometimes vary between 15 to 40 residues [60]. On the other hand, the loops are hydrophobic and often contain glycines and pralines amino acids, which make turns [61]. Kyte and Doolittle have used sliding window of 19 residues long, and determined the transmembrane segments around 20 to 30 residues long based on hydrophobicity scale [37; 62; 63]. For improving the quality of predictors, hydrophobicity scales, window sizes, and amino acid propensities have been optimized [64; 65]. A ‘positive inside rule’ is defined, in which more positive charge amino acids, namely, Arg and Lys, are found in short cytoplasmic loops, which connect transmembrane segments to extracellular loops [66]. Besides hydrophobicity, other physicochemical properties of amino acids for instance, charges [67; 68; 69], non-polar phase helicity [70], and multiple sequence alignment [71; 72], DAS-TMfilter [73], TOP-Pred [69], and SOSUI [68] are utilized for prediction of topology and transmembrane segments. The focal problem with these methods is that they have recognized transmembrane segments with high precision, but has not achieved good results in topology prediction. Further, in order to increase the accuracy of membrane protein topology, computational methods have been used. Majority of computational methods are based on Machine learning algorithms such as Hidden Markov Models (HMM) [2; 74], neural networks [71], and SVM. PHDhtm has been introduced, using neural network in conjunction with evolutionary profiles for predicting transmembrane segments [71]. Later, it has been improved by computing the probability of each residue in transmembrane segments [75], which identify the local sequence patterns as transmembrane regions. On the other hand, TMHMM [76] and HMMTOP [74] have been developed using HMM, because it can detain the global
sequence patterns such as repeated patterns of transmembrane segments and loops. Looking at the significance of transmembrane segments, several studies have only emphasized on sensitivity instead of accuracy [77; 78; 79; 80; 81]. Some computational methods have predicted the position and number of transmembrane segments with high accuracy including SPLIT4 [67], TMAP [82], and HMMTOP2 [83], while MEMSAT2 [84] has identified the starting and ending positions of transmembrane segments with high precision. In contrast, TMHMM2 [2] has performed well in both. Recently, some prediction models are developed on consensus based. In consensus based models, the decision is made on an agreement among the participants. In addition, consensus models have merged and optimized the data of other prediction models in order to get the reliable transmembrane topology [69]. For instance, ConPred-II is developed by combining the predicted results of eight different models: DAS [84], MEMSAT [85], TMPred [86], TMHMM2 [2], TMAP [82], TopPred-II [69], KKD [85], and SOSUI [68]. Similarly, TOPCONS combines the success rates of five models [87] including PRODIV-TMHMM and PRO-TMHMM [88], OCTOPUS [89], SCAMPI-single and SCAMPI-multi [90]. The performance of the consensus models is boosted because of the agreement among multiple predictors. Some methods are developed using evolutionary information for the prediction of transmembrane topology such as MemBrain [91], poly-Phobius [92], and MEMSAT3 [93]. Evolutionary information has improved the accuracy of transmembrane topology. Though, evolutionary techniques involve multiple alignments, which are considered computationally expensive. Recently, Zaki et al. have used GA and compositional index for the prediction of transmembrane topology.

2.2.2. Outer Membrane Proteins

In the last few decades, huge amount of membrane protein sequences is identified. Most of them are alpha-helical membrane proteins. In contrast, owing to the vague patterns and limited availability of structure, the efforts regarding the prediction of outer membrane proteins (OMPs) are confined, which are discussed below.

Using conventional experimental techniques for discriminating OMPs, the ranges of beta-barrel strands are determined from 8 to 24 residues long [94]. The identification of beta-
barrel strands is slightly difficult, because it is less hydrophobic compared to alpha-helical membrane proteins. The difficulty is further intricate, because of the lack of a clear pattern in their membrane spanning strands [11]. Some physicochemical properties such as hydrophobicity, amphipathicity, and charge distribution have been used for the prediction of OMPs [95; 96; 97; 98; 99]. Paul and Rosenbusch have predicted transmembrane beta-strands by removing beta-turns and selecting a length of 6 residues for a strand [100]. Likewise, Vogel and Jahnig have used amphipathic properties of beta-strand for prediction of transmembrane strand [101] whereas Gromiha and Ponnuswamy have used the hydrophobicity property of amino acids for predicting the beta-strand [102].

On the other hand, several algorithms are used for the discrimination of beta-strand such as neural networks, SVMs, HMMs, and sequence alignment information [103; 104; 105]. Many efforts have been carried out for discriminating OMPs from non-OMPs. Mostly, prediction methods are based on topogenic sequence information. In addition, few web servers are also developed, which are freely available to research community [106; 107; 108]. According to the adopted algorithms, the development of the OMPs predictors has been considered in two different ways: simple statistical theory [109; 110] and Machine learning based methods. Statistical theory based developed OMPs predictors are simple and comprehensible these include DD [111], WED [110], WED_HFS [110], BOMP [108], and TMB-Hunt [12]. On the other hand, the importance of Machine learning based methods is that it can easily integrate the features with learning methods. Gnanasekaran et al. have used profiles of structure-based sequence alignment for discriminating beta-stranded integral membrane proteins [112]. Similarly, Wimley has developed hydrophobicity based model for identifying beta-barrel membrane proteins in genomic sequences [113]. Likewise, Zhai and Saier have developed a OMPs finder using various kinds of information such as secondary structure, amphipathicity, and hydropathy [114]. Bogas et al. have applied HMM in conjunction with conditional maximum likelihood criterion [11]. Park et al. have used discrete models AA and dipeptide composition in combination with SVM [115]. Gromiha and Suwa have also used AA composition as a feature extraction strategy as well as some statistical and Machine learning methods as classification algorithms for discriminating OMPs [14;
Wu et al. have introduced a new measure of information discrepancy for discriminating OMPs [116]. Yan et al. have utilized the Weighted Euclidean distance approach and KNN [110; 117]. Lin has adopted the quadratic discriminant analysis method [118] whereas Ou et al. have introduced a model implemented on the basis of radial basis function networks and PSSM profiles [109]. Furthermore, Gao et al. have introduced different forms of PseAA composition and SVM for the discrimination of OMPs [5]. Moreover, Mizianty and Kurgan have utilized the discrimination power of two feature extraction strategies namely, physicochemical properties and evolutionary information coupled with SVM [119], on contrary, Liang et al. have used sequence based approach and SVM for discrimination of OMPs [120].
Various processes and sub-processes of Machine learning, which are utilized in this research, are illustrated in Figure 3.1.

Figure 3.1. Overall framework of Machine learning processes and sub-processes.
3.1. Database

Before developing a statistical prediction method, one needs to construct or select an appropriate benchmark datasets according to the classification problem. In this study, we have used three benchmark datasets of membrane protein types including dataset1 [15], dataset2 [32], and dataset3 [9], one of non-membrane proteins, two of transmembrane proteins such as low-resolution [81] and high resolution datasets [88; 89], and two of outer membrane proteins including DS1 and DS2 [111; 115]. Each dataset contains protein sequences, which comprised of twenty amino acids polymer. Each sequence is represented by Fasta format (>) like this.

>41BB_HUMAN Q07011 homo sapiens
MGNSCYNIVATLLLVLNFERTSRLQDPCSNCPAGTFCDNRRNQICSCPNSFSSAGGQRTCDICRQCKGVFRTRKECSSTSNAECDCTPGFHCLGAGCSMCEQDCKQGQEILTKKGCKDCCFGTFNQDKRGICRPWTCNSLDGKSVLVNGTKERDVVCPSADLSPGASSVTPAPAREPGHSPQIQSF

3.2. Implemented Feature extraction Schemes

In pattern recognition and classification problems, an object is classified on the basis of specific patterns. These patterns or features are highlighted by various feature extraction strategies. In this section, we explain some of the feature extraction strategies, which are applied in this work.

3.2.1. Biochemical properties

This section explains the biochemical properties of amino acids, which are used as feature extraction schemes.

3.2.1.1. Amino Acid (AA) Composition

A protein sequence is a polymer of twenty amino acids. These amino acids are represented by a single letter code such as A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y. In AA composition, a relative frequency of each amino acid in protein sequence is calculated. Consequently, twenty discrete values are obtained where each value represents the calculated frequency of one of the twenty amino acids [16; 17; 18; 19; 20; 121]. AA composition is calculated as:
where \( i = 1, 2, 3 \ldots, 20 \), \( n \) is the occurrence frequency of amino acid \( i \), and \( N \) is the length of the protein sequence. Relative frequency of each amino acid is represented in Eq. 3.2.

\[
P = \left[ \begin{array}{c} f_1 \\ f_2 \\ \vdots \\ f_{20} \end{array} \right]
\]

where \( f_1 \) represents the normalize frequency of amino acid ‘A’, \( f_2 \) is the normalize frequency of amino acid ‘C’, and \( f_{20} \) is the normalize frequency of amino acid ‘Y’ and \( P \) is a protein sequence.

3.2.1.2. Pseudo Amino Acid (PseAA) Composition

In conventional AA composition, if two sequences belong to two different types of proteins having the same frequencies then classifier has no ability to differentiate between them. This issue may be resolved through the sequence order and sequence length information, but unfortunately, conventional AA composition does not preserve such information. To overcome the issue regarding conventional AA composition, Chou has introduced the concept of PseAA composition in 2001. PseAA composition represents not only conventional AA composition components but also some additional correlation factors. Let us consider a series of \( N \) amino acid residues

\[
R_{1}, R_{2}, R_{3}, R_{4}, R_{5}, R_{6}, R_{7}, \ldots, R_{N}
\]

The correlation factors can be defined as:

\[
\begin{aligned}
\Theta_1 &= \frac{1}{N-1} \sum_{j=1}^{N-1} \Theta(R_{j}, R_{j+1}) \\
\Theta_2 &= \frac{1}{N-2} \sum_{j=1}^{N-2} \Theta(R_{j}, R_{j+2}) \\
\Theta_3 &= \frac{1}{N-3} \sum_{j=1}^{N-3} \Theta(R_{j}, R_{j+3}) \\
&\quad \vdots \\
\Theta_4 &= \frac{1}{N-\hat{k}} \sum_{j=1}^{N-\hat{k}} \Theta(R_{j}, R_{j+\hat{k}})
\end{aligned}
\]
where $\Theta_1$ is the first-rank correlation factor represents the sequence order correlation among all the neighboring residues, $\Theta_2$ is the second-rank correlation factor represents the sequence order correlation among all the second neighboring residues in a protein sequence, and so on. Thus, the correlation function can be defined as:

$$\Theta(R_i,R_j)=\frac{1}{2}\left\{\left[H_1(R_i)-H_1(R_j)\right]^2 + \left[H_2(R_i)-H_2(R_j)\right]^2\right\}$$

(3.5)

where $H_1(R_i)$ and $H_2(R_i)$ represent the hydrophobicity and hydrophilicity value of amino acids $R_i$. $H_1(i)$ and $H_2(i)$ are the corresponding values for the $R_i$.

$$H_1(i) = \frac{H_1(i) - \sum_{j=1}^{20} H_1^w(i)}{\sqrt{\sum_{j=1}^{20} \left[H_1^w(i) - \sum_{j=1}^{20} H_1^w(i)\right]^2}}$$

(3.6)

$$H_2(i) = \frac{H_2(i) - \sum_{j=1}^{20} H_2^w(i)}{\sqrt{\sum_{j=1}^{20} \left[H_2^w(i) - \sum_{j=1}^{20} H_2^w(i)\right]^2}}$$

$$p_u = \begin{cases} \frac{f_u}{\sum_{i=1}^{20} f_i + w \sum_{j=1}^{\lambda} \Theta_j} & (1 \leq u \leq 20) \\ \frac{w \Theta_{u-20}}{\sum_{i=1}^{20} f_i + w \sum_{j=1}^{\lambda} \Theta_j} & (20+1 \leq u \leq 20+\lambda) \end{cases}$$

(3.7)

where $w=0.5$ is a weighted factor

$$PseAA = p_{AA} + p_{cor} = \begin{bmatrix} f_1 \\ \vdots \\ f_{20} \\ f_{20+1} \\ \vdots \\ f_{20+\lambda} \end{bmatrix}$$

(3.8)

where $f_1...f_{20}$ represent the occurrence frequency of twenty amino acids while $f_{20+1}...f_{20+\lambda}$ are the sequence order correlation factors. PseAA composition has been employed by many investigators for improving the quality of the prediction system [21; 22; 24; 32; 44; 122; 123; 124; 125; 126; 127].
3.2.1.3. Split Amino Acid Composition

There are many proteins in a cell, which have vital informative peptides at their N- or C-terminus regions. It is not an easy task to directly identify these informative peptides from protein sequences. In order to extract the complementary informative peptides, there is needed an approach, which can split the protein sequence into several fragments. For this purpose, SAAC is proposed. SAAC is a protein encoding scheme, in which a protein sequence is decomposed into several fragments and composition of each fragment is computed independently [121; 128; 129]. In our SAAC model, each protein sequence is decomposed into three fragments; (i) 25-AA of N-terminus, (ii) 25-AA of C-terminus, and (iii) region between these two termini. We have also resolved the problem related to small length sequences by taking 10 amino acids on N- and C-terminus. The N-terminus is an amino-terminus refers to the start of polypeptide whereas C-terminus is the carboxyl-terminus; it is the end of the chain. The rationale behind adopting this approach is that the percentage composition of a whole sequence does not provide sufficient weight to the compositional bias.

3.2.1.4. Discrete Wavelet Transform

Wavelet analysis has the capability of localizing both spectral and temporal information in a signal simultaneously, hence widely used by the research community to explore various parts of a signal [35]. In Wavelet transform, a scaled and a shifted version of the wavelet function $\psi(t)$ is multiplied with the signal $f(t)$ and then summed. The transformed coefficients $T(a,b)$ of the signal $f(t)$ can be defined as:

$$T(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t) \psi\left(\frac{t-b}{a}\right) dt$$

(3.9)

where $a$ is a scale and $b$ is a shifted parameter. Both are real numbers $R(n)$, $a > 0$, $\psi\left(\frac{t-b}{a}\right)$ is the evaluating wavelet function, and $t$ is the length of a sequence. The transformed coefficients $T(a,b)$ are established for both specific wavelet periods and specific locations on the signal, $t=b$ [34]. The topogenic sequences can be decomposed by DWT coefficients at different dilations to remove the noise from the profiles, which provide local information about the sequences. Therefore, DWT reflects sequence order.
information effectively. In this study, a binary dilation of $2^n$ and a dyadic translation of $n2^{-n}$ can be obtained by setting $a_0 = 2$ and $b_0 = 1$. Therefore, 
\[ \psi_{m,n}(t) = 2^n \psi(2^m t - n) \]  
(3.10)
where $m = 1, 2 \ldots$ and $n = 0, 1, 2 \ldots$ the wavelet coefficients of the signal $f(t)$ can be calculated as:

\[ T(a,b) = \langle f(t), \psi_{a,b}(t) \rangle = 2^{-\frac{m}{2}} \int_{-\infty}^{\infty} f(t) \psi(2^{-m} t - n) \]  
(3.11)

The transform coefficient $T(a,b)$ can be decomposed into two components: approximation coefficient $A^j(n)$ contains low frequency and high scale components and detail coefficient $D^j(n)$ consists of high frequency and low scale components of the signal $f(t)$. Approximation and detail coefficients for a signal at level $j$ can be defined as:

\[ A^j(n) = \sum_{k} h_{j-2n} A^{j+1}(k) \]  
(3.12)

\[ D^j(n) = \sum_{k} h_{j-2n} D^{j+1}(k) \]  
(3.13)

In this study, we have dealt with topogenic sequences where variation in the amino acid composition is considered in terms of position, therefore, the time variable $t$ will be substituted by the position variable. First, the topogenic sequence is transformed into Kyte and Doolittle hydrophobicity scale [37].

Figure 3.2. DWT decomposition-tree at level four
Then the numeric sequence is decomposed up to level 4, consequently, the approximation coefficient is $cA4$ and the detail coefficients are $cD4$, $cD3$, $cD2$ and $cD1$ as shown in Figure 3.2. Different statistical attributes such as Shannon entropy, log entropy, energy entropy, variance, mean, max, and min, respectively are utilized to explore each component of a signal.

### 3.2.2. Evolutionary Information

In this section, the details about evolutionary profiles are presented, which is utilized as a feature extraction scheme.

#### 3.2.2.1. Position Specific Scoring Matrix

PSSM is evolutionary profiles and patterns based representative that exploits multiple alignments and information regarding protein families. PSSM has initially used for identifying distant information related proteins. PSI-BLAST [130; 131] is widely used for evaluating PSSM profiles to detect remote information about homologous proteins. In PSSM, each amino acid residue against 20 values, which determine the frequencies of substitutions detected at the specific position in a protein family. PSSM matrix consists of negative and positive scores; negative indicates that the specified amino acids are substituted less frequently in the alignment while the positive shows that the substitutions take place more frequently. Let us consider a topogenic sequence $P$ with $N$ residues long, PSSM can be obtained as:

$$P_{PSSM} = \begin{bmatrix}
X_{1 \rightarrow 1} & X_{1 \rightarrow 2} & \ldots & X_{1 \rightarrow j} & \ldots & X_{1 \rightarrow 20} \\
X_{2 \rightarrow 1} & X_{2 \rightarrow 2} & \ldots & X_{2 \rightarrow j} & \ldots & X_{2 \rightarrow 20} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
X_{i \rightarrow 1} & X_{i \rightarrow 2} & \ldots & X_{i \rightarrow j} & \ldots & X_{i \rightarrow 20} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
X_{N \rightarrow 1} & X_{N \rightarrow 2} & \ldots & X_{N \rightarrow j} & \ldots & X_{N \rightarrow 20}
\end{bmatrix}$$

(3.14)

where $X_{i \rightarrow j}$ indicates the $i^{th}$ position residue score in the protein sequence, which is substituted by amino acid type $j$ in the biological evolutionary process. The values of $j=1\ldots 20$ represent the alphabetical order of 20 native amino acids. The $P_{PSSM}$ is obtained by executing PSI-BLAST [132; 133], which explored the Swiss-Prot database in three
iterations with the cutoff \( E \)-value of 0.001 for multiple sequence alignment against the sequence of the protein query \( P \). Consequently, \( N \times 20 \) scoring matrix as illustrated in Eq.3.14 has produced. Furthermore, standard conversion method is applied to normalize the \( \text{P}_{\text{PSSM}} \) as follows

\[
X_{i \rightarrow j} = \frac{X_i^0 - \tilde{X}_i^0}{SD\left(\tilde{X}_i^0\right)} \quad (i = 1, 2, ..., N; j = 1, 2, ..., 20)
\]  

(3.15)

where \( X_{i \rightarrow j} \) shows the original scoring matrix generated by PSI-BLAST [14, 65]. \( \tilde{X}_i^0 \) represents the average of \( X_{i \rightarrow j}^0 \) and \( SD\left(\tilde{X}_i^0\right) \) the standard deviation over \( (j=1, 2, ..., 20) \). Consequently, the \( N \) length matrix has been generated against \( N \) length protein sequence. However, the dimension of \( \text{P}_{\text{PSSM}} \) is different for each protein sequence. Therefore, the development of a predictor with the ability of handling proteins of different lengths is complicated. To overcome this issue, a protein query \( P \) may be represented as

\[
\tilde{P}_\text{Evo} = \begin{bmatrix} \tilde{X}_1 & \tilde{X}_2 & ... & \tilde{X}_{20} \end{bmatrix}^T
\]  

(3.16)

\[
\tilde{X}_j = \frac{1}{N} \sum_{i=1}^{N} X_{i \rightarrow j} \quad (j = 1, 2, ..., 20)
\]  

(3.17)

However, employing \( P_{\text{Evo}} \) of Eq.3.16 for protein \( P \) representation, all the sequence order information could be lost in the biological evolutionary process. In order to retain sequence order information, we have utilized the concept of PseAA composition, which was introduced by Chou [26]. The protein \( P \) can be represented as:

\[
P_{\text{PseEvo}} = \begin{bmatrix} \hat{X}_1 & \hat{X}_2 & ... & \hat{X}_{20} & \hat{X}_1^\lambda & \hat{X}_2^\lambda & ... & \hat{X}_{20}^\lambda \end{bmatrix}^T
\]  

(3.18)

\[
X_j^\lambda = \frac{1}{N-\lambda} \sum_{i=1}^{N-\lambda} \left[ X_{i \rightarrow j} - X_{(i+\lambda) \rightarrow j} \right]^2 \quad (1, 2, ..., 20; \lambda < N)
\]  

(3.19)

where \( X_j \) is the first-rank correlation factor by combining the PSSM scores of the closest neighbors in the protein sequence for a particular amino acid type \( j \); \( X_j^\lambda \) is the second-rank calculating the frequency of the second closest neighboring PSSM scores; and so on.
The value of $\lambda$ must be less than the length of the shortest protein sequence in the benchmark dataset.

### 3.2.3. Physiochemical properties

Physicochemical properties represent the behaviors of amino acids. In order to extract information from topogenic sequences, various physicochemical properties of amino acids are used, which are mentioned below.

#### 3.2.3.1. Composition and Translation

Composition and translation feature descriptors are used to indicate the amino acid distribution patterns of a particular structural or physicochemical property of proteins [134; 135]. To compute these feature descriptors, seven distinct physicochemical properties of amino acids are utilized including hydrophobicity, normalized Van der Waals volume, polarity, polarization, charge, secondary structure, and solvent accessibility. For each physicochemical property, the twenty amino acids are categorized into three sub-groups [5; 136]. For instance, the sub-group of hydrophobicity is polar, neutral, and hydrophobic while the sub-group of solvent accessibility is buried, exposed, and intermediate. On the other hand, charge includes positive, neutral, and negative etc. as shown in Table 3.1. First, the amino acid sequence is converted into the sequence of physicochemical properties according to the corresponding value of the residues. Secondly, composition and translation feature descriptors are computed for each physicochemical property, which represents the composition of each of the three sub-groups in protein. Of the seven distinct physicochemical properties, the total descriptor values of composition are $3 \times 7 = 21$.

Translation expresses the percentage frequencies with which the feature changes its index along the entire sequence of the protein. For instance, a protein sequence is converted into the charge feature that includes positive, neutral, and negative residues. The translation descriptors are calculated as:

- Positive residue is followed by neutral residue or
- Positive residue is followed by negative residue or
- Neutral residue is followed by positive residue or
- Neutral residue is followed by negative residue or
- Negative residue is followed by positive residue or
- Negative residue is followed by the neutral residue

Consequently, translation contains of a total of \(3 \times 7 = 21\) descriptors.

Table 3.1 Division of amino acids according to their physicochemical properties

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Divisions</th>
<th>Hydrophobicity</th>
<th>Normalized Vander Waals volume</th>
<th>Polarity</th>
<th>Polarizability</th>
<th>Charge</th>
<th>Secondary Structure</th>
<th>Solvent accessibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume range</td>
<td>Volume range</td>
<td>2.95-94.0</td>
<td>4.03-8.08</td>
<td>8.0-9.2</td>
<td>10.4-13.0</td>
<td>0.128-120.186</td>
<td>GNPSD</td>
<td>MPSTHY</td>
</tr>
</tbody>
</table>

### 3.2.3.2. Composite Protein Sequence Representation

Various descriptors of Composite Protein Sequence Representation (CPSR) are listed below.

- **Sequence Length**
  
  \(L\) represents the length of a protein sequence.

- **2-gram Exchange Group Composition**
  
  In 2-gram exchange group composition, the amino acid sequence is replaced by its equivalent 6-letter exchange group representation [48; 137] shown in Table 3.2, which has been obtained from the PAM matrix [138]. As a result, thirty-six features are extracted. The exchange groups are considering the broader categories of amino acids, which reflect the effects of evolution. For instance, in the original sequence H, R, and K
amino acids are substituted by $e_1$, D, E, N, and Q are substituted by $e_2$ and so forth. After substituting the whole sequence of amino acids, the resulting sequence contains only six different characters instead of twenty amino acids. Finally, the occurrence frequency, of each possible 2-gram pair of the consecutive amino acids exchange group, is calculated [139; 140].

**Hydrophobic Group**

In Table 3.2, amino acids are divided in two groups, i.e. hydrophobic and hydrophilic [141]. These two features are computed by counting the hydrophobic (non-polar) and hydrophilic (polar) residues in the protein sequence.

**Electronic Group**

The electronic groups indicate whether a specified amino acid is the electron donor, electron acceptors, or neutral. These features are calculated by counting the occurrence frequency of amino acids in each of the electronic groups as shown in Table 3.2.

<table>
<thead>
<tr>
<th>Table 3.2 Deriving features based division of amino acids [140]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group</strong></td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>Exchange group</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Hydrophobic group</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Electron group</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>R-group</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
The sum of Hydrophobicity

Each amino acid has corresponded hydrophobic value that is often calculated using a hydrophobic index. In this study, we have used the Eisenberg hydrophobic index, listed in Table 3.3, which gives some information regarding the membrane associated helices [142]. The index is normalized and ranges between -2.53 for R (the least hydrophobic) and 1.38 for I (the most hydrophobic). Sum of hydrophobicity index is calculated by adding the hydrophobic index value of each amino acid in the protein sequence.

<table>
<thead>
<tr>
<th>Amino Acid</th>
<th>Index value</th>
<th>Amino Acid</th>
<th>Index value</th>
<th>Amino Acid</th>
<th>Index value</th>
<th>Amino Acid</th>
<th>Index value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.62</td>
<td>E</td>
<td>-0.74</td>
<td>L</td>
<td>1.06</td>
<td>S</td>
<td>-0.18</td>
</tr>
<tr>
<td>R</td>
<td>-2.53</td>
<td>Q</td>
<td>-0.85</td>
<td>K</td>
<td>-1.5</td>
<td>T</td>
<td>-0.05</td>
</tr>
<tr>
<td>N</td>
<td>-0.78</td>
<td>G</td>
<td>0.48</td>
<td>M</td>
<td>0.64</td>
<td>W</td>
<td>0.81</td>
</tr>
<tr>
<td>D</td>
<td>-0.9</td>
<td>H</td>
<td>-0.4</td>
<td>F</td>
<td>1.19</td>
<td>Y</td>
<td>0.26</td>
</tr>
<tr>
<td>C</td>
<td>0.29</td>
<td>I</td>
<td>1.38</td>
<td>P</td>
<td>0.12</td>
<td>V</td>
<td>1.08</td>
</tr>
</tbody>
</table>

R-Group

The side chain of each amino acid is different. However, there are some characteristics, which are similar on these side-chains. On the basis of these similarity amino acids are clustered into five groups [143], which are shown in Table 3.2. The composition of each of these groups is computed.

3.3. Feature Selection Techniques

The critical issue in pattern recognition and classification is the feature selection. Instead of using the whole features space to check the discriminative system, a small optimal subset is utilized. Therefore, feature selection or dimensionality reduction is considering a vital step before investigation of data. The major aim of the dimensionality reduction is to preserve most of the relevant information, remove redundant and unnecessary features, reduce data noise, and computational cost. On the other hand, it improves classification accuracy and model generalization. Few feature selection techniques are mentioned below:
3.3.1. Principal Component Analysis

PCA is an unsupervised linear dimensionality reduction approach, which performs covariance analysis between factors. PCA is eigenvector-based multivariate analysis that preserve high variation in the data [144]. It transforms correlated variables into uncorrelated variables called principal components [145]. It is commonly used for identifying patterns in high dimensional data and maps high dimension space in low dimensional space with minimum loss of relevant information. Let suppose, we have M×N data matrix $A$ where $M$ is the number of attributes, $N$ is the number of instances, and $k$ is the desired dimension of the feature space. The value of $k$ must be less than $N$. First, the data is centered on means for each attribute.

- Calculates mean: $\bar{x}_j = \frac{1}{M} \sum_{i=1}^{M} x_{ij}$
- Subtract the mean $\bar{x}$ from $x_i$: $\phi_i = x_i - \bar{x}$
- Calculates the covariance matrix: $C = (x_i - \bar{x})(x_i - \bar{x})^T = AA^T$

where $A = \{\phi_1, \phi_2, ..., \phi_N\}$ (N×M)
- Computes eigenvalues of $C$: $\lambda_1 > \lambda_2 > ... > \lambda_N$
- Computes eigenvectors of $C$: $v_1, v_2, ..., v_N$
- Finally, choose $k$ components, which have the highest eigenvalues.

The first component is the highest eigenvalue; the second component is the second highest eigenvalue, and so on. The importance of PCA is, it can effectively reduce the dimensionality of a feature space with minimum correlation as well as with minimum loss of significant information. PCA uses global Euclidean structure and is more sensitive to outliers.

3.3.2. Neighborhood Preserving Embedding

NPE is a linear approximation dimensionality reduction technique, which can retain the local neighborhood structure of the data in the feature space. It is used to extract the indispensable features from high dimensionality data [146]. NPE establishes nearest neighbor search in a low dimensional space because local structure contains more information than global structure [147]. For example, given a set of data points, NPE first
creates a weight matrix, which shows the neighborhood relationship between the data points. Especially, it describes linear combination neighborhood and combination coefficients of the data points. After that, tries to find an optimal embedding where the neighborhood structure can be preserved in the low dimensional space. Finally, computes the linear projection.

Suppose, \( X \) is a dataset of \( N \) dimensional \( X = \{x_1, x_2, \ldots, x_M\} \in \mathbb{R}^N \). The aim of NPE is to seek a transformation matrix \( A \) to transform \( X \) into a new dimensional space \( Y = \{y_1, y_2, \ldots, y_N\} \in \mathbb{R}^d \) where \( d < N \) and \( Y = A^T X \).

Algorithm:
- Constructing an adjacency graph
  
  Let \( G \) denotes a graph with \( M \) nodes and the \( i^{th} \) node is corresponded to the \( x_i \) data point. The graph is constructed using nearest neighbors. Nodes are directly connected by an edge if \( x_j \) is among \( k \) nearest neighbors of \( x_i \).
  
  - Computing the weights

  Let \( W \) be a weight matrix contains the weight of edge from node \( i \) to node \( j \), and 0 in case of no such edge. The weight of the edges can be computed in order to minimize the following objective function.

  \[
  \min \sum_i \left| x_i - \sum_j W_{ij} x_j \right|^2
  \]
  (3.20)

  where \( \sum_j W_{ij} = 1 \) more details are mentioned in [105].

  - Computing the projections

  In this step, linear projection is computed to solve the generalized eigenvector problem.

  \[
  X S X^T a = \lambda X X^T a
  \]
  (3.21)

  where
  
  \( X = \{x_1, x_2, \ldots, x_M\} \)
  
  \( S = (I - W)^T (I - W) \)
  
  \( I = \text{diag} (1, \cdots, 1) \)

  So, it can be easily checked that \( S \) is symmetric and semi positive definite.
Let the column vectors $a_1, a_2, ..., a_d$ be the solutions of equation (I), ordered according to their eigenvalues, $\lambda_1 < \lambda_2 < ... < \lambda_d$. Thus, the embedding is as follows:

$$x_i \rightarrow y_i = A^T x_i$$

$$A = (a_1, a_2, a_3, ..., a_d)$$

where $y_i$ is a d-dimensional vector and $A$ is an $N \times d$ matrix.

### 3.3.3. Minimum Redundancy Maximum Relevance

mRMR is a feature selection technique that selects a feature subspace, which can better characterize the statistical property of target classification variables. The aim of mRMR is to find out a feature subspace that has minimum redundancy with other features and maximum relevance to the target class. The correlation and relevance can be estimated by calculating the mutual information between the features themselves and between the features and the class variables [148]. The mutual information (MI) between the features can be defined as:

$$MI(x, y) = \sum_{i,j \in S} p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(x_i) p(y_j)}$$

(3.22)

Where $x$ and $y$ are two features, $p(x_i, y_j)$ is a joint probabilistic density function and $p(x_i) p(y_j)$ is a marginal probabilistic density function. Similarly, the mutual information between class target and features can be calculated as:

$$MI(x, z) = \sum_{i,k \in S} p(x_i, z_k) \log \frac{p(x_i, z_k)}{p(x_i) p(z_k)}$$

(3.23)

where $x$ is a feature and $z$ is a class target. Minimum redundancy in a whole features can be computed as:

$$\min(mR) = \frac{1}{|S|^2} \sum_{x,y \in S} MI(x, y)$$

(3.24)

where $S$ describes the feature space and $|S|$ represents total number of features in $S$.

The maximum relevance condition can be satisfied by increasing relevance between features and target classification variable.

$$\max(MR) = \frac{1}{|S|} \sum_{x,z \in S} MI(x, z)$$

(3.25)
Finally, combines the minimum redundancy and maximum relevance by obtaining feature subspace, the last two conditions should be optimized simultaneously.

\[ \max(\nabla MI) = MR - mR \]

In case of continuous features in feature space, the features must be converted into discretized feature space [149]. In order to convert continuous features into discretized form \((\text{Mean} \pm \text{standard deviation})/2\) and three states are used. If the feature value is greater than \((\text{mean} + \text{standard deviation})/2\), it will be converted into state 1. If the feature value between the \((\text{mean} - \text{standard deviation})/2\) and \((\text{mean} + \text{standard deviation})/2\) then it will be transformed into state 2, otherwise state 3.

### 3.3.4. Singular Value Decomposition

SVD is a dimensionality reduction technique that plays a vital role in many multivariate data analysis. It tries to find out such a reduced dimensional matrix, which has strong correlation with no noise effect. In addition, it reconstructs the best possible matrix, which has least possible information and emphasis strong patterns and trends. There are three mutual compatible points for considering SVD [150; 151]. The first one is transforming correlated variables into uncorrelated variables, which expose the relationship between the original data. The second one is identifying and ordering the dimensions along with exhibition of the most variation in data points. Once, highest variation is identified then it is probably easy to discover the best approximation of the original space in the form of fewer dimensions.

Feature representation vector usually contains redundant, irrelevant, and mutually dependent information. In order to eradicate the irrelevant and mutually dependent information, SVD transform the feature vector into an orthogonal dimension space. Suppose, we have a matrix \(A\) of size \(M \times N\). SVD splits the matrix \(A\) of \(M \times N\) into three matrices \(U\), \(W\), and \(V\) like that \(A=UWV^T\) when \(M>N\).

where \(U\) is a \(M\times M\) orthogonal matrix that represents the left singular vectors of \(A\), \(V\) is a \(N\times N\) orthogonal matrix that indicates the right singular vectors of \(A\), and \(W\) is a \(M\times N\) diagonal matrix having the singular values of \(A\) in decreasing order \((\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(M,N)})\) along its diagonal. The rank shows nonzero singular values of matrix \(A\).
The required matrix is obtained by multiplying the first $k$ columns of the $U$ matrix by the first $k$ singular values from the $W$ matrix and the first $k$ rows of the $V^T$ matrix as shown in Figure 3.3. The eigenvalues represent the energy (variance) in the corresponding dimensions.

### 3.4. Data Partition (Validation)

The performance of the classification algorithms is measured in term of error rate. In order to estimate the classification error, the given benchmark dataset is partitioned. These partitions are made through the most prevalent using statistical techniques like holdout and cross validation. In holdout, certain amount of data is used for testing and the remaining for training. Usually, one-third for testing and two-third for training are reserved. Since, the main problem with holdout method is that all the instances of dataset have no representation. It might be possible that certain classes have no representation in the training set and could expect that a classifier might be learned more from the missing data. Therefore, it is important that the partition sets must have a right proportion of all the class representatives. This process is called stratification.

Cross-validation is another data partitioning technique. It is widely used for estimating the performance of classifiers. In cross-validation, the dataset is divided in a fixed number of mutually exclusive folds. Mostly, four types of cross-validation (self-consistency, jackknife, independent dataset, and subsampling tests) are used to assess the performance of learning algorithms.
3.4.1. Self-consistency Test

Self-consistency test is conceived the basic and simple test to measure the quality of the learning algorithms. While performing self-consistency test, the process of training and testing is accomplished using the same dataset. However, if the performance of learning algorithm is worse on self-consistency then it cannot conceive as a good one. Therefore, the self-consistency test is definitely crucial but not sufficient for estimating the performance of the learning algorithms. It is also called resubstitution test. It often gives an extremely optimistic estimate of the true generalization error.

3.4.2. Independent Dataset Test

In independent dataset test, the training and testing process are performed using different datasets. However, to select the independent proteins for investigating the predictor could be quite arbitrary unless the number of independent proteins is sufficiently large [152].

3.4.3. Jackknife Test

In jackknife test, the dataset is partitioned into \(n\)-fold. It is also called leave-one-out cross validation test. In jackknife test, one instance is left out for testing and the model is trained by the remaining dataset instances. The whole process is repeated \(n\) times. Finally, the predicted results of all \(n\) judgments are averaged, which indicated the estimated error. In cross validation tests, jackknife test is conceived the most rigorous, effective and objective one, because it always yields unique results. Jackknife test is attractive for the main two reasons over other tests. The first one is, in each iteration, the utmost possible amount of data can be utilized for training, which presumably enhances the generalization power of the classifiers. The second quality of jackknife test is the deterministic and additionally, no arbitrariness is entailed. Hence, owing to these eminences, jackknife test is applied by most of the researchers to investigate the power of the learning models. The main drawback of jackknife test is computational cost, because it is executed \(n\) times.
3.4.4. Sub-sampling Test

In sub-sampling test, the original dataset is partitioned into $k$-fold cross validation. One fold is held for testing and the remaining $k-1$ folds for training. This process is repeated $k$ times where each fold is used exactly once for testing. Ten-fold cross validation is commonly used. Since, the main problem in sub-sampling test is the very small fraction of the possible selections. On the other hand, different selections will always predict different results even for the same benchmark dataset and same predictor. Accordingly, the sub-sampling test cannot evade the arbitrariness and is unable to yield a unique result.

3.5. Classification Algorithms

There are various machine learning algorithms, which are applied in the areas of data mining and Bioinformatics. However, the basic premises of all the machine learning algorithms are the same by using a set of known examples to obtain information about unknown example. The known examples are usually called training set and unknown examples are called testing set. Machine learning algorithms can be classified into supervised and unsupervised learning. Supervised learning usually consists of concerning a series of attributes of the data to a specific class or numerical value known as a label of that specific example. In contrast, in unsupervised learning, there are no predefined classes or labels.

Classification is the sub-discipline of data mining where data is assigned to the predefined groups. It is usually known as supervised learning because the labels of these groups or classes are known in advance. In a classification process, classes are determined on the basis of data attribute values and characteristics of already known data for which these classes are defined. Various classification algorithms, used in this research, are discussed below:

3.5.1. AdaBoost

AdaBoost is one of the most popular and widely used ensemble classification methods in Machine learning. It has successfully performed the concept of adoptive boosting (AdaBoost). AdaBoost combines a set of weak learners to form a high prediction model
in order to enhance the accuracy. Due to its significance, several researchers have utilized the concept of AdaBoost in various fields such as face recognition, video sequences, classification, and signal processing systems.

The concept of AdaBoost was first introduced by Freund and Schapire in 1996 [153], to handle the problems regarding with earlier boosting algorithms. AdaBoost generates a set of hypothesis. These hypotheses are generated by training a simple and moderately accurate classifier (weak learner) repeatedly in specified intervals. In the first iteration, each training instance is initialized with equal weight. Further, weight is updated in each training phase to focus on the hard instances in the training data. In the training phase, the instances, misclassified by the previous learner, are more likely to be included in the training of the next learner. The final hypothesis is obtained by combining the individual hypotheses through majority voting.

Let us consider the input dataset

\[ S = \{ (x_1, y_1), (x_2, y_2), ..., (x_n, y_n) \}, \quad y \in \{-1, +1\} \]

where \( x_i \) is a vector of attribute values is a subset of \( X \) and \( y_i \) is a label of target classes is a subset of label set \( Y \).

Input: \( S = \{ (x_i, y_i) \} \) \( (i=1, 2, ..., n) \)

\( K \): number of iterations

- Initialize the weights \( x_i \): \( w_i(i) = \frac{1}{n} \)
- For \( k=1:K \)
  - Compute the normalize weights \( x_i \): \( p_i(i) = \frac{w_i(i)}{\sum_i w_i(i)} \)
  - Train weak learner \( h_k := \text{learn}(S, p_i) \)
  - Calculate the error of \( h_k \): \( \varepsilon_i = \sum_i p_i(i) [ h_k(x_i) \neq y_i ] \)
  - If \( \varepsilon f \frac{1}{2} \) then
    - Go to next iteration
  - \( \beta_i = \frac{\varepsilon_i}{1 - \varepsilon_i} \)
• Compute new weights $w_{k+1} = w_k(i) \beta_{k}^{[h_k(y_i)]}$

• End for loop

- Output: $H(x) = \arg \max_{y \in \mathcal{Y}} \sum_{k=1}^{K} \left( \log \frac{1}{\beta_k} \right) [h_k(x) = y]$ 

$H(x)$ is the final hypothesis obtained from the $K$ weak hypotheses $\{h_1, h_2, \ldots, h_K\}$ through majority voting.

### 3.5.2. Random Forest

RF is an ensemble classification technique developed by Breiman in 2001[154]. It can be used for regression, classification, clustering, and feature selection. It is the collection of an individual tree hypothesis where each tree grows with respect to different bootstrap sample with the same distribution [155]. The notion of RF is introduced by adding an additional layer of randomness to bootstrap aggregation. In RF, the instances are drawn randomly from original training dataset with replacement using bootstrap technique to build multiple trees. However, each tree is produced with a randomized subset of predictors, hence the name “random” forests. A large number of trees (500 to 2000) are produced, hence a “forest” of trees. The number of predictors used to find the optimal split at each node is a randomly chosen subset of the total number of predictors. Each tree casts a single vote for the most popular class at input $x$. Finally, the performance of the classification decision is calculated by ensemble the individual hypotheses using majority voting. RF can help in reducing the bias of the individual weak predictors and does not require variable rescaling. In addition, it is more robust against overfitting, noise, and considering a user friendly, because it requires only two parameters: number of variables in random subset at each node and number of trees in the forest.

The performance of the standard RF is suffered in case of extremely imbalance dataset. The RF classifier tends to be biased towards the majority class, where the minority class has a penalty of misclassification. To handle this problem and makes the RF more suitable for learning from the extremely imbalance data, weighted random forest is proposed [156]. In weighted RF, weight is assigned to each class where larger weight is assigned to minority class. The class weights are utilized in two places of the RF algorithm; in tree induction procedure, to weight the Gini criterion for seeking splits and
in the terminal nodes of each tree. The class prediction of each terminal node is determined by a weighted majority vote. RF determines the final class prediction by aggregating the weighted vote from each individual tree, where the weights are the average weights in the terminal nodes.

### 3.5.3. Support Vector Machine

SVM is a supervised learner, and one of the most effective machine learning algorithms based on statistical learning theory. It is extensively used for classification, pattern recognition, and regression related problems. In order to better classify, SVM transforms the data into high dimensional space to maximize and clear the gaps between the two categories of instances. It was first introduced by Cortes and Vapnik in 1995 [125; 157; 158; 159; 160; 161] and later updated by the Vapnik in 1998 [162]. SVM was initially introduced for binary classification. Later, it is adopted for multiclass problem in several ways such as one-against-one, one-against-all, and directed acyclic graph SVM. In case of two class problem, SVM draws a parallel line to the hyperplane that determines the distance between dividing line and the closest points in the training set; the points are called support vectors and the distance is called margin depicted in Figure 3.4. Subsequently, SVM tries to find out an optimal decision surface that has the maximum distance from support vector to minimize the error for a novel instance. Suppose, we have $N$ training pair $(x_i, y_i)$ where $x_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$ the decision surface is calculated as:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i x_i^T x + \text{bias}, \quad \alpha_i > 0$$  \hspace{1cm} (3.26)

where $\alpha$ is the Lagrange multiplier. The linear SVM uses the dot product of two points in input space as a kernel function. In case of non-separable patterns, the hyperplane is calculated as:

$$\varphi(W, \zeta) = 1/2 W^T W + C \sum_{i=1}^{N} \zeta_i$$  \hspace{1cm} (3.27)

Subject to the condition

$$y_i (W^T \varphi(x_i) + b) \geq 1 - \zeta_i, \quad \zeta_i > 0$$
where $C > 0$ is the penalty term of $\sum_{i=1}^{N} \xi_i \cdot \varphi(x)$ is the nonlinear mapping, and weight vector $w$ minimizes the cost function term $W^T W$. SVM maps data from the low dimension $N$ to high dimension $M$ through $\varphi(x)$ such that $\varphi: R \rightarrow F^M$, $M > N$, for nonlinear data.

After transformation, the point $\varphi(x)$ is subject to Mercer’s theorem. Non-linear decision surface $f(x)$ is defined as:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x_i, x) + \text{bias}$$

(3.28)

where $N$ represents support vectors, $K(x_i, x)$ is the kernel function and are defined as:

$$K(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$$

(3.29)

SVM has several types of kernel function including linear, polynomial, Gaussian (RBF), and sigmoid with parameters $\gamma$ and $C$.

The linear kernel is represented as:

$$K(X, Y) = X \cdot Y$$

(3.30)

where $X$ is the feature vector and $Y$ is the target label vector. The linear kernel is faster than nonlinear kernel. In linear kernel based function, the objects do not map into a high-dimensional feature space, because $\phi(X) = X$.
The polynomial kernel is represented as:
\[
K(X,Y) = (X^d + 1)^{g_d}
\]
(3.31)
where \(d\) is the degree of polynomial kernel. As the degree of polynomial kernel increases, the shape of the decision boundary becomes more complex in the input space. Polynomial kernel maps the data into an infinite dimensional space. When \(d=1\), the polynomial and linear kernels are treated similarly.

RBF is defined as:
\[
K(X,Y) = \exp\left(-\gamma \|X - Y\|^2\right)
\]
(3.32)
where the parameter ‘\(\gamma\)’ shows the width of Gaussian function. The cost parameter ‘\(C\)’ controls the tradeoff between margin and classification error.

### 3.5.4. Probabilistic Neural Network

The computational power and flexibility of the attributes of PNN are similar to back-propagation neural network. On the other hand, it has the simplicity and transparency of conventional statistical classification approaches. The concept of PNN was introduced by Specht in 1990 [163], which based on the Bayes theory. It estimates the probability of a instance being part of a learned category [164]. The most remarkable and practicable advantage of the PNN is that, it operates completely in parallel, so there is no need of feedback from the individual neurons to the inputs, unlike many other neural networks. PNN has four layers: input, pattern, summation, and output layers are presented in Figure 3.5. The input node has \(N\) nodes, each corresponding to one independent variable. These input nodes are then fully connected to the \(M\) nodes of the pattern layer. One particular node in pattern layer corresponds to one training object only. An input vector \(P_i\) is processed by pattern node \(j\) using an activation function, whose most popular form is the exponential one:
\[
u_j = \exp\left(-\gamma \frac{\|P - P_j\|^2}{\sigma^2}\right)
\]
(3.33)
where \(v_{ij}\) denotes the output of the pattern node \(j\) and \(\delta\) is a smoothing factor that controls the width of the activation function.
If the $\|P_j - P_i\|$ distance between the input vector $P_i$ and the vector $P_j$ of the pattern node $j$ increases, similarity between the two data vectors decreases and vice versa. The outputs of the pattern layer are provided to the summation layer, which contains $v$ competitive nodes each corresponding to one class. Now each summation node $v$ is connected to the pattern nodes that are associated to training objects of class $v$.

![Diagram of PNN](image)

**Figure 3.5. Framework of PNN**

For an input vector $P_i$, the summation node $k$ receives the outputs of the associated pattern nodes for producing an output:

$$f_s(P_i) = \frac{1}{N_v} \sum_{j=1}^{N_v} u_{ij}$$

(3.34)

where $Q_v$ is the label of the class corresponding to the summation node $v$, while $N_v$ is the number of training objects belonging to this class. If we assume that all data vectors are normalized then, the previous equation can be formulated as:

$$f_s(P_i) = \frac{1}{N_v} \sum_{j=1}^{N_v} \exp \left( \frac{(P_i - P_j)^2}{\sigma^2} \right)$$

(3.35)

The outputs of the summation layer can be expressed in terms of the posterior class membership probabilities:

$$P(Q_v = v|P_i) = \frac{f_s(P_i)}{\sum_{v'} f_s(P_i)}$$

(3.36)
Using this above equation, a classification rule is incorporated at the output layer for assigning the input vector to $P_i$ to a particular class. The straightforward approach is to select the class whose $P(v|P_i)$ is maximum.

### 3.5.5. K-Nearest Neighbor

KNN is the most popular classifier in the area of pattern recognition, regression, and classification owing to its simplicity, adaptability, good performance, and easy to comprehend. Regardless of its simplicity, it can generate competitive and incredible performance compared to many other learning algorithms. KNN is a non-parametric classification algorithm and has no prior information about the distribution of the data [165]. It has no explicit training phase, while keeping all the training data in testing phase. The unlabeled instances are classified through nearest neighbors in the feature space. It is also called instance base learner or lazy learner. The KNN learner based on the notion of distance, which calculates the distance between the protein query and the training instances [161; 166]. Subsequently, the specified number of $K$ instances from the feature space is selected, which has been closing distance from protein query. Finally, assigns the most frequently occurring class to the protein query. When the number of $K=1$ it is called nearest neighbor classifier, otherwise, it refers to KNN classifier, which makes the decision on majority voting. In case of a tie, the decision is made by assigning randomly one of the associated classes with the tie to protein query. However, this situation happens very rarely, because the number of $K$ is mostly odd. The prediction performance of KNN classifier enhances and reduces the effect of noise in the classification when the number of nearest neighbors increases. On the other hand, the computational cost rises and also makes the boundaries less distinct between classes. Mathematically, it can be expressed as:

Suppose, we have a training dataset of $N$ protein sequences $P = \{p_1, p_2, \ldots p_N\}$ labeled $\{y_1, y_2, \ldots y_N\}$. Now, we have a protein query $Z$ that needs to be identified. The first step is to calculate the distance between $Z$ and $P_i$. Here, we have chosen Euclidean distance. It can be formulated as:

$$S(Z, P_i) = 1 - \left( \frac{Z \cdot P_i}{\|Z\| \|P_i\|} \right) \quad \{i = 1, 2 \ldots N\} \quad (3.37)$$
\[ \|Z\| P \] is the dot product of the two vectors and \[ \|Z\| P \] are the modulii. The minimum Euclidean distance is calculated as:

\[ S(Z, P_i) = \text{Min}(S(Z, p_1), S(Z, p_2), \ldots, S(Z, p_n)) \]  (3.38)

The next step is to select the number of closest neighbors and finally, assigns the majority occurring class \( P_k \) to the protein query \( Z \).

### 3.5.6. Fuzzy K-nearest Neighbor

Simple KNN classifier has some limitations and completely fails in particular circumstances. For instance, it is unable to determine the type of protein query that does not belong to the training data. The second problem is the behavior of KNN is similar for all \( K \) neighbors assigns equal weights, it does not consider the distance between protein query and its neighbors. In order to overcome these problems, Keller has introduced a new classification method by incorporating the Fuzzy set theory with KNN classifier called Fuzzy KNN [167]. The searching procedure of both KNN and Fuzzy KNN is similar, however, in KNN; every data point has only one class, which is the majority class in nearest neighbors [31; 168; 169]. In contrast, in Fuzzy KNN, data point can be more than one class and several membership functions are associated to these classes. In Fuzzy KNN method, the fuzzy class membership functions \( u_i(p) \) is calculated for protein query \( p \) according to the following relation.

\[
u_i(p) = \left( \frac{\sum_{j=1}^{K} u_i(p_j)^m D_{ij}^{2/m-1}}{\sum_{j=1}^{K} D_{ij}^{2/m-1}} \right)^{1/2} \quad i=1, \ldots, C \]  (3.39)

where \( m \) is a fuzzy strength parameter that can be identified through cross validation technique. It controls the distance magnitude of the neighboring data points from the test data point [170], \( k \) represents number of nearest neighbors, and \( C \) denotes number of classes. \( D = \|p - p_j\| \) computes the Euclidean distance between the query point \( p \) and \( p_j \). There are several methods used to calculate the distance such as Euclidean, Hamming, Mahalanobis distance etc. However, in Euclidean distance, the distance between any two instances is not affected due to the inclusion of new instances for investigation, which may be outliers. \( u_j(p_j) \) indicates the membership value of \( p_j \) for class \( C_i \) would be 1 if \( x_j \)}
belongs to the \( j^{th} \) class, otherwise 0. After calculating all the membership values, the highest membership value class will be assigned to query point. The limitations of simple KNN have been overcome through Fuzzy KNN by considering the distance between the protein query and its neighbors. It is assumed that if the two instances are closed to each other, they will be considered as the same types. In addition, a new class label can be determined if the maximum membership value is less than predefined threshold \( \theta \). The identified class will be assigned to protein query and included in the training data for future reference. Due to this characteristic, fuzzy KNN is deemed the type of similarity-based classification methods.

3.5.7. Evidence theoretic K- Nearest Neighbor Classifier

Due to its simplicity, easy to use, and apparent efficiency, still KNN remains a desirable classifier of many researchers and endeavoring to improve its performance. Denoeux has released a new version of KNN, ET-KNN in 1995 [171]. In ET-KNN, the notion of Dempster Shafer theory is incorporated with KNN classifier [171; 172; 173]. In this classification method, each neighbor of a protein query is considered as a part of evidence to support certain hypotheses about the class membership of that protein query [174]. Basic belief assignment masses (BBA) are computed for each subset of all classes based on this sort of evidence. The BBA masses are acquired for each of the K-nearest neighbors and aggregated using Dempster’s rule of combination [175]. Finally, the decision is made by assigning maximum credibility class to the protein query. Let \( C = \{C_1, C_2, \ldots, C_M\} \) denotes \( M \) classes. Consider \( X \) as the classified protein query and \( \phi_k \) is a set of its K-nearest neighbor of the training dataset. The label of each member of \( \phi_k \) \((\forall X^i \in \phi_k, L_i = q)\) can be considered as a part of evidence and \( X \) is belonged to \( C_q \). This part of evidence can be referred to BBA \( m^i \) assigning a fraction \( \alpha_q^i \) of the unit mass to the singleton \( \{C_q\} \) and the rest of the \( C \).

\[
m^i(\{C_q\}) = \alpha_q^i \quad \text{(3.40)}
\]

\[
m^i(C) = 1 - \alpha_q^i \quad \text{(3.41)}
\]
where $m'(A) = 0$ for all $A \subseteq C$ and $A \notin \{C, \{C_i\}\}$ [171]. The mass $\alpha_q'$ is a decreasing function of the Euclidean distance $d_i$ between $X$ and $X_i$.

$$\alpha_q' = \alpha_o \exp\left(-\gamma_q^2 (d_i')^2\right)$$

(3.42)

where $\gamma_q$ is a parameter associated to class $C_q$ and $\alpha_o$ is a constant parameter. The BBA, $m^1, m^2, \ldots, m^k$ corresponding to the $K$-nearest neighbors of the $X$ can then be merged using Dempster rule.

$$m(A) = \sum_{A \cap A_q = A} m'(A)m'(A_q)/\sum_{A \cap A_q \neq A} m'(A)m'(A_q)$$

(3.43)

where $m = m^1 \oplus m^2 \oplus \ldots \oplus m^k$. Consequently, credibility and plausibility of each class can be achieved $C_q (q=1, 2 \ldots M)$

Bel $(C_q) = m(C_q)$

Pl $(C_q) = m(C_q) + m(C)$

where Bel is a credibility and Pl is plausibility.

Finally, a protein query $X$ assigns that class, which has the maximum credibility, or maximum plausibility $C_q max$.

$$X = \arg\max_q m(C_q)$$

(3.44)

### 3.5.8. Generalized Regression Neural Network

GRNN architecture is similar to PNN but the main difference between in both, PNN carries out classification where the target value is categorical and GRNN performs regression where the target value is continuous. It is the variation of PNN developed by Specht in 1991 for regression problems [176]. It is a parzens non-parametric kernel regression estimator based on Parzen-Rosenblatt density estimation. The importance of GRNN against other neural networks is that it can truly calculate the approximation function from light data and select the proper regression model from the data automatically. It is robust to noise and outliers owing to its property of being instance based technique that work with weighted averages of the stored model [177; 178]. It calculates the conditional mean estimate for each class. Let $X = \{x_1, x_2, x_n\}$ is $n$ dimensional feature vector of a function $y = f(x)$. Its corresponding output is real values.

The target of the approximation function is to acquire an estimate $\hat{f}$ of $f$ using $X$. 

68
\[
\hat{f}(x) = E[y | x] = \int_{-\infty}^{\infty} yP(y, x)dy / \int_{-\infty}^{\infty} P(y, x)dy
\]

(3.45)

\(P(x, y)\) is joint probability density function and \(E[y | x]\) is the conditional expectation.

For instance, the density \(P(x)\) from a set of \(n\) training instances is given as:

\[
P(x) = \frac{1}{n\sigma^d} \sum_{i=1}^{n} W(x-x_i/\delta)
\]

(3.46)

where \(W\) is the kernel function with the required condition that \(\delta w(\xi) \geq 0\) and \(\int w(\xi)d\xi = 1\).

If the parameter \(\delta > 0\) then the kernel controls the smoothness of the approximation. The main drawback of GRNN is the high degree of smoothness and suffers from a curse of dimensionality.

### 3.5.9. Genetic Programming

GP is an evolutionary learning technique, solves problems automatically and offers an immense potential for pattern recognition and classification. GP performs similar functions to GA like evolution and natural selection in order to develop solutions to problems. However, GA uses fixed length binary strings while GP using variable length tree structure [179]. Tree structures are encoded both genotype as genetic information and phenotype like an executable computer program [180; 181]. After that, the tree structure representation is transformed in a computer program as LISP expression [182; 183]. GP programs are comprised of a set of primitive components terminals and functions. Functions are the internal nodes of trees while terminals are the external nodes of trees, which take no arguments.

The procedure of GP can be illustrated as following.

1. An initial population of individuals is randomly generated from the existing primitives.
2. Executes and calculates fitness function for each individual in the population.
3. Select individuals from the population to participate in genetic operations using fitness values.
4. Create a new generation using genetic operators (Crossover, mutation, and reproduction).
5. Repeat Step 2 to Step 4 until some termination criterion is met.
6. Return the best so far individual.
➢ **Representation**

The following text presents a discussion on the representation of GP.

- **Initialization of Population**

The initial population in GP is generated randomly through various approaches.

  - **Full**

  In this method, the initial population is generated where the depth of all the tree branches will be the same. The depth of a node is the number of edges that need to be traversed to reach the node starting from the root node. The root node is presumed to be at depth 0. The depth of a tree is a depth of its deepest leaf. Nodes are picked at randomly from the function set until maximum tree depth is reached. At the depth limit, nodes are selected from terminal sets.

  - **Grow**

  This method allows generating trees of varied sizes and shapes. It is similar as Full method, but it selects nodes from completely a primitive set of functions and terminals, until the depth limit is reached. When it reaches to the specified depth then only selects the terminals.

  - **Ramped half-and-half**

  In Ramped half-and-half method, the population is generated using a mixture of Full and Grow methods. Half of the population is generated by adopting Full method while remaining half by using Grow method. This method is considered the best one for generating the initial population, because the trees are diverse structure and size.

- **Selection**

In Selection, the individuals are selected from a population for later breeding. These individuals are selected based on fitness value and then applying some genetic operators namely, mutation, crossover, and reproduction. The most common methods used for selection are mentioned below.
- **Roulette-wheel selection**: In this method, a roulette wheel is divided into several sectors. One sector is occupied by each individual in the population, but the size of each sector depends on fitness value of the individual in the sector. The highest fitness individuals take larger sectors while small fitness individuals take smaller sectors. The roulette-wheel is rotated many times to select individuals for crossover, mutation, and reproduction operations.

- **Tournament selection**: In tournament selection, several individuals are selected randomly from population. Then the individuals in the tournament compete with each other to select for the next generation. Usually, a few best individuals in the tournament are selected based on tournament size. Each individual has a chance to compete in several tournaments, but only those individuals win the tournament, which has the highest fitness value.

- **Genetic operator**

  Crossover, mutation, and reproduction are widely used genetic operators in GP to create a next generation.

  - **Crossover**: crossover operator usually is the most prevalent genetic operator. In crossover, the genetic materials are combined from the two parents and producing two offspring trees. A crossover point is selected randomly from each parent tree then swapping the sub-tree below the crossover point on each of the parent trees in order to generate offspring trees.

  - **Mutation** is the second most commonly used operator in GP, which is adopting for randomly change in existing individual tree. In mutation, a single function or terminal is randomly selected from the parent tree, which makes up the point of mutation. Then the mutation point is replaced with a new randomly generated sub-tree.

  - **Reproduction** is the third genetic operator in GP. In reproduction, only a copy of individual is made to go into the next generation. There is no change in a program.
• **Terminal and Function Set**
  
  - The terminal set is usually variables, functions having no arguments, and constants.
  - On the other hand, function set is arithmetic operators (+, -, *, /) mathematical Operators (SIN, COS, EXP), Looping (FOR, REPEAT), Boolean Operators (AND, OR, NOT), Conditional Operators (IF – THEN – ELSE), etc
• **Fitness Function**

Genetic Programming explores all search space in order to make all possible solutions by comprising the primitive set in all possible ways. Now the question arises that how to decide that which solution is the best. This thing is achieved by assigning the fitness value to each individual in the population.

There are several ways to measure the fitness depending on the problem domain. Some examples are; the magnitude of error between desired output and the predicted output; the amount of time (fuel, money etc.) required stabilizing a system; the accuracy of a program in recognition pattern in or classifying objects; the pay-off that a game-playing problem produces; the compliance of a structure with the design criterion specified by the user.

• **Parameters of Genetic Programming**

GP has different control parameters that have to be set before running a problem such as population size, the probability of genetic operators, the maximum depth of the tree, and the selection mechanism.

• **Termination**

The termination criterion includes the specified number of generations is completed, or the desired fitness is achieved.

**3.6. Evaluation criteria**

After developing a model, there needs some criteria in order to assess the performance of a model that how much it successfully implemented the problem. Several parameters are used to measure or estimate the performance of machine learning algorithms. However, which parameter to use is still the major issue, because it depends on types of data and classification. All the performance measures are computed through confusion matrix. In confusion matrix the classification results are compared with actual results. It is represented by a matrix where each row determines the predicted class, while each column represents actual class.
Table 3.4. Confusion Matrix

<table>
<thead>
<tr>
<th></th>
<th>Predicted:</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive</td>
<td>Negative</td>
<td></td>
</tr>
<tr>
<td><strong>Real:</strong></td>
<td>Positive</td>
<td>True Positives</td>
<td>False Negatives</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>False Positives</td>
<td>True Negatives</td>
</tr>
</tbody>
</table>

Table 3.4 shows the confusion matrix where True positive (TP) is positive instances, which are predicted positive, true negative (TN) is negative instances predicted as negative, false positive (FP) is negative instances predicted positive, it is also called type-I error, and false negative (FN) is positive instances predicted as negative, it is also called type-II error. There are few extensively used standard performance parameters, which are mentioned below:

### 3.6.1. Accuracy

Almost, the performance of a classifier is evaluated through error rate or accuracy, because it is the simplest performance measure amongst the machine learning community. Accuracy determines the degree of true prediction of a model either true positive or true negative. It is the proportion of true predictions. It can be calculated as:

\[
Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100
\]

(3.47)

Even though, accuracy to be intuitively appropriate for classifier performance comparison but facing many challenges in real life problems.

### 3.6.2. Sensitivity and Specificity

Sensitivity indicates the proportion of true positive while specificity measures the proportion of true negative. Sensitivity shows the ratio between the predicted true positive instances and total number of true positive instances whereas specificity indicates the ratio between the predicted true negative instances and total number of true negative instances.

\[
Sensitivity = \frac{TP}{TP + FN} \times 100
\]

(3.48)

\[
Specificity = \frac{TN}{TN + FP} \times 100
\]

(3.49)
For example, in medical diagnosis tests, sensitivity represents the percentage of correctly identified diseases while specificity shows the percentage of correctly recognized normal condition. Normally, the decision in medical diagnosis is made in two states, normal or diseased. Both sensitivity and specificity have equal importance, because a test can be specific without being sensitivity and percipient without being specificity. Accuracy is dependent on sensitivity and specificity, because the accuracy is derived from both these. If both sensitivity and specificity are high then accuracy will be high. If sensitivity is high and specificity is low then accuracy will bias towards sensitivity. In case of opposite, it will bias towards specificity. If both are low then accuracy will be low. In medical point of view, both are needed should be high but sometimes have tradeoff in term of sensitivity and specificity. In addition, accuracy is also suffering due to the majority class in imbalance dataset. If the majority class is highly predicted then the accuracy will be high otherwise low.

3.6.3. Mathew’s Correlation Coefficient

MCC is considered one of the most rigorous and favorable performance measures in machine learning. The range of its values is \([-1 \ 1]\), whereby 1 means perfect prediction, 0 mean average prediction, and -1 mean inverse prediction.

\[
MCC(i) = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\]  

(3.50)

MCC is the powerful evaluating measure especially in case of imbalance dataset. Therefore, MCC has the ability to resolve the challenges of accuracy. For instance, if the number of positive instances is greater than the number of negative instances, then the classifier can easily predict all the instances as positive, because it bias towards majority class. Thus, the performance of the classifier will be worse, because it predicts all the negative instances incorrectly. In this case, the accuracy is 100 of positive instances and MCC is zero. Therefore, the MCC is considered as the best performance parameter for the classification of imbalance data.
3.6.4. F-measure

F-measure is a statistical measure based on harmonic mean often utilized for assessing the quality of a classifier. It is the combination of precision and recall, where precision represents the percentage of true predictions among all returned predictions and recall indicates the percentage of true predictions among total number of observed instances. The best value of F-measure is one, while the worst is 0.

\[ F - \text{measure} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \]  
(3.51)

\[ \text{Precision} = \frac{TP}{TP + FP} \]  
(3.52)

\[ \text{Recall} = \frac{TP}{TP + FN} \]  
(3.53)

The F-measure can be easily generalized for multiclass classification [184]. Where recall and precision for label \( k \) are formulated as:

\[ \text{Recall}_k = \sum_{\{i|E \in D \land k \in Y_i\}} \frac{|Y_i \cap Z_i|}{|Y_i|} \]  
(3.54)

\[ \text{Precision}_k = \sum_{\{i|E \in D \land k \in Z_i\}} \frac{|Y_i \cap Z_i|}{|Z_i|} \]  
(3.55)

3.6.5. Q-statistics

In order to measure diversity in ensemble classification the average value of \( Q \)-statistic [185] is used. The \( Q \)-statistic value of any two classifiers \( C_i \) and \( C_j \) can be calculated as:

\[ Q_{ij} = \frac{N^{11} \times N^{00} - N^{10} \times N^{01}}{N^{11} \times N^{00} + N^{10} \times N^{01}} \]  
(3.56)

| Table 3.5. Relation between classifier \( C_i \) and classifier \( C_j \) |
|---------------------------------|-----------------|-----------------|
| \( C_i \) correct (1)          | \( C_j \) correct (1) | \( C_j \) wrong (0) |
| \( C_i \) correct (1)          | \( N^{11} \)     | \( N^{10} \)     |
| \( C_i \) wrong (0)            | \( N^{01} \)     | \( N^{00} \)     |

Table 3.5 depicts the relation between two classifiers where \( N^{11} \) represents the number of correct predictions and \( N^{00} \) incorrect predictions of both classifiers. However, \( N^{10} \) is the correct predictions of classifier first and incorrect predictions of classifier second and \( N^{01} \) is the correct predictions of classifier second and incorrect of first. The value of \( Q \) is
between -1 and 1. Where -1 indicates inverse, zero means no diversity in that case the classifiers are statistically independent, and 1 mean high diversity. The diversity in ensemble classifier is computed by taking the average value of Q-statistic among all pairs of the $L$ base classifiers, which can be defined as:

$$Q_{avg} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} Q_{i,k}$$  \hspace{1cm} (3.57)

### 3.6.6. Receiver Operating Characteristic Curve

ROC curve is a handy tool, which is extensively used for summarizing the intrinsic properties of classification methods. It demonstrates the accuracy of a predictor comprehensively and visualize in an attractive way. It was initially used in the Second World War for signal detection of radar to display the tradeoff between the rate of hit and false alarm of friend and enemy airplanes [186; 187]. ROC curve plots true positive rate (sensitivity) versus false positive rate with varied thresholds. True positive rate is plotted on the vertical axis and false positive rate on the horizontal axis. Area under the ROC curve represents the performance of a classifier. If the area under the ROC curve is closer to the left top corner, the model will be considered the most accurate. ROC curve is also used to visualize the behavior of the diagnostic systems.

![Figure 3.8. Shows the number of healthy and diseased patients](image)
Figure 3.8 illustrates the number of healthy and diseased patients arranged according to diagnostic test. The criterion value determines the number of true positives, false positives, true negatives, and false negatives. If the objective is to limited number of false positives then a test will be required high sensitivity and good specificity.

![Figure 3.8](image)

Figure 3.8. Depicts the ROC curve

On the other hand, if every positive test result demonstrates a diseased tissue with high accuracy, then the diagnostic test should have a high specificity and good sensitivity. Sensitivity and specificity of a diagnostic test depend on the quality of a test. In addition, there is needed to use different cut-off points for different clinical situations to minimize the error of the test results. Figure 3.9 depicts the tradeoff between sensitivity and specificity, because when the sensitivity is increased the specificity is decreased. Since, the cutoff-points show whether the diagnostic result is better. The ideal location is the upper left corner (0, 1) where sensitivity and specificity are both 100%, it is also called to perfect classification.
4. Prediction of Membrane Protein Types Using Composite Protein Sequence Representation

4.1. Introduction

In previous chapters, literature survey and various processes of Machine learning are discussed. This chapter commences the discussion of the first phase of this research carried out for the prediction of membrane protein types. Membrane protein is a vital part of a cell and involves many cellular processes. It remains the primary target in biological and pharmacological research, because about 50% drugs are directly targeted against membrane proteins. Each type of membrane protein can reflect different biological functions of the membrane. Therefore, the prediction of each membrane protein type with high confidence is necessary. Many efforts have been carried out in this regard and reasonable results have consequently been achieved, however, some room for improvement still exists. In this chapter, we have introduced CPSR as a feature extraction strategy, which is the composition of seven distinct feature spaces. Further, PCA is employed to reduce the dimensionality of a feature space. SVM, PNN, and GRNN, are used as base learners. The aim of this work is to develop an accurate and high throughput performance predictor for membrane protein types by exploring the discrimination power of CPSR and the learning capability of the classifiers. The performance of the classifiers is evaluated by performing three statistical tests including self-consistency, jackknife, and independent dataset tests on a benchmark dataset. Accuracy, sensitivity, specificity, MCC, and F-measure are utilized to analyze the performance of the proposed method.
4.2. Materials and Methods

In this section, first the dataset and its constituent classes are presented. Next, the feature extraction scheme is explained. Then, a feature selection technique utilized in this work is presented.

4.2.1. Dataset

In order to develop a prediction model, the first step is to choose appropriate benchmark datasets. Therefore, we have used a standard dataset for performance analysis and evaluation, which was originally constructed by Chou and Elrod in 1999 [15]. To reduce homology bias and redundancy, it is essential to preprocess the dataset. Therefore, the dataset has been passed from several processes. Initially, those sequences are excluded from the dataset whose descriptions are ambiguous. Next, only one protein sequence has been selected from those who has the same name, but from different species. Finally, all multiclass sequences have been excluded due to lack of uniqueness. After mentioned screening procedures, the resultant dataset contains only 2,059 protein sequences in training dataset and 2,625 in testing dataset. Both of the datasets have been comprised of five types of membrane protein. The training dataset contains 435 type-I transmembrane proteins, 152 type-II transmembrane proteins, 1,311 multipass transmembrane proteins, 51 lipid chain-anchored membrane proteins, and 110 GPI-anchored membrane protein sequences. On the other hand, testing dataset consists of 487 type-I transmembrane proteins, 180 type-II transmembrane proteins, 1,867 multipass transmembrane proteins, 14 lipid chain-anchored membrane proteins, and 86 GPI-anchored membrane protein sequences.

4.2.2. Composite Protein Sequence Representation

In this study, we have proposed CPSR as a feature extraction strategy. CPSR is comprised of seven distinct feature sets including AA composition, sequence length, 2-gram exchange group, hydrophobic group, electronic group, sum of hydrophobicity, and R-group. Each feature set has its own corresponding number of features; the details are provided in Chapter 3. The dimensionality of the CPSR is 71-D where 20-D of AA
composition, 1-D of sequence length, 36-D of 2-gram exchange group, 2-D of hydrophobic group, 6-D of electronic group, 1-D of the sum of hydrophobicity, and 5-D of R-group.

4.2.3. Principal Component Analysis

Mostly, feature space contains some irrelevant and redundant information, which disgraces the performance of classifier. In addition, such feature spaces consumes more memory and computational cost. A technique is needed that not only eliminates redundancy but also selects such features, which can efficiently perform discrimination among the types of membrane protein. In this study, we have employed PCA as feature selection technique, because it reduces the dimensionality of a feature space without losing much information. PCA is based on eigenvector multivariate analysis, which retains high variation in the data. The original dimensionality of the feature space is 71-D that is reduced to 48-D after applying PCA. The variance between the first principal component and the second principal component is shown in Figure 4.1, where almost 99% of the variance is reported for the first two principal components.

Figure 4.1. First principal component versus second principal component
4.3. Experimental Results and Discussion

In this work, we have examined the performance of the classifiers by three statistical cross validation tests: self-consistency, jackknife, and independent dataset tests. Each test has its own significance, but jackknife test is deemed the most excellent and effective one, because it always generates unique results. It gives a chance to every instance of benchmark dataset. Therefore, it does not require any stratification, in addition, maximum possible amount of data uses for training the model. The performance of the classifiers is tested through several measures such as accuracy, sensitivity, specificity, MCC, and F-measure, because sometimes, single measure is not sufficient to evaluate the performance of a classifier.

4.3.1. Self-consistency Test

Self-consistency test is a basic test, which is only used to test the behavior of datasets. If the performance of the classifier on self-consistency test is prominent then it is further evaluated using other statistical tests. The predicted result of our proposed approach using self-consistency test is reported in Table 4.5. All of the three classifiers have yielded similar results, which are 99.9%. The result through self-consistency test is not adequate for estimating the performance of the classifiers. However, we have used this test merely for comparison with the results reported by other authors, because several investigators have adopted this for testing the performance of their models.

4.3.2. Jackknife Test

The success rates of the classifiers using jackknife test are shown in Table 4.1. Among the three classifiers, SVM has achieved the highest accuracy of 86.01%. Its sensitivity is 85.1%, specificity 85.8%, MCC 0.63, and F-measure 0.71 that is higher compared to PNN and GRNN. The second highest accuracy is obtained by PNN, which is 82.51%. Its sensitivity is 82.46%, specificity is 82.09%, MCC is 0.56, and F-measure is 0.65. GRNN has yielded comparatively low accuracy of 78.14%. The performance of GRNN is low reported, because the generalization capability of GRNN is low compared to SVM and PNN.
Table 4.1. Performance of classifiers using Jackknife test.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>86.01</td>
<td>85.15</td>
<td>85.82</td>
<td>0.63</td>
<td>0.71</td>
</tr>
<tr>
<td>PNN</td>
<td>82.51</td>
<td>82.46</td>
<td>82.09</td>
<td>0.56</td>
<td>0.65</td>
</tr>
<tr>
<td>GRNN</td>
<td>78.14</td>
<td>73.54</td>
<td>78.77</td>
<td>0.45</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Table 4.2. Performance of classifiers using Independent dataset test.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>95.23</td>
<td>93.28</td>
<td>95.51</td>
<td>0.85</td>
<td>0.88</td>
</tr>
<tr>
<td>PNN</td>
<td>95.73</td>
<td>96.22</td>
<td>95.41</td>
<td>0.87</td>
<td>0.89</td>
</tr>
<tr>
<td>GRNN</td>
<td>93.71</td>
<td>92.24</td>
<td>93.74</td>
<td>0.81</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Figure 4.2. Performance of classifiers using Jackknife test.
The accuracy, sensitivity, and specificity are shown in Figure 4.2 whereas MCC and F-measure are depicted in Figure 4.3. Both the Figures reveal that the performance of the SVM using jackknife test is better compared to PNN and GRNN in all the measured parameters.

4.3.3. Independent Dataset Test

The predicted outcomes of the classifiers are reported in Table 4.2. It can be observed that PNN outperforms other utilized classifiers with 95.73% accuracy. Likewise, sensitivity (96.22%), specificity (95.41%), MCC (0.87), and F-measure (0.89) are also promising. SVM has obtained 95.23% accuracy, 93.28% sensitivity, 95.51% specificity, 0.85 MCC, and 0.88 F-measure. The accuracy of SVM is slightly higher than that of PNN, which is about 0.50%; however, the MCC of PNN is 2% higher than that of SVM. Since, MCC is considered as the best performance measure compared to accuracy, therefore, in this particular case PNN is reflected on better than SVM. In contrast, the performance of GRNN is worse compared to SVM and PNN.
4.3.4. Individual Membrane Protein Types

In case of jackknife test, the accuracy of SVM, PNN and GRNN for each membrane protein type is listed in Table 4.3. The accuracies of the classifiers, for each membrane protein type, are in dispersed nature. One classifier has yielded good result for one type while other classifier has yielded better result for the other. However, overall, the performance of SVM is good for each type of membrane proteins. It has achieved the highest accuracy of 81.61 and 95.34% for type-I and multipass transmembrane proteins, respectively. In contrast, PNN and GRNN have also achieved good results for them, because these two types have large sample representations in the dataset. Multipass and type-I transmembrane proteins have occupied 84.7% of examples of the whole dataset, which exhibits high prediction performance compared to the rest of types. In contrast, the performance of SVM is affected owing to the type II transmembrane, lipid-chain anchored, and GPI-anchored membrane proteins, since the data contains only 15% of examples of these three types. The accuracies of PNN for type-II transmembrane and GPI-anchored membrane proteins are higher than that of SVM and relatively good for other types. The performance of GRNN is poor for type-I transmembrane and GPI-anchored membrane proteins while better for type-II transmembrane proteins than that of other classifiers.

Table 4.3. Performance of classifiers for each membrane protein type using Jackknife test

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Type-I</th>
<th>Type-II</th>
<th>multipass</th>
<th>anchored</th>
<th>GPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>81.61</td>
<td>50.00</td>
<td>95.34</td>
<td>56.86</td>
<td>55.45</td>
</tr>
<tr>
<td>PNN</td>
<td>76.09</td>
<td>51.97</td>
<td>91.30</td>
<td>49.02</td>
<td>60.91</td>
</tr>
<tr>
<td>GRNN</td>
<td>56.78</td>
<td>58.55</td>
<td>90.92</td>
<td>49.02</td>
<td>50.91</td>
</tr>
</tbody>
</table>

Table 4.4. Performance of classifiers for each membrane protein type using Independent dataset test.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Type-I</th>
<th>Type-II</th>
<th>multipass</th>
<th>anchored</th>
<th>GPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>89.33</td>
<td>82.22</td>
<td>99.09</td>
<td>85.71</td>
<td>74.41</td>
</tr>
<tr>
<td>PNN</td>
<td>94.14</td>
<td>90.00</td>
<td>97.05</td>
<td>85.71</td>
<td>89.53</td>
</tr>
<tr>
<td>GRNN</td>
<td>85.77</td>
<td>89.44</td>
<td>96.83</td>
<td>71.42</td>
<td>82.55</td>
</tr>
</tbody>
</table>
Figure 4.4 illustrates the performance comparison among SVM, PNN, and GRNN for each membrane protein type. Each of the classifier has achieved the highest accuracy for multipass transmembrane protein.

On the other hand, the performance of SVM, PNN, and GRNN using an independent dataset test is reported in Table 4.4. The predicted results revealed that the success rate of PNN for each membrane protein type is better than that of SVM and GRNN. The accuracy of SVM is very high for multipass transmembrane protein while comparatively low for other types than that of PNN. The prediction of each type of membrane proteins with high accuracy is necessary, because each type has its own importance in pharmaceutical industries for various drugs discovery. Figure 4.5 represents the performance of the three classifiers using the independent dataset test for membrane protein types.
4.3.5. Comparison with the Existing Approaches

In Table 4.5, comparison of the proposed method is drawn with existing approaches on the same dataset. The first automated model for the prediction of membrane protein types was developed by Chou and Elrod [15]. The performance of their predictor is, respectively, 81.1, 76.4, and 79.4% using self-consistency, jackknife, and independent dataset tests. Afterwards, Chou has introduced CDA coupled with PseAA composition for improving the performance of their predictor and obtained accuracy of 90.9% through self-consistency test, 80.9% through jackknife test, and 87.5% through independent dataset test [26]. Other methods have also been proposed by various researchers [23; 24; 28], among which, the one introduced by Wang et al. is considered the most successful. They have achieved the highest results 98.7, 85.4, and 94.3%, using self-consistency, jackknife, and independent dataset tests, respectively, by applying PseAA composition and stacking generalization [44]. Recently, Rezaei et al., Qiu et al., and Wang et al. have proposed other methods but have not achieved good results compared to the one previously introduced by Wang et al. [34; 35; 36; 44].
Table 4.5. Comparative analysis between the proposed approach and existing approaches

<table>
<thead>
<tr>
<th>Methods</th>
<th>Self-consistency test (%)</th>
<th>Jackknife test (%)</th>
<th>Independent dataset test (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDA [15]</td>
<td>81.1</td>
<td>76.4</td>
<td>79.4</td>
</tr>
<tr>
<td>CDA &amp; PseAA [26]</td>
<td>90.9</td>
<td>80.9</td>
<td>87.5</td>
</tr>
<tr>
<td>AA Composition &amp; SVM [23]</td>
<td>96.2</td>
<td>80.4</td>
<td>85.4</td>
</tr>
<tr>
<td>Low Frequency Fourier Spectrum [28]</td>
<td>99</td>
<td>78.0</td>
<td>87</td>
</tr>
<tr>
<td>Weighted u-SVM using PseAA [24]</td>
<td>99.8</td>
<td>82.3</td>
<td>90.3</td>
</tr>
<tr>
<td>PseAA and Stacking [44]</td>
<td>98.7</td>
<td>85.4</td>
<td>94.3</td>
</tr>
<tr>
<td>Wavelet &amp; Cascade Neural Network [35]</td>
<td>96.8</td>
<td>81.3</td>
<td>91.4</td>
</tr>
<tr>
<td>Discrete wavelet and SVM [36]</td>
<td>80.0</td>
<td>78.1</td>
<td>….</td>
</tr>
<tr>
<td>Dipeptide Composition, NPE &amp; KNN [34]</td>
<td>---</td>
<td>82.0</td>
<td>90.1</td>
</tr>
<tr>
<td>Proposed CPSR &amp; GRNN</td>
<td>99.9</td>
<td>78.1</td>
<td>93.7</td>
</tr>
<tr>
<td>Proposed CPSR &amp; PNN</td>
<td>99.9</td>
<td>82.5</td>
<td>95.7</td>
</tr>
<tr>
<td>Proposed CPSR &amp; SVM</td>
<td>99.9</td>
<td>86.0</td>
<td>95.2</td>
</tr>
</tbody>
</table>

Our proposed approach has outperformed all the previously published methods and yielded the highest results for membrane protein types. The importance of the proposed approach is that it utilizes the biochemical and physicochemical of seven distinct feature spaces, while other approaches have used only one feature space. In addition, high computational cost problem is also resolved by decreasing the curse of dimensionality of the feature space. In the proposed method, SVM shows the best accuracy of 99.9 and 86.01% using self-consistency, and jackknife tests, respectively. In contrast, PNN achieves the highest success rates of 99.9 and 95.73% in case of self-consistency and independent dataset tests, respectively. The performance of our proposed approach is 1.2, 0.62, and 1.4% higher than the previous highest success rates of the Wang et al. method [44] in case of self-consistency, jackknife, and independent dataset tests. Similar improvements have been reported in other measures. After ascertaining, we have observed that this successful and remarkable enhancement is possible because of the good discriminative capability of the CPSR and the learning power of the SVM and PNN.
5. Prediction of Membrane Protein Types Using Ensemble Classification

5.1. Introduction

In the previous chapter, some individual classification algorithms have been discussed regarding the prediction of membrane protein types. In this chapter, we introduced the concept of ensemble classification.

In the last few years, ensemble classification has attracted great attention of researchers, and proved its superiority over individual classifiers. Ensemble architecture can show better performance than the individual classifiers. In individual classification, each classifier uses different resolution in each feature dimension where chance of the error rate is different. In contrast, if the hypothesis of these individual classifiers is combined, there might be a possibility to reduce the error rate.

Upon providing accurate and diverse hypothesis, this combination of classifiers will give more accurate results. However, an accurate classifier might show better result than a random guess classifier, while diverse classifiers might produce different predictions on new sample, which has different error rates. Empirically, ensemble classifier always generates better result, whenever significant diversity exists in the nature of models. Therefore, accuracy and diversity are considered the most important parameters in ensemble classification. Besides, main issue in ensemble classification is the selection of individual classifiers. In order to select appropriate classification algorithms, it is necessary to figure out the strengths and weaknesses of each individual classifier, and find a tradeoff among the classification algorithms. In this work, we have chosen five different classification algorithms of distinct natures including RF, AdaBoost, PNN, KNN, and SVM. The protein sequences are expressed by three feature extraction schemes namely, DWT, PseAA composition, and SAAC. The hybrid versions of these
feature extraction schemes are also constructed. NPE is employed as a feature selection technique, to check the feature space for redundant and extraneous information. We have adopted two approaches of ensemble classification in order to develop ensemble architecture: simple majority voting and GA based majority voting. Two latest datasets and three cross validation tests are performed to assess the performances of the individual as well as the ensemble classifiers. Accuracy, sensitivity, specificity, MCC, F-measure, and Q-statistics are employed as performance measures.

5.2. Materials and Methods

In this section, first the dataset and its constituent classes are presented. Next, the feature extraction schemes and ensemble classification are explained.

5.2.1. Datasets

In this chapter, we have utilized two latest benchmark datasets: dataset2 and dataset3, which were originally developed by Chou & Cai in 2005, and Chou & Shen, in 2007 [9; 32]. Both the datasets were constructed from the SWISS-PROT data bank. Dataset2 consists of six types of membrane protein, whereas dataset3 contains eight types of membrane protein. The initial datasets contain some redundancy and homology bias. Therefore, in order to reduce redundancy and homology bias, first dataset2 is passed through the same procedure like dataset1 as discussed in Chapter 4. Next, sequence identity is performed for each type of membrane proteins. For instance, the length of one membrane protein sequence is \( N_1 \) residues, while the length of other membrane protein sequence is \( N_2 \) residues \( (N_1 > N_2) \). \( M \) indicates the matching residues of one sequence along the other. The sequence identity between the two protein sequences is calculated as \( (M/N)*100\% \). The average sequence identity for type-I transmembrane proteins is 7.97\%, type-II transmembrane proteins 7.94\%, multipass transmembrane proteins 8.31\%, lipid chain-anchored membrane proteins 7.94\%, GPI-anchored membrane proteins 7.92\%, and peripheral membrane proteins is 11.36\%. These percentages reveal that, the most of pairs in these types have very low sequence identity. After the above screening procedures, the training dataset consists of 2,628 protein sequences, of which 372 are type-I transmembrane proteins, 151 type-II transmembrane proteins, 1,903 multipass
transmembrane proteins, 104 lipid chain-anchored membrane proteins, 68 GPI-anchored membrane proteins, and 30 are peripheral membrane proteins. In addition, average sequence identity percentages are also calculated for testing dataset test, which are 8.34% for type-I transmembrane proteins, 9.53% for type-II transmembrane proteins, 8.55% for multipass transmembrane proteins, 10.22% for lipid chain-anchored membrane proteins, 11.75% for GPI-anchored membrane proteins, and 5.00% for peripheral membrane proteins. Training and testing datasets are mutually exclusive. The testing dataset consists of 3,160 protein sequences, of which type-I transmembrane proteins are 462, type-II transmembrane proteins are 144, multipass transmembrane proteins are 2,402, lipid chain-anchored membrane proteins are 67, GPI-anchored membrane proteins are 83, and peripheral membrane proteins are 2 sequences. Dataset3 has been passed through different phases; initially, the sequences, annotated with fragments and having ambiguity, are excluded. The original dataset consists of 3,249 membrane protein sequences. The training dataset contains 610 single-pass type-I transmembrane proteins, 312 single-pass type-II transmembrane proteins, 24 single-pass type-III transmembrane proteins, 44 single-pass type-IV transmembrane proteins, 1,316 multipass transmembrane proteins, 151 lipid chain-anchored membrane proteins, 182 GPI-anchored membrane proteins, and 610 are peripheral membrane protein sequences. After the above screening procedure, still the dataset contains some redundancy. Therefore, 30% CD-HIT is applied to remove the redundancy. We have included only less than 30% identity sequences in the dataset. Furthermore, sequences, having length shorter than 50 residues, are removed from the dataset. In addition, 25% similarity cutoff is applied to remove homology between the sequences. Then, the sequences having 25% or greater identity are excluded. Finally, the resultant dataset contains 2,978 membrane protein sequences of which single-pass type-I transmembrane proteins are 576, single-pass type-II transmembrane proteins are 269, single-pass type-III transmembrane proteins are 17, single-pass type-IV transmembrane proteins are 34, multipass transmembrane proteins are 1,285, lipid chain-anchored membrane proteins are 97, GPI-anchored membrane proteins are 154, and peripheral membrane proteins are 546 sequences.
5.2.2. Feature Extraction Schemes

In this study, we have investigated the discrimination power of the three different feature extraction methods namely, DWT, PseAA composition, and SAAC. DWT is widely used in order to explore various parts of a signal, because it has the capability of extracting both spectral and temporal information, simultaneously. DWT divides the signal into two components: detail components and approximation components; more details are provided in Chapter 3. We have adopted DWT as a feature extraction strategy to extract valuable information from protein sequences. First, the protein sequence is transformed into a numeric sequence using Kyte and Doolittle hydrophobic scale [37]. Then, we decomposed the numeric sequence up to level 4 in order to extract all the high-level information. Further, we applied several statistical measures to explore each signal component, such as Shannon entropy, log entropy, energy entropy, variance, mean, max, and min, respectively. Finally, the obtained feature space of 35-D is normalized through Euclidean normalization to bring the value into similar ranges.

PseAA composition is the substitution of conventional AA composition. PseAA composition preserves two kinds of information; one is normalize occurrence frequency of each amino acid, and the other is sequence order information; more details are provided in Chapter 3. In this study, we have used the hydrophobicity and hydrophilicity scales of amino acids for converting membrane protein sequences into numeric sequences. Finally, the total length of the extracted features is 62-D, which is constituted from 20 features of conventional AA composition and 42 sequence order effects information. In this study, the value of $\lambda$ is selected 21, because the best performance is reported on this value.

Sometimes, the vital information contains in a special part of a sequence, which is not possible to extract by conventional methods. Therefore, a method is required to divide the protein sequence into several peptides, and extract information from those peptides. In that situation, SAAC is the best choice, where it splits the protein sequence into several fragments and takes composition of each fragment independently; more details are provided in Chapter 3. In this study, we have split the protein sequence into three different fragments: 25-N terminus, 25-C terminus, and the remaining amino acids.
between these two fragments. As a result, the obtained feature space is of 60-D instead of 20-D, because it contains three different kinds of composition features.

### 5.2.3. Ensemble Classification

In the last decade, the concept of ensemble classification has attracted great attention in Machine learning and Data mining community. In individual classification, each classifier evaluates the data in its own way. Consequently, the predicted results of individual classifiers are diverse in nature, and no single classifier obtains an acceptable level of accuracy. In such circumstances, it would be better to pool the prediction of several classifiers in order to reduce the classification error and achieve optimal results. The aim of the ensemble classification is to develop a model that generates more accurate and precise results. The framework of ensemble classification, combines several basic learners together in order to reduce the variance caused by the peculiarities of a single training set, and hence be able to learn a more expressive concept in ensemble classification than a single classifier [42; 43; 185]. In this study, we have used two types of ensemble classification; simple majority voting based ensemble and GA majority voting based ensemble. The framework of ensemble classification is shown in Figure 5.1.

#### 5.2.3.1. Simple Majority Voting Based Ensemble

Majority voting is a simple ensemble classification technique in which each classifier is assigned an equal weight. Here, we have combined five different classifiers including AdaBoost, RF, SVM, PNN, and KNN (detailed discussion is provided in Chapter 3). The majority voting based ensemble performs its operations in two steps. In the first step, each individual classifier is trained and its prediction is noted down. In the second step, the ensemble classifier is developed by fusing the predictions of the individual classifiers.

\[
EnsC = AdaBoost \oplus RF \oplus SVM \oplus PNN \oplus KNN
\]  

(5.1)

where the symbol \( \oplus \) indicates the fusing operator and EnsC is the ensemble classifier.
The fusing process of the ensemble classifier EnsC works as follow.
Let suppose the success rates of individual classifiers for the protein query $P$ is given by
\[ \{C_1, C_2, \ldots, C_N \} \in \{S_1, S_2, \ldots, S_M \} \]  
(5.2)
where $C_1, C_2, \ldots, C_N$ represents individual classifiers and $S_1, S_2, \ldots, S_M$ denotes membrane protein types.

\[
Y_j = \sum_{i=1}^{N} \delta(C_i, S_j) \quad (j = 1, \ldots, N, N=5)
\]  
(5.3)
where \[ \delta(C_i, S_j) = \begin{cases} 1, & \text{if } C_i \in S_j \\ 0, & \text{otherwise} \end{cases} \]

Finally, the predicted output of the individual classifiers, fused through majority voting, is formulated as:
\[ C_{ensC} = Max\{Y_1, Y_2, \ldots, Y_N\} \]  
(5.4)
where $C_{ensC}$ indicates the predicted output of the ensemble classifier, $Max$ shows selecting the maximum one.
5.2.3.2. GA Majority Voting Based Ensemble

The main drawback of simple majority voting based ensemble is that, it treats all classifiers equally without considering the performance of the classifiers. In order to overcome this drawback, weighted majority voting is used. In weighted majority voting, weight is assigned to each classifier on its performance basis. However, the challenging task is the assignment of optimal weights to classifiers. In this study, we have used GA to find the optimal weights.

GA based ensemble classifier follows the following procedure.

Suppose, \( X \) is a pattern space that contains \( N \) mutually exclusive sets \( X = \{ C_1, C_2, ..., C_N \} \) where \( C_i \) represents a class. The output of the classifiers for the input pattern is \( e_j(x) = \{ v'_j(x) \} \) where \( j \) is the number of classifiers \( j = 1, 2, ..., N \) and \( i \) is the number of classes \( i \in \{ 1, 2, ..., M \} \). It means that classifier \( j \) assigns the input pattern \( x \) to each class \( i \) with a value of \( v'_j(x) \). The final output of the input pattern \( x \) can be \( E_i(x) \) for each class \( i \), which is computed by the weighted sum of measured values of \( v'_j(x) \) and their corresponding weight \( W'_j(f) \) can be formulated as:

\[
E_i(x) = \sum_{j=1}^{N} W'_j v'_j(x) \quad \{ i = 1, 2, ..., M \} \tag{5.5}
\]

Finally, the highest value of \( E_i(x) \) class label is selected for each input pattern \( x \). The weight vector is encoded by a string value called chromosomes. In initial population generation, the weight is assigned to each classifier randomly. Fitness of each individual is computed. On the basis of fitness value population is evolved from generation to generation. At the end, GA gives a set of optimal weights to each classifier. Here, we have used accuracy and MCC as fitness function.

5.3. Experimental Results and Discussion

In order to evaluate the performance of the proposed method, three statistical tests: self-consistency, jackknife, and independent dataset tests are conducted. The performance of the classifiers is examined through accuracy, sensitivity, specificity, MCC, F-measure, and Q-statistics.
5.3.1. Dataset2

The experimental results of the individual and ensemble classifiers using self-consistency, jackknife, and independent dataset tests are presented below.

5.3.1.1. Self-consistency Test

All the individual classifiers as well as the ensemble classifier have yielded prominent results on self-consistency test. Only the predicted result of Mem-EnsSAAC is shown in Table 5.7.

5.3.1.2. Jackknife Test

The success rates of the individual and ensemble classifiers on each feature extraction strategy: DWT, PseAA composition, and SAAC are reported in Table 5.1. The experimental results reveal that classifiers have yielded poor performance using DWT based feature compared to PseAA composition, and SAAC. However, RF outperforms than other classifiers by achieving 80.7% accuracy. In contrast, ensemble classifier has achieved low accuracy compared to individual classifiers where the ensemble classifier is developed by combining the prediction of individual classifiers through simple majority voting. It is realized that the prediction of RF, Adaboost, SVM, PNN, and KNN, using DWT has low diversity. In case of PseAA composition, ensemble classifier has achieved the accuracy of 89.5%, which is 14.6% greater than the accuracy of DWT. Using SAAC, the accuracy of the ensemble classifier (Mem-EnsSAAC) is 92.2%, which is 17.3 and 2.7% higher than that of the DWT, and PseAA composition. In addition, the performance of individual classifiers using SAAC based feature is pronounced compared to DWT and PseAA composition. Other performance measures of Mem-EnsSAAC are 91.0% sensitivity, 92.2% specificity, 0.75 MCC, 0.79 F-measure, and 0.91 Q-Statistics. SVM has obtained the best accuracy of 90.8% in case of individual classifiers. The simulated results reveal that SAAC based model is effective for the prediction of membrane protein types, since it explores information from the three different parts of the protein sequence.
5.3.1.3. Independent Dataset Test

The predicted results of the classifiers using DWT, PseAA composition, and SAAC are reported in Table 5.2. Still the performance of the ensemble classifier using DWT is lower than that of the individual classifiers. In individual classifiers, RF obtains the highest accuracy of 83.8%. In case of PseAA composition, the performance of individual as well as ensemble classifiers has been enhanced. Where the ensemble classifier has obtained an accuracy of 88.6%, which is 7.5% higher than that of DWT.

Table 5.1. Success rates of classifiers on Jackknife test using individual feature extraction strategies.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F measure</th>
<th>Q Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>74.4</td>
<td>66.2</td>
<td>75.3</td>
<td>0.34</td>
<td>0.48</td>
<td>0.89</td>
</tr>
<tr>
<td>R F</td>
<td>80.7</td>
<td>72.1</td>
<td>81.6</td>
<td>0.46</td>
<td>0.56</td>
<td>0.93</td>
</tr>
<tr>
<td>PNN</td>
<td>74.7</td>
<td>66.8</td>
<td>75.6</td>
<td>0.35</td>
<td>0.48</td>
<td>0.92</td>
</tr>
<tr>
<td>KNN</td>
<td>74.8</td>
<td>66.4</td>
<td>75.8</td>
<td>0.35</td>
<td>0.48</td>
<td>0.90</td>
</tr>
<tr>
<td>SVM</td>
<td>77.8</td>
<td>68.6</td>
<td>78.8</td>
<td>0.40</td>
<td>0.52</td>
<td>0.93</td>
</tr>
<tr>
<td>EnsC</td>
<td>74.9</td>
<td>67.0</td>
<td>75.8</td>
<td>0.36</td>
<td>0.48</td>
<td>0.91</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>81.1</td>
<td>72.3</td>
<td>82.0</td>
<td>0.47</td>
<td>0.57</td>
<td>0.93</td>
</tr>
<tr>
<td>PseAA composition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>85.0</td>
<td>80.1</td>
<td>85.5</td>
<td>0.57</td>
<td>0.65</td>
<td>0.93</td>
</tr>
<tr>
<td>R F</td>
<td>85.4</td>
<td>80.6</td>
<td>85.3</td>
<td>0.58</td>
<td>0.65</td>
<td>0.93</td>
</tr>
<tr>
<td>PNN</td>
<td>88.6</td>
<td>87.6</td>
<td>88.6</td>
<td>0.67</td>
<td>0.73</td>
<td>0.95</td>
</tr>
<tr>
<td>KNN</td>
<td>87.8</td>
<td>89.3</td>
<td>87.5</td>
<td>0.66</td>
<td>0.72</td>
<td>0.94</td>
</tr>
<tr>
<td>SVM</td>
<td>88.9</td>
<td>87.1</td>
<td>89.0</td>
<td>0.67</td>
<td>0.73</td>
<td>0.95</td>
</tr>
<tr>
<td>EnsC</td>
<td>89.5</td>
<td>89.4</td>
<td>86.3</td>
<td>0.69</td>
<td>0.74</td>
<td>0.94</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>90.5</td>
<td>89.8</td>
<td>90.4</td>
<td>0.72</td>
<td>0.76</td>
<td>0.93</td>
</tr>
<tr>
<td>SAAC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>88.1</td>
<td>85.5</td>
<td>88.2</td>
<td>0.65</td>
<td>0.71</td>
<td>0.89</td>
</tr>
<tr>
<td>R F</td>
<td>88.2</td>
<td>85.8</td>
<td>88.3</td>
<td>0.65</td>
<td>0.71</td>
<td>0.88</td>
</tr>
<tr>
<td>PNN</td>
<td>90.4</td>
<td>90.0</td>
<td>90.4</td>
<td>0.72</td>
<td>0.76</td>
<td>0.91</td>
</tr>
<tr>
<td>KNN</td>
<td>90.4</td>
<td>89.9</td>
<td>90.3</td>
<td>0.71</td>
<td>0.76</td>
<td>0.89</td>
</tr>
<tr>
<td>SVM</td>
<td>90.8</td>
<td>89.8</td>
<td>90.8</td>
<td>0.72</td>
<td>0.77</td>
<td>0.91</td>
</tr>
<tr>
<td>EnsC</td>
<td>92.2</td>
<td>91.0</td>
<td>92.2</td>
<td>0.75</td>
<td>0.79</td>
<td>0.91</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>92.4</td>
<td>91.1</td>
<td>92.5</td>
<td>0.76</td>
<td>0.80</td>
<td>0.92</td>
</tr>
</tbody>
</table>
Table 5.2. Success rates of classifiers on Independent dataset test using individual feature extraction strategies.

<table>
<thead>
<tr>
<th>Methods</th>
<th>DWT</th>
<th>PseAA composition</th>
<th>SAAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Sensitivity</td>
<td>Specificity</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>74.1</td>
<td>70.4</td>
<td>74.5</td>
</tr>
<tr>
<td>R F</td>
<td>83.8</td>
<td>75.3</td>
<td>84.8</td>
</tr>
<tr>
<td>PNN</td>
<td>80.9</td>
<td>71.2</td>
<td>82.1</td>
</tr>
<tr>
<td>KNN</td>
<td>78.5</td>
<td>69.0</td>
<td>79.7</td>
</tr>
<tr>
<td>SVM</td>
<td>82.1</td>
<td>74.4</td>
<td>83.0</td>
</tr>
<tr>
<td>EnsC</td>
<td>81.4</td>
<td>73.7</td>
<td>82.3</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>84.0</td>
<td>74.9</td>
<td>85.1</td>
</tr>
<tr>
<td></td>
<td>PseAA composition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>84.1</td>
<td>82.5</td>
<td>84.1</td>
</tr>
<tr>
<td>R F</td>
<td>87.9</td>
<td>81.0</td>
<td>88.7</td>
</tr>
<tr>
<td>PNN</td>
<td>88.1</td>
<td>81.7</td>
<td>88.9</td>
</tr>
<tr>
<td>KNN</td>
<td>86.3</td>
<td>81.8</td>
<td>86.8</td>
</tr>
<tr>
<td>SVM</td>
<td>87.3</td>
<td>80.6</td>
<td>88.2</td>
</tr>
<tr>
<td>EnsC</td>
<td>88.9</td>
<td>84.6</td>
<td>89.3</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>89.6</td>
<td>84.6</td>
<td>90.1</td>
</tr>
<tr>
<td></td>
<td>SAAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>86.5</td>
<td>82.8</td>
<td>86.8</td>
</tr>
<tr>
<td>R F</td>
<td>90.1</td>
<td>84.4</td>
<td>90.8</td>
</tr>
<tr>
<td>PNN</td>
<td>89.3</td>
<td>82.9</td>
<td>90.0</td>
</tr>
<tr>
<td>KNN</td>
<td>87.6</td>
<td>83.1</td>
<td>88.2</td>
</tr>
<tr>
<td>SVM</td>
<td>89.4</td>
<td>84.1</td>
<td>90.1</td>
</tr>
<tr>
<td>EnsC</td>
<td>92.0</td>
<td>88.4</td>
<td>92.4</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>92.2</td>
<td>88.2</td>
<td>92.6</td>
</tr>
</tbody>
</table>

On the other hand, using SAAC based features; the performance of the classifiers is enhanced, significantly. The accuracy of Mem-EnsSAAC is 92.0%, which is 10.6% and 3.1% higher than that of the DWT and PseAA composition, respectively. In addition, it sensitivity is 88.4%, specificity 92.4%, MCC 0.74, F-measure 0.79, and Q-Statistics 0.95. Among individual classifiers, the highest accuracy of 90.1% has been achieved by RF.

### 5.3.1.4. Hybrid Features vs. NPE based Selected Features

Apart from the individual feature extraction strategies, the performance of the classifiers is also analyzed using hybrid features. The feature extraction strategies are concatenated in the form of two hybrid models: hybrid1 and hybrid2. Hybrid1 is the integration of PseAA composition and SAAC whereas hybrid2 is the concatenation of DWT, PseAA
composition, and SAAC. Table 5.3 shows the performance of individual as well as ensemble classifiers coupled with hybrid features using jackknife test. Using hybrid1 model, SVM and PNN have yielded better accuracy of 91.6% among individual classifiers. Accuracy of the both classifiers is similar; however, MCC of the SVM is better than that of PNN.

Table 5.3. Classification outcomes of the Jackknife test using hybrid entire and reduced features.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Jackknife Test</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Sensitivity</td>
<td>Specificity</td>
<td>MCC</td>
<td>F measure</td>
<td>Q Statistics</td>
</tr>
<tr>
<td>Hybrid1(SAAC +PseAA composition)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>88.0</td>
<td>85.8</td>
<td>88.1</td>
<td>0.65</td>
<td>0.71</td>
<td>0.90</td>
</tr>
<tr>
<td>RF</td>
<td>88.7</td>
<td>86.6</td>
<td>88.8</td>
<td>0.67</td>
<td>0.72</td>
<td>0.92</td>
</tr>
<tr>
<td>PNN</td>
<td>91.6</td>
<td>91.3</td>
<td>91.5</td>
<td>0.75</td>
<td>0.79</td>
<td>0.93</td>
</tr>
<tr>
<td>KNN</td>
<td>91.5</td>
<td>91.2</td>
<td>91.3</td>
<td>0.74</td>
<td>0.78</td>
<td>0.91</td>
</tr>
<tr>
<td>SVM</td>
<td>91.6</td>
<td>91.2</td>
<td>92.0</td>
<td>0.76</td>
<td>0.80</td>
<td>0.94</td>
</tr>
<tr>
<td>EnsC</td>
<td>92.4</td>
<td>92.4</td>
<td>91.8</td>
<td>0.77</td>
<td>0.81</td>
<td>0.93</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>92.6</td>
<td>92.7</td>
<td>92.4</td>
<td>0.77</td>
<td>0.81</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>Hybrid1 (NPE)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>86.8</td>
<td>85.0</td>
<td>86.9</td>
<td>0.63</td>
<td>0.69</td>
<td>0.88</td>
</tr>
<tr>
<td>RF</td>
<td>88.3</td>
<td>89.1</td>
<td>88.0</td>
<td>0.67</td>
<td>0.73</td>
<td>0.91</td>
</tr>
<tr>
<td>PNN</td>
<td>89.4</td>
<td>88.7</td>
<td>89.4</td>
<td>0.69</td>
<td>0.74</td>
<td>0.90</td>
</tr>
<tr>
<td>KNN</td>
<td>89.5</td>
<td>88.6</td>
<td>89.4</td>
<td>0.69</td>
<td>0.74</td>
<td>0.93</td>
</tr>
<tr>
<td>SVM</td>
<td>90.1</td>
<td>89.2</td>
<td>90.1</td>
<td>0.71</td>
<td>0.75</td>
<td>0.93</td>
</tr>
<tr>
<td>EnsC</td>
<td>90.5</td>
<td>89.8</td>
<td>90.9</td>
<td>0.72</td>
<td>0.76</td>
<td>0.92</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>91.8</td>
<td>91.2</td>
<td>91.3</td>
<td>0.73</td>
<td>0.78</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>Hybrid2 (SAAC+PseAA composition+ DWT)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>87.8</td>
<td>86.3</td>
<td>87.9</td>
<td>0.64</td>
<td>0.70</td>
<td>0.87</td>
</tr>
<tr>
<td>RF</td>
<td>88.2</td>
<td>86.0</td>
<td>88.3</td>
<td>0.65</td>
<td>0.71</td>
<td>0.91</td>
</tr>
<tr>
<td>PNN</td>
<td>90.4</td>
<td>90.1</td>
<td>90.3</td>
<td>0.72</td>
<td>0.76</td>
<td>0.89</td>
</tr>
<tr>
<td>KNN</td>
<td>90.5</td>
<td>90.1</td>
<td>90.5</td>
<td>0.72</td>
<td>0.77</td>
<td>0.91</td>
</tr>
<tr>
<td>SVM</td>
<td>90.8</td>
<td>90.4</td>
<td>90.7</td>
<td>0.73</td>
<td>0.77</td>
<td>0.87</td>
</tr>
<tr>
<td>EnsC</td>
<td>92.3</td>
<td>92.5</td>
<td>91.1</td>
<td>0.75</td>
<td>0.79</td>
<td>0.88</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>92.4</td>
<td>91.8</td>
<td>92.5</td>
<td>0.76</td>
<td>0.80</td>
<td>0.89</td>
</tr>
<tr>
<td></td>
<td>Hybrid2 (NPE)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>87.0</td>
<td>84.9</td>
<td>87.1</td>
<td>0.63</td>
<td>0.69</td>
<td>0.85</td>
</tr>
<tr>
<td>RF</td>
<td>88.1</td>
<td>88.9</td>
<td>87.8</td>
<td>0.67</td>
<td>0.72</td>
<td>0.89</td>
</tr>
<tr>
<td>PNN</td>
<td>89.7</td>
<td>88.5</td>
<td>89.7</td>
<td>0.70</td>
<td>0.75</td>
<td>0.91</td>
</tr>
<tr>
<td>KNN</td>
<td>89.6</td>
<td>88.5</td>
<td>89.6</td>
<td>0.69</td>
<td>0.74</td>
<td>0.90</td>
</tr>
<tr>
<td>SVM</td>
<td>89.9</td>
<td>88.2</td>
<td>90.1</td>
<td>0.70</td>
<td>0.75</td>
<td>0.89</td>
</tr>
<tr>
<td>EnsC</td>
<td>90.8</td>
<td>89.4</td>
<td>90.7</td>
<td>0.71</td>
<td>0.76</td>
<td>0.90</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>91.0</td>
<td>90.0</td>
<td>91.0</td>
<td>0.72</td>
<td>0.77</td>
<td>0.89</td>
</tr>
</tbody>
</table>

In contrast, the ensemble classifier has achieved 92.4% accuracy, which is not only higher than the individual classifiers using hybrid features but also higher than the
individual and ensemble classifiers using individual feature extraction strategy. Hybrid1 model is also tested using independent dataset test, where the predicted results of the classifiers are reported in Table 5.4.

Table 5.4. Classification outcomes of the Independent dataset test using hybrid entire and reduced features.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Independent Dataset Test</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Accuracy</td>
<td>Sensitivity</td>
<td>Specificity</td>
<td>MCC</td>
</tr>
<tr>
<td>Hybrid1(SAAC+PseAA composition)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>85.8</td>
<td>82.1</td>
<td>86.0</td>
<td>0.59</td>
<td>0.66</td>
</tr>
<tr>
<td>R F</td>
<td>89.0</td>
<td>82.7</td>
<td>89.5</td>
<td>0.65</td>
<td>0.71</td>
</tr>
<tr>
<td>PNN</td>
<td>88.7</td>
<td>82.4</td>
<td>89.3</td>
<td>0.65</td>
<td>0.71</td>
</tr>
<tr>
<td>KNN</td>
<td>87.2</td>
<td>82.9</td>
<td>87.5</td>
<td>0.62</td>
<td>0.69</td>
</tr>
<tr>
<td>SVM</td>
<td>89.8</td>
<td>85.7</td>
<td>90.0</td>
<td>0.68</td>
<td>0.74</td>
</tr>
<tr>
<td>EnsC</td>
<td>89.9</td>
<td>85.5</td>
<td>90.3</td>
<td>0.69</td>
<td>0.74</td>
</tr>
<tr>
<td>EnsC_GA</td>
<td>91.3</td>
<td>86.7</td>
<td>91.8</td>
<td>0.72</td>
<td>0.77</td>
</tr>
</tbody>
</table>

| Hybrid1 (NPE) |                           |          |             |             |     |           |              |
| AdaBoost       | 83.2                     | 83.1     | 83.8       | 0.55        | 0.62 | 0.89      |
| R F            | 85.3                     | 83.2     | 86.4       | 0.62        | 0.67 | 0.89      |
| PNN            | 87.8                     | 84.6     | 88.2       | 0.63        | 0.69 | 0.91      |
| KNN            | 85.9                     | 82.8     | 86.1       | 0.59        | 0.66 | 0.88      |
| SVM            | 86.7                     | 83.3     | 87.8       | 0.62        | 0.66 | 0.92      |
| EnsC           | 87.2                     | 83.1     | 90.1       | 0.65        | 0.71 | 0.90      |
| EnsC_GA        | 88.5                     | 84.1     | 89.3       | 0.67        | 0.73 | 0.91      |

| Hybrid2 (SAAC+PseAA composition+ DWT) |                           |          |             |             |     |           |              |
| AdaBoost       | 85.0                     | 84.0     | 84.9       | 0.59        | 0.66 | 0.94      |
| R F            | 89.4                     | 83.8     | 90.0       | 0.67        | 0.73 | 0.95      |
| PNN            | 88.2                     | 83.5     | 88.7       | 0.64        | 0.71 | 0.95      |
| KNN            | 87.2                     | 82.9     | 87.7       | 0.62        | 0.69 | 0.96      |
| SVM            | 89.7                     | 85.6     | 90.1       | 0.68        | 0.74 | 0.96      |
| EnsC           | 91.3                     | 87.2     | 90.5       | 0.70        | 0.75 | 0.95      |
| EnsC_GA        | 91.5                     | 88.4     | 91.9       | 0.72        | 0.77 | 0.94      |

| Hybrid2 (NPE) |                           |          |             |             |     |           |              |
| AdaBoost       | 84.3                     | 83.6     | 85.1       | 0.57        | 0.65 | 0.89      |
| R F            | 88.2                     | 82.1     | 89.0       | 0.65        | 0.71 | 0.89      |
| PNN            | 86.9                     | 82.7     | 87.3       | 0.63        | 0.70 | 0.93      |
| KNN            | 86.7                     | 80.4     | 87.2       | 0.61        | 0.70 | 0.91      |
| SVM            | 88.5                     | 83.9     | 90.5       | 0.67        | 0.72 | 0.92      |
| EnsC           | 90.1                     | 86.6     | 90.8       | 0.69        | 0.74 | 0.93      |
| EnsC_GA        | 90.6                     | 86.9     | 91.0       | 0.70        | 0.75 | 0.92      |

Ensemble classifier has achieved the highest success rates, whereas SVM has yielded the highest accuracy among the individual classifiers. The performance of the classifiers is also examined using hybrid2 model. After ascertaining, we observed that ensemble
classifier efficiently utilizes DWT that is why its performance suffers less compared to individual classifiers. In case of independent dataset test, the ensemble classifier has yielded 91.3% accuracy. Using jackknife test, the performance of the ensemble classifier coupling with hybrid2 model is well compared to DWT, PseAA composition, and SAAC individually. Overall, the performance of hybrid1 model is better, but there is only about 0.2% difference between Mem-EnsSAAC and hybrid1 model accuracies.

On the other hand, the computational cost and complexity have increased due to the high dimensionality, 60-D for Mem-EnsSAAC vs. 122-D for hybrid1 model.

In order to handle the problems regarding high dimensionality, we have employed NPE feature reduction technique. The performance of the classifiers is examined under the effect of various dimensionalities of hybrid models. Finally, the best result is reported on 100 dimensionalities. We have realized that NPE has reduced the dimension from 122-D and 157-D to 100-D, resulting in decreased performance of the classifiers. In case of hybrid1 model, 1.9 and 2.7% decrease is observed in the performance of ensemble classifier due to NPE, using jackknife and independent dataset tests, respectively. Likewise, in case of hybrid2 model, the performance of ensemble classifier is decreased by 1.5 and 1.2% using jackknife, and independent dataset tests, respectively.

5.3.1.5. Ensemble through GA

The major problem in simple majority voting is that it treats all classifiers with equal weights; however, GA assigns different weight to each classifier. Therefore, in order to enhance the performance of the ensemble classifier, the prediction of the individual classifiers is fused through optimization technique GA. After applying GA based majority voting, 6.2 and 2.7% improvement has been recorded using jackknife and independent dataset tests, respectively, in case of DWT as shown in Figure 5.2. It reveals that there exists some diversity in the prediction of individual classifiers, but is not explored due to simple majority voting. In all feature extraction strategies along with hybrid models, GA has enhanced the performance of ensemble classifiers. The highest accuracy of 92.6% has been yielded by the ensemble classifier using hybrid1 model, as depicted in Figure 5.3, similarly, 92.4% of accuracy has been obtained by Mem-
EnsSAAC. Again, there is a slightly difference between in accuracy of hybrid1 model and SAAC, which is 0.2%.

Figure 5.2. Performance of DWT-EnsC using GA for jackknife test

Figure 5.3. Performance of Hybrid-EnsC using GA for jackknife test.
In contrast, SAAC is 60-D and hybrid1 model is 122-D. Finally, we have reckoned that SAAC has efficiently discriminated the types of membrane protein compared to the rest of individual and hybrid models.

5.3.1.6. Individual Membrane Protein Types

The accuracy of ensemble classifiers for each type of membrane protein is listed in Table 5.5a-b. DWT based ensemble classifier has achieved reasonably bad accuracy of 26.5, 30.8 and 13.2% for type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins, respectively. In contrast, it has yielded comparatively good accuracies of 59.3, 90.1, and 40.0% for type-I, multipass transmembrane, and peripheral membrane proteins. The empirical results reveal that the overall accuracy of DWT based prediction is disgraced owing to type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins. Here the main issue with DWT based feature is the uneven length of protein sequences, because the average sequence length of type-I transmembrane, type-II transmembrane, multipass transmembrane, lipid chain-anchored, GPI-anchored, and peripheral membrane proteins is 772, 366, 521, 271, 456, and 467 residues long, respectively. Accordingly, the length of type-II transmembrane and lipid chain-anchored membrane protein sequences is shorter than that of the rest of types. However, the level of decomposition is 4, which has introduced feature redundancy, because of the short residue sequences. In case of PseAA composition based ensemble classifier, the accuracy for type-I transmembrane is 91.9%, type-II transmembrane 45.7%, multipass transmembrane 97.4%, lipid chain-anchored 63.5%, GPI-anchored membrane 25.0%, and peripheral membrane proteins is 20.0%. Almost, the same issue arises with PseAA composition; as the tier value increases, the accuracy of the short residue classes decreases. In contrast, if the tier value decreases, the important information about the long residue sequences will be lost.
Table 5.5a-b. Success rates of ensemble classifiers for each type of membrane proteins using Jackknife test

<table>
<thead>
<tr>
<th>Membrane types</th>
<th>DWT-EnsC</th>
<th>PseAA-EnsC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>MCC</td>
</tr>
<tr>
<td>Type-I</td>
<td>51.3</td>
<td>0.24</td>
</tr>
<tr>
<td>Type-II</td>
<td>26.5</td>
<td>0.15</td>
</tr>
<tr>
<td>Multipass</td>
<td>88.5</td>
<td>0.48</td>
</tr>
<tr>
<td>Lipid</td>
<td>30.8</td>
<td>0.45</td>
</tr>
<tr>
<td>GPI</td>
<td>13.2</td>
<td>0.42</td>
</tr>
<tr>
<td>Peripheral</td>
<td>40.0</td>
<td>0.35</td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th>Membrane types</th>
<th>Mem-EnsSAAC</th>
<th>Hybrid-EnsC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>MCC</td>
</tr>
<tr>
<td>Type-I</td>
<td>93.8</td>
<td>0.72</td>
</tr>
<tr>
<td>Type-II</td>
<td>51.0</td>
<td>0.62</td>
</tr>
<tr>
<td>Multipass</td>
<td>98.0</td>
<td>0.81</td>
</tr>
<tr>
<td>Lipid</td>
<td>68.3</td>
<td>0.78</td>
</tr>
<tr>
<td>GPI</td>
<td>36.7</td>
<td>0.76</td>
</tr>
<tr>
<td>Peripheral</td>
<td>36.6</td>
<td>0.73</td>
</tr>
</tbody>
</table>

In contrast, the prediction performance of Mem-EnsSAAC for each membrane protein type is higher than that of DWT and PseAA composition based ensembles. It has obtained the accuracy of 93.8, 51.0, 98.0, 68.3, 36.7, and 36.6% for type-I transmembrane, type-II transmembrane, multipass transmembrane, lipid chain-anchored, GPI-anchored, and peripheral membrane proteins, respectively. Table 5.5a-b also reported MCC and F-measure of each membrane protein type. The performance of hybrid based ensemble Hybrid-EnsC is better for type-II transmembrane and lipid chain-anchored membrane proteins, but not for other types. It has yielded 91.6, 53.6, 97.8, 70.2, 33.8, and 33.3% accuracies for type-I transmembrane, type-II transmembrane, multipass transmembrane, lipid chain-anchored, GPI-anchored membrane, and peripheral membrane proteins, respectively. The accuracy of ensemble classifier for each membrane protein type is depicted in Figure 5.4.

The heuristic results show that the performance of the classifiers is affected owing to the imbalance nature of the datasets. Still, the performance of the Mem-EnsSAAC is high for almost each type of membrane proteins than that of PseAA composition, DWT, and
hybrid models. In addition, Mem-EnsSAAC is not affected due to the uneven length of protein sequences. The MCC of Mem-EnsSAAC is (0.76, 0.73) and F-measure (0.80, 0.79), respectively, using jackknife and independent dataset tests.

Figure 5.4. Accuracy of various ensemble classifiers for each type of membrane proteins using Jackknife test.

5.3.1.7. Comparison with the Existing Approaches

The comparison of the proposed method Mem-EnsSAAC with existing methods on the same dataset in the literature is shown in Table 5.6. Mem-EnsSAAC generates the highest success rates in all the three tests including self-consistency, jackknife, and independent dataset tests, which are 99.9, 92.4, and 92.2%, respectively. The pioneer work on this dataset has been carried out by Chou and Cai [32]. They have employed two feature extraction strategies AA composition and amphipathic PseAA composition using four classification algorithms namely, least hamming distance, least Euclidean distance, ProtLock, and CDA.
Table 5.6. Comparative analysis between the proposed approach and existing approaches on dataset2

<table>
<thead>
<tr>
<th>Methods</th>
<th>Self-consistency test (%)</th>
<th>Jackknife test (%)</th>
<th>Independent dataset test (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amphipathic PseAA &amp; Least Hamming Distance [32]</td>
<td>---</td>
<td>74.1</td>
<td>74.3</td>
</tr>
<tr>
<td>Amphipathic PseAA &amp; Least Euclidean Distance [32]</td>
<td>---</td>
<td>74.6</td>
<td>75.2</td>
</tr>
<tr>
<td>Amphipathic PseAA &amp; Prot-Lock [32]</td>
<td>---</td>
<td>77.6</td>
<td>82.9</td>
</tr>
<tr>
<td>Amphipathic PseAA &amp; CDA [32]</td>
<td>---</td>
<td>86.1</td>
<td>90.6</td>
</tr>
<tr>
<td>Proposed Mem – EnsSAAC</td>
<td>99.9</td>
<td>92.4</td>
<td>92.2</td>
</tr>
</tbody>
</table>

Among these classification algorithms, the highest accuracy is obtained by CDA in conjunction with amphipathic PseAA composition, which is 86.15 and 90.60% using jackknife and independent dataset tests, respectively. Therefore, we have made a comparison with the highest previously reported results. Our proposed Mem-EnsSAAC has obtained 6.3 and 1.6% higher accuracies using jackknife and independent dataset test, respectively. A similar improvement has been reported in other measures. In conclusion, we have observed that this effectual and significant enhancement is due to the powerful discrimination capabilities of SAAC and the learning capability and robustness of the majority voting based ensemble method.

5.3.2. Dataset3

We have only utilized SVM, KNN, and PNN for dataset3. The best feature extraction strategy SAAC and only jackknife test is brought under consideration. The importance of this dataset is that it contains eight types of membrane protein, but the problem with it is the imbalance nature of data. The overall and each membrane protein type prediction accuracies of the individual and ensemble classifier are reported in Table 5.7. In individual classifiers, the highest accuracy of 84.2% is obtained by SVM. In contrast, ensemble classifier has obtained better results compared to individual classifiers.
Table 5.7. Comparative analysis between the proposed approach and existing approaches on dataset3

<table>
<thead>
<tr>
<th>Membrane Types</th>
<th>KNN</th>
<th>PNN</th>
<th>SVM</th>
<th>Proposed Mem-EnsSAAC</th>
<th>MemType-2L [9]</th>
<th>Mahdavi [40]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type-I</td>
<td>89.0</td>
<td>89.0</td>
<td>89.2</td>
<td>91.5</td>
<td>87.2</td>
<td>83.7</td>
</tr>
<tr>
<td>Type-II</td>
<td>68.8</td>
<td>69.1</td>
<td>64.3</td>
<td>73.6</td>
<td>72.8</td>
<td>53.2</td>
</tr>
<tr>
<td>Type-III</td>
<td>58.8</td>
<td>47.0</td>
<td>52.9</td>
<td>76.5</td>
<td>41.7</td>
<td>29.2</td>
</tr>
<tr>
<td>Type-IV</td>
<td>67.6</td>
<td>67.6</td>
<td>79.4</td>
<td>85.3</td>
<td>75.0</td>
<td>50.0</td>
</tr>
<tr>
<td>Multipass</td>
<td>89.9</td>
<td>91.0</td>
<td>92.2</td>
<td>92.8</td>
<td>95.7</td>
<td>92.2</td>
</tr>
<tr>
<td>Lipid</td>
<td>73.2</td>
<td>72.2</td>
<td>62.9</td>
<td>75.2</td>
<td>56.3</td>
<td>45.0</td>
</tr>
<tr>
<td>GPI</td>
<td>74.7</td>
<td>74.7</td>
<td>80.5</td>
<td>81.8</td>
<td>68.7</td>
<td>67.0</td>
</tr>
<tr>
<td>Peripheral</td>
<td>69.8</td>
<td>71.1</td>
<td>75.8</td>
<td>74.7</td>
<td>80.5</td>
<td>65.9</td>
</tr>
<tr>
<td>Overall Accuracy</td>
<td>82.4</td>
<td>83.0</td>
<td>84.2</td>
<td><strong>86.2</strong></td>
<td>85.0</td>
<td>76.8</td>
</tr>
</tbody>
</table>

It has achieved 86.2% accuracy, which shows the superiority against individual classifiers. The performance of the ensemble classifier for individual type of membrane protein is prominent, because each type has obtained above 73% accuracy. The obtained accuracies for type-I, type-II, type-III, type-IV transmembrane, multipass transmembrane, lipid chain-anchored, GPI-anchored membrane, and peripheral membrane proteins are 91.5, 73.6, 76.5, 85.3, 92.8, 75.2, 81.8, and 74.7%, respectively. The proposed method is also compared with the Chou and Shen method’s (MemType-2L) [9] and Mahdavi’s method [40] as shown in Table 5.7. The success rate of Mem-EnsSAAC is 1.2 and 9.4% higher than the predicted results of Chou and Shen method’s MemType-2L [9] and Mahdavi’s [40] method, respectively, and is best reported so far.
6. Prediction of Membrane Protein Types Using Evolutionary Information and BCH Coding

6.1. Introduction

In the previous chapter, the problem of membrane protein types has been handled through ensemble classification. In this chapter, we endeavor to tackle the problem of membrane protein types by powerful feature extraction schemes and error correction code. In order to reduce the classification errors and enhance the reliability of a model, it is very imperative to extract all the relevant information from protein sequences, which play a key role in discriminating proteins. Therefore, we have utilized an evolutionary information technique PSSM for representation of membrane protein sequences. Evolutionary information has the ability of determining the motifs of amino acids. These motifs expose the evidence of evolutionary constraint and information about the structure. Sometimes, individual methods do not express all the required information, which becomes a cause of disgracing the performance of the classifier. To overcome this weakness and further strengthen the discrimination power of a features space, we combine biochemical features (SAAC) with evolutionary information (PSSM). SAAC based features have more importance compared to other biochemical properties, because it excerpts the information from the various parts of a protein sequence. In addition, it gives weight to different parts of protein sequence. SVM is used as a classification algorithm. Owing to its good characteristics, SVM is considered the most prominent and stable classifier. It does not suffer from curse of dimensionality and has no local minima. SVM is evaluated through various kernels such as linear, polynomial, and radial based functions. Apart from it, error correction code is incorporated with SVM to reduce the generalization error. Two benchmark datasets are utilized to investigate the performance
of the proposed model. The performance of classifiers is assessed through accuracy, sensitivity, specificity, MCC, and F-measure.

6.2. Materials and Methods

In this section, first datasets are discussed. Then feature extraction scheme, error correction code and proposed system are explained.

6.2.1. Datasets

In order to assess the learning and discriminating power of a classification model, we have exercised two benchmark datasets: dataset1 and dataset2. Dataset1 contains only integral membrane protein types, while dataset2 comprises of both integral and peripheral membrane protein types. Dataset1 has five, whereas dataset2 has six types of membrane proteins. The details about dataset1 and dataset2 are provided in Chapter 4 and 5, respectively. Dataset1 contains 2,059 membrane protein sequences. The training set consists of 435 type-I transmembrane proteins, 152 type-II transmembrane proteins, 1,311 multipass transmembrane proteins, 51 lipid chain-anchored membrane proteins, and 110 GPI-anchored membrane protein sequences. Dataset2 has 2,628 membrane protein sequences for training; 372 of type-I transmembrane proteins, 151 of type-II transmembrane proteins, 1,903 of multipass transmembrane proteins, 104 of lipid chain-anchored membrane proteins, 68 of GPI-anchored membrane proteins, and 30 of peripheral membrane protein sequences.

6.2.2. Feature Extraction Schemes

In this section, we provide details about feature extraction scheme PSSM, which is given below.

6.2.2.1. Position Specific Scoring Matrix

Recently, small amount of identified sequences are available in the protein database, which are not sufficient for protein prediction. In contrast, huge amounts of unidentified sequences exist in the public databases. These unidentified sequences have conservative evolutionary information about each sequence position, which may help in recognizing
the motifs of amino acid substitutions, through protein sequence alignment. The pattern of residue substitution reveals the evidence of evolutionary constraint and information about the structure. Evolutionary information is derived from homologous sequences through multiple sequence alignments, which calculate the estimated effective result of the changes of amino acids at each position in evolutionary processes. In this study, we have used PSSM for extracting evolutionary information from membrane protein sequences. PSSM is widely used for pattern representation in biological sequences. PSSM is evolutionary profiles and patterns based representative that exploits multiple alignments and information regarding protein families. PSSM has been initially used for identifying distant information related to the proteins. Its profiles determine the correlation and dependency among the neighboring residues of each amino acid in a protein (for more details see Chapter 3). In this study, we have selected the λ value as 49, because the shortest protein sequence in benchmark dataset is 50 residues long. The obtained feature space is 1000-D.

6.2.3. BCH Coding

BCH coding is a multilevel, cyclic, variable-length, and error-correcting digital code utilized to correct multiple random error patterns. It can correct almost 25% of errors of the total number of digits [188]. It can also be adopted with multilevel phase shifting key, whenever the power of a prime number or level is a prime number. It is the generalization of hamming code that allows multiple random error corrections. BCH coding was initially introduced by Hocquenghem in 1959, and independently in 1960 by Bose and Ray-Chaudhuri [189]. The acronym of BCH is based on the name of the innovators. Initially, BCH coding was used for binary problems, with binary codes of length $2^m-1$ for some integer $m$. Later, it has been extended into binary codes with symbols from Galois field $GF (q)$ by Gorenstein and Ziegler [190]. Galois field is a finite field order, which is always a prime or power of a prime number. $GF (q)$ is called the prime field of order $q$ where $q$ elements are 0, 1… $q-1$. BCH coding can be determined by a polynomial generator.

Binary BCH coding can be represented with parameters $(n, k, t)$, where $n$ represents the length of the codeword, $k$ is the size of message, and $t$ is randomly correcting bits code.
For any arbitrary integers $m \geq 3$ and $t < 2^{m-1}$, there exists a primitive BCH coding with the parameters $n = 2^m - 1$, $n - k \leq mt$, and $d_{\text{min}} \geq 2t + 1$, where $n$ is a block length, $n-k$ is the number of parity check bits, and $d_{\text{min}}$ denotes the minimum hamming distance between two vectors. This code can correct $t$ random errors over a span of $2^m - 1$ bit positions. In this work, we have exercised different combinations of BCH coding, and consequently, achieved the best result on BCH (31, 6, 7) encoding.

### 6.2.4. Proposed Method

The first step for developing an accurate and reliable prediction system is to formulate the protein sequences with an effective mathematical expression. This mathematical expression should accurately reflect their intrinsic relationship with the attribute to be predicted [191]. Sometimes, single sequence representation method does not explore all the relevant information regarding proteins. Due to insufficient information, the performance of the classifier is badly affected. Therefore, we have adopted the concept of hybridization to dig out all the imperative and relevant information, which might be helpful in identifying the appropriate classifier for the prediction of membrane protein types. In this study, we have combined the biochemical and evolutionary information of membrane protein sequences. The membrane protein sequences are expressed by two powerful sequence representation methods including SAAC and PSSM. Further, we have fused these two types of features to enhance the discriminative power of the classification algorithms. Consequently, the dimension of the features space reached to 1060 (1000+60)-D. In addition, we have merged the error correction code BCH with classification algorithms in order to reduce the generalization error of the classifier. BCH coding has the ability to correct the classification errors.

First, create an $n$-bit binary code for each target class, which has minimum hamming distance $d$, where $n \geq \lceil \log_2 k \rceil + d$ and $k$ is the number of classes. Then, each class is assigned a corresponding unique binary code. The unique binary code for each class is reported in Table 6.1.
Table 6.1. A unique binary code for each class

<table>
<thead>
<tr>
<th>Class</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 0 0 0 0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>0 1 1 0 1 0</td>
</tr>
<tr>
<td>3</td>
<td>0 0 0 0 1 0</td>
</tr>
<tr>
<td>4</td>
<td>1 1 0 0 0 0</td>
</tr>
<tr>
<td>5</td>
<td>1 1 0 0 1 0</td>
</tr>
<tr>
<td>6</td>
<td>0 0 1 1 0 0</td>
</tr>
</tbody>
</table>

After that, each instance of the target class in the label set is substituted by its defined unique binary code. Then, BCH encoding algorithm is applied to transform the label set into $N$-bit encoded label set. Here, we have used BCH $(31, 6, 7)$ where the length of the codeword is $N=31$, size of the message $n=6$, and error correction bit $t=7$. After BCH encoding, the multiclass problem is decomposed into $N$-binary problem, because SVM treats multiclass problem as one-vs.-all, resulting in $N$-binary SVMs. Finally, the $N$-binary SVM decisions are combined and transformed into $n$ binary values using the BCH decoding algorithm. The $n$ binary predicted labels are converted into original class labels using the hamming distance. The architecture of the proposed approach is depicted in Figure 6.1.

![Figure 6.1. Architecture of merging BCH coding with SVM](image-url)
The functioning of each sub module of the proposed approach is mentioned below.

➤ **Training Phase**

The training process of \(N\)-binary SVMs is as follows:

(i) Select appropriate values for \(n\) and \(d\) in \(k\)-class problem such that 
\[
\log_2 k + 2d + 1.
\]

(ii) Create \(n\)-bit unique binary code for each class such that hamming distance is minimum among the codes.

(iii) Assign a unique binary code \(C(k)\) to each training instance \(x\) of class \(k\), represented as \((x, C(k))\)

(iv) Create \(N\)-binary training sets \(S^i = \{ S_0^i, S_1^i \} \) \((i=1,2,...N)\) using the training set 
\(S_x = \{ x, C(k), k=1,2,...K \} \) \((K\) is the total number of instances) in such a way that
\[
\forall (x, C(k)) \in S_x, \text{ belong to the subset of } S_i^i \text{ if the } i^{th} \text{ bit of } C(k) \text{ is } 1, \text{ otherwise, to } S_0^i.
\]

(v) Binary SVM is trained on each \(S_i^i\) for learning the binary function \(f(x)\). The predicted value of \(y_i\) is 1 if \(x \in S_i^i\) and –1 if \(x \in S_0^i\).

➤ **Testing Phase**

The testing process of \(N\)-binary SVMs is as follows:

(i) A novel instance \(x\) is used to identify its type.

(ii) To classify \(x\), use each binary SVM and obtain the predicted result \(\hat{y}_i\) by
\[
\hat{k} = \arg \min_i [HD(k)]
\]

(iii) Combine the predicted results of \(N\)-binary SVMs as. \(\hat{C} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_n]\)

(iv) Calculate hamming distance between the predicted result \(\hat{C}\) and unique binary code \(C(k)\)
\[
HD(k) = \text{hamming Distance}(\hat{C}, C(k)) = \sum_{i=1}^{n} | \hat{y}_i - C(k) |, \text{ for } k = 1,2,...K
\]
Finally, categorize \( x \) as the \( k \) class, where \( x \) has minimum hamming distance; \( \hat{k} = \arg \min \{HD(k)\} \)

### 6.3. Experimental Results and Discussion

Besides, using two powerful feature extraction schemes for membrane protein sequence representation, we have also used different kernels based SVM as classifiers. SVM is the most prominent and stable classifier, because it does not suffer from curse of dimensionality and has no local minima. Despite, its high performance, an error correction code BCH is incorporated in order to further boost its learning capability. The behavior of the three kernels based SVM is examined in order to select the best kernel function, since the selection of the kernel function is a key problem during SVM investigation. The used kernel function of SVM can be represented as linear (lin-SVM), polynomial (Poly-SVM), and radial based function (RBF-SVM). The parameters of the SVM are optimized through grid searching using training data. The performance of the different kernels based SVM is assessed through various measures including accuracy, sensitivity, specificity, MCC, and F-measures.

#### 6.3.1. PSSM vs. Hybrid (PSSM+ SAAC)

In order to find a good discriminative feature extraction strategy, first, we have compared the performance of various kernels based SVM using individual PSSM and a hybrid of SAAC and PSSM. The success rates of SVMs using PSSM and hybrid model on dataset1 and dataset2 are listed in Table 6.2 and 6.4, respectively. The simulated results demonstrate that Lin, Poly, and RBF based SVMs have achieved better results using hybrid model compared to PSSM alone on both datasets. On the other hand, RBF-SVM has obtained the highest results 88.9, 90.3% on dataset1 and 91.8, and 92.5% on dataset2, respectively, using both feature extraction strategies. In addition, after incorporating BCH code with SVMs, the performance of each kernel is boosted in case of both feature extraction schemes for the both datasets. The predicted results of SVMs with BCH coding are reported in Table 6.2 and 6.4. Hybrid model has still obtained better results
than that of single PSSM whereas RBF-SVM has yielded the highest results compared to the rest of kernels.

Table 6.2. Prediction performance of SVMs with and without BCH coding on dataset1

<table>
<thead>
<tr>
<th>Methods</th>
<th>PSSM</th>
<th>Lin-SVM</th>
<th>Poly-SVM</th>
<th>RBF-SVM</th>
<th>Lin-SVM &amp; BCH</th>
<th>Poly-SVM &amp; BCH</th>
<th>RBF-SVM &amp; BCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>84.5</td>
<td>87.2</td>
<td>88.9</td>
<td>85.1</td>
<td>87.8</td>
<td>89.5</td>
<td>84.8</td>
</tr>
<tr>
<td>Type-I</td>
<td>80.7</td>
<td>85.5</td>
<td>77.9</td>
<td>86.4</td>
<td>86.4</td>
<td>78.6</td>
<td>78.4</td>
</tr>
<tr>
<td>Type-II</td>
<td>45.4</td>
<td>61.2</td>
<td>78.9</td>
<td>63.1</td>
<td>80.3</td>
<td>80.3</td>
<td>50.9</td>
</tr>
<tr>
<td>Multipass</td>
<td>93.0</td>
<td>93.7</td>
<td>97.1</td>
<td>94.0</td>
<td>97.4</td>
<td>95.6</td>
<td>96.1</td>
</tr>
<tr>
<td>Lipid</td>
<td>51.0</td>
<td>54.9</td>
<td>54.9</td>
<td>60.8</td>
<td>58.8</td>
<td>76.1</td>
<td>52.9</td>
</tr>
<tr>
<td>GPI</td>
<td>69.1</td>
<td>66.4</td>
<td>63.6</td>
<td>67.3</td>
<td>65.4</td>
<td>40.0</td>
<td>38.2</td>
</tr>
</tbody>
</table>

Table 6.3. Performance of SVMs with and without BCH coding on dataset1.

<table>
<thead>
<tr>
<th>Methods</th>
<th>SAAC</th>
<th>Lin-SVM</th>
<th>Poly-SVM</th>
<th>RBF-SVM</th>
<th>Lin-SVM &amp; BCH</th>
<th>Poly-SVM &amp; BCH</th>
<th>RBF-SVM &amp; BCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>84.23</td>
<td>88.20</td>
<td>86.37</td>
<td>84.67</td>
<td>88.93</td>
<td>86.91</td>
<td>83.43</td>
</tr>
<tr>
<td>Specificity</td>
<td>84.22</td>
<td>86.55</td>
<td>89.20</td>
<td>84.79</td>
<td>87.23</td>
<td>89.87</td>
<td>84.75</td>
</tr>
<tr>
<td>MCC</td>
<td>0.60</td>
<td>0.67</td>
<td>0.69</td>
<td>0.61</td>
<td>0.68</td>
<td>0.71</td>
<td>0.61</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.69</td>
<td>0.73</td>
<td>0.76</td>
<td>0.69</td>
<td>0.74</td>
<td>0.77</td>
<td>0.69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>SAAC+PSSM</th>
<th>Lin-SVM</th>
<th>Poly-SVM</th>
<th>RBF-SVM</th>
<th>Lin-SVM &amp; BCH</th>
<th>Poly-SVM &amp; BCH</th>
<th>RBF-SVM &amp; BCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>83.43</td>
<td>84.50</td>
<td>91.52</td>
<td>84.55</td>
<td>85.21</td>
<td>92.42</td>
<td>84.33</td>
</tr>
<tr>
<td>Specificity</td>
<td>84.75</td>
<td>88.11</td>
<td>89.71</td>
<td>84.84</td>
<td>88.80</td>
<td>90.54</td>
<td>84.75</td>
</tr>
<tr>
<td>MCC</td>
<td>0.61</td>
<td>0.66</td>
<td>0.74</td>
<td>0.63</td>
<td>0.68</td>
<td>0.76</td>
<td>0.69</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.69</td>
<td>0.73</td>
<td>0.79</td>
<td>0.71</td>
<td>0.75</td>
<td>0.81</td>
<td>0.73</td>
</tr>
</tbody>
</table>

Other performance parameters on both datasets such as sensitivity, specificity, MCC, and F-measures are shown in Table 6.3 and 6.5. The predicted outcomes of SVMs with and without BCH coding for individual membrane protein types are reported in Table 6.2 and 6.4 for both datasets.
In both cases, the accuracy of RBF-SVM for each type of membrane proteins, reported is better than that of Lin-SVM and Poly-SVM. Finally, we have realized that the discrimination power of the hybrid model is more strengthened than individual PSSM, because it has merged the discrimination power of SAAC and PSSM, which have enhanced the learning power of classifiers.

6.3.2. Hybrid (PSSM+SAAC)

In this section, a comparison has been expressed among Lin-SVM, Poly-SVM, and RBF-SVM using a hybrid model considered of with and without BCH coding.

6.3.2.1. Lin-SVM vs. Poly-SVM

Table 6.2 and 6.4 show the success rates of Lin-SVM and Poly-SVM along with and without BCH coding for both datasets. Lin-SVM has yielded an accuracy of 84.8 and 89.7%, whereas Poly-SVM has 87.7 and 90.4% accuracies for dataset1 and dataset2, respectively. Poly-SVM of degree 3 has yielded 2.82 and 0.7% higher accuracies than that of Lin-SVM on dataset1 and dataset2. In case of BCH coding, Lin-SVM has obtained 85.9 and 88.3% accuracies on dataset1 and dataset2, respectively. Similarly, Poly-SVM has achieved 90.5, and 91.0% accuracies on these two datasets, respectively. In case of dataset1, Lin-SVM has yielded uneven accuracies for individual membrane protein types. Type-I and multipass transmembrane have acquired dominant accuracies, whereas type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins have obtained poor accuracies. On the other hand, Poly-SVM has obtained relatively well results for each membrane protein type. After incorporating BCH coding with SVM, the performances of both the Lin-SVM and Poly-SVM have been improved for all types of membrane protein, but still have not achieved the desired results. For dataset2, Lin-SVM has yielded accuracies of 89.5, 45.0, 96.5, 65.4, 50, and 60% for type-I, type-II, multipass transmembrane, Lipid chain-anchored, GPI-anchored membrane, and peripheral membrane proteins, respectively. On the other hand, the performance of Poly-SVM is better than that of Lin-SVM for each type of membrane proteins using the same dataset. The improvement has also been reported in the performance of the Lin-SVM and
Poly-SVM after incorporating BCH coding for dataset2. Still the performance of Lin-SVM is lower than that of Poly-SVM.

6.3.2.2. Lin-SVM vs. RBF-SVM

The performance of Lin-SVM is also compared with that of RBF-SVM reported in Table 6.2 and 6.4. RBF-SVM has achieved accuracy of 90.3 and 92.6%, which is 5.44 and 2.9% higher than that of Lin-SVM on dataset1 and dataset2, respectively. In addition, after integrating BCH coding, the accuracies of RBF-SVM are 91.1 and 93.4%, which are still 5.2 and 2.4% higher than that of Lin-SVM, respectively, on dataset1 and dataset2. For each membrane protein type, RBF-SVM has obtained better accuracy. It has reduced the issue of bias somehow by treating all the types of membrane protein equally. Further, the accuracy for each membrane protein type is more boosted, after the addition of BCH coding with RBF-SVM, particularly, in case of type-I, type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins. The yielded accuracies of RBF-SVM for type-I transmembrane are 90.8%, type-II transmembrane 75.0%, multipass transmembrane 95.6%, lipid chain-anchored 70.6%, and GPI-anchored membrane proteins are 70.9%. The predicted results of RBF-SVM are 11.0, 21.1, 11.8, and 30.9% higher than that of Lin-SVM in case of type-I, type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins whereas 1.0% less in case of multipass transmembrane protein on dataset1. In contrast, the performance of RBF-SVM in both cases, with and without BCH coding is higher than that of Lin-SVM on dataset2. Besides BCH coding, the RBF-SVM has obtained better results than that of Lin-SVM with BCH coding. The success rates of lin-SVM are 91.1% for type-I transmembrane, 48.3% for type-II transmembrane, 96.6% for multipass transmembrane, 68.3% for lipid chain-anchored, 57.3% for GPI-anchored membrane, and 63.3% for peripheral membrane proteins. The predicted results of Lin-SVM are 4.8, 12.6, 0.8, and 5.8% less than the predicted results of RBF-SVM without BCH coding, while 2.9 and 3.3% higher in case of GPI-anchored and peripheral membrane proteins. On the other hand, in case of BCH coding, the highest success rates have been achieved by RBF-SVM for all types of membrane proteins except lipid chain-anchored membrane protein. The predicted accuracy for type-I transmembrane is 97.0%, for type-II transmembrane 63.6%, for
multipass transmembrane 97.7%, for lipid chain-anchored 60.3%, for GPI-anchored membrane 60.3%, and for peripheral membrane protein is 66.7%.

6.3.2.3. Poly-SVM vs. RBF-SVM

Both the kernels have yielded better results than Lin-SVM, which reveal that linear kernel is not adequate for the prediction of membrane protein types. Therefore, in this section, only the performances of Poly-SVM and RBF-SVM with and without BCH coding are analyzed to find out the appropriate kernel for membrane protein types prediction. The outcomes of both the kernels are reported in Table 6.2 and 6.4, respectively. RBF-SVM has achieved 2.77 and 2.62% higher accuracies than that of Poly-SVM with and without BCH coding, in case of dataset1. The performance of Poly-SVM is comparatively good for each type of membrane proteins. On the other hand, RBF-SVM has achieved outstanding results for type-I, type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins.

Table 6.4. Prediction performance of SVMs with and without BCH coding on dataset2

<table>
<thead>
<tr>
<th></th>
<th>Jackknife test</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Accuracy</td>
<td>Type-I</td>
<td>Type-II</td>
<td>Multipass</td>
</tr>
<tr>
<td>PSSM</td>
<td></td>
<td>Lin-SVM</td>
<td>87.6</td>
<td>87.1</td>
<td>43.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poly-SVM</td>
<td>88.5</td>
<td>88.7</td>
<td>45.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBF-SVM</td>
<td>91.8</td>
<td>93.5</td>
<td>60.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lin-SVM &amp; BCH</td>
<td>88.4</td>
<td>89.0</td>
<td>46.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poly-SVM &amp; BCH</td>
<td>89.3</td>
<td>90.0</td>
<td>48.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBF-SVM &amp; BCH</td>
<td>92.5</td>
<td>94.6</td>
<td>63.6</td>
</tr>
<tr>
<td>PSSM+SAAC</td>
<td></td>
<td>Lin-SVM</td>
<td>89.7</td>
<td>89.5</td>
<td>45.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poly-SVM</td>
<td>90.4</td>
<td>91.7</td>
<td>47.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBF-SVM</td>
<td>92.6</td>
<td>95.9</td>
<td>60.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lin-SVM &amp; BCH</td>
<td>90.5</td>
<td>91.1</td>
<td>48.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poly-SVM &amp; BCH</td>
<td>91.0</td>
<td>92.4</td>
<td>49.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBF-SVM &amp; BCH</td>
<td>93.4</td>
<td>97.0</td>
<td>63.6</td>
</tr>
</tbody>
</table>
Table 6.5. Performance of SVMs with and without BCH coding on dataset2.

<table>
<thead>
<tr>
<th></th>
<th>Methods</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SAAC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lin-SVM</td>
<td>86.34</td>
<td>87.60</td>
<td>0.65</td>
<td>0.70</td>
<td></td>
</tr>
<tr>
<td>Poly-SVM</td>
<td>87.27</td>
<td>88.51</td>
<td>0.67</td>
<td>0.72</td>
<td></td>
</tr>
<tr>
<td>RBF-SVM</td>
<td>91.61</td>
<td>91.73</td>
<td>0.75</td>
<td>0.79</td>
<td></td>
</tr>
<tr>
<td>Lin-SVM &amp; BCH</td>
<td>87.47</td>
<td>88.41</td>
<td>0.67</td>
<td>0.72</td>
<td></td>
</tr>
<tr>
<td>Poly-SVM &amp; BCH</td>
<td>88.28</td>
<td>89.34</td>
<td>0.69</td>
<td>0.74</td>
<td></td>
</tr>
<tr>
<td>RBF-SVM &amp; BCH</td>
<td>92.47</td>
<td>92.41</td>
<td>0.77</td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td><strong>SAAC+PSSM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lin-SVM</td>
<td>88.26</td>
<td>89.76</td>
<td>0.69</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td>Poly-SVM</td>
<td>89.42</td>
<td>90.43</td>
<td>0.71</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>RBF-SVM</td>
<td>92.81</td>
<td>92.54</td>
<td>0.77</td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td>Lin-SVM &amp; BCH</td>
<td>89.34</td>
<td>90.54</td>
<td>0.71</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>Poly-SVM &amp; BCH</td>
<td>90.06</td>
<td>91.01</td>
<td>0.73</td>
<td>0.77</td>
<td></td>
</tr>
<tr>
<td>RBF-SVM &amp; BCH</td>
<td>93.67</td>
<td>93.31</td>
<td>0.80</td>
<td>0.83</td>
<td></td>
</tr>
</tbody>
</table>

In case of dataset2, still RBF-SVM has achieved the highest results, which are 92.6% accuracy, 92.8% sensitivity, 92.5% specificity, 0.77 MCC, and 0.81 F-measure. The performance of RBF-SVM is more enhanced after incorporating BCH coding and is the highest among the three kernels based SVM. The predicted results of RBF-SVM are 93.4, 93.7, 93.3%, 0.80, and 0.83 for accuracy, sensitivity, specificity, MCC, and F-measure, respectively. So, the accuracy of RFB-SVM is 2.39 and 2.89% higher than that of Poly-SVM, respectively, on dataset1 and dataset2. RBF-SVM has yielded the highest accuracies for type-I, type-II transmembrane, lipid chain-anchored, and GPI-anchored membrane proteins. The experimental results reveal that during RBF-SVM classification, the minority classes have obtained good accuracies compared to Poly-SVM classification. Therefore, RBF is an adequate kernel for the prediction of membrane protein types.
7. Online web Predictor: Mem-PHybrid Predictor

7.1. Introduction

The previous chapters have presented useful discussion about membrane protein types as well as individual and ensemble classification algorithms. This chapter seeks to extend one-step further, from single-layer prediction to two-layer prediction. In the last few decades, huge amounts of annotated protein sequences are entered into protein data banks. The identification of these proteins by experimental ways is difficult, but sometimes even impossible, because there are only 1% of protein structures identified. Despite it, computer and internet have brought a great revolution in the world where everyone demands for reliable and fast outcomes. In this regards, it is highly desirable to develop an automated method, which has high quality, reliability, and fast speed. Therefore, in order to facilitate the scientific and student community, we have developed a two-layer predictor (Mem-PHybrid) for classification of membrane proteins. The earlier computational methods for membrane proteins are single layer, where only membrane and non-membrane or types of membrane protein are predicted. The main problem in membrane protein type predictor is that whenever a non-membrane protein sequence is provided for identification, the predictor classifies it as a membrane protein type, which is not really a membrane protein type. To avoid such problems, we have developed two-layer predictor for membrane proteins. In the first layer, it classifies the protein query as a membrane or a non-membrane protein. If the protein query is identified as a membrane protein then further, it predicts the type of membrane proteins. The web predictor is developed by looking at various parameters such as efficiency, speed, and reliability. The features are extracted from protein sequences through various feature extraction strategies including SAAC, composition and translation. The biochemical and physicochemical
features are fused by forming a hybrid model. The important characteristic of a hybrid model is that the deficiency of one method is compensated by the advantages of the other methods. To resolve the problems of curse of dimensionality, complexity, and computational cost, the irrelevant and redundant information is eradicated through mRMR technique. Three different kinds of classification algorithms are investigated to select the best one for the proposed predictor. Two statistical tests (jackknife and independent dataset test) are brought under consideration. Accuracy, sensitivity, specificity, MCC, F-measure, and ROC curve are utilized to assess the performance of the classification algorithms.

7.2. Materials and Methods

Datasets, and feature extraction schemes are presented in this section.

7.2.1. Datasets

In order to develop a two-layer classification model, we have utilized three benchmark datasets dataset1, dataset2, and dataset4. Dataset1 and dataset2 are membrane protein type datasets, while dataset4 is a non-membrane proteins dataset. Dataset1 comprises of only integral membrane protein types, which are distributed in five different classes whereas dataset2 contains five types of integral membrane proteins and the one peripheral membrane proteins. More details about dataset1 and dataset2 are provided in Chapter 4 and 5, respectively. In addition, both the datasets are internally divided in training and testing datasets. Dataset1 contains 2,059 membrane protein sequences in training dataset while 2,625 membrane protein sequences in testing dataset.

Similarly, dataset2 has 2,628 membrane protein sequences in training dataset whereas 3,160 membrane protein sequences in testing dataset. On the other hand, dataset4 contains 2,029 non-membrane proteins sequences. It is constructed by using 25% cutoff threshold and includes only those protein sequences, which have less than 25% sequence identity. Smaller cutoff threshold builds the dataset more strengthen by reducing redundancy and homology bias.
7.2.2. Feature Extraction Schemes

In this section, physicochemical properties of amino acids are used as feature extraction strategy. Some discussion about physicochemical properties are provided below.

7.2.2.1. Physicochemical Properties

Primary structure of a protein is formed by the polymer of twenty amino acids; each amino acid performs functions according to the behavior of its side chain. Each amino acid plays a vital role in the formation of proteins. The identification of protein behavior through discrete methods is difficult, because AA and dipeptide composition may lead to the bias between two adjacent amino acids. Therefore, we have utilized physicochemical properties of amino acids for protein sequence representation. The significance of physicochemical properties is that they represent several regions of a protein sequence and use a set of average values for discrimination. Physicochemical properties are thus useful for many applications, particularly, for the study of conservation of functionally imperative residues in a protein family.

The physicochemical properties of amino acids are expressed by composition and translation. Composition and translation features determine the distributed patterns of amino acids for a particular structure or physicochemical properties of proteins [134; 135]. To extract the essential information from protein sequences, we have used seven main physicochemical properties of amino acids including hydrophobicity, normalized Van der Waals volume, polarity, polarization, charge, secondary structure, and solvent accessibility. Each physicochemical property corresponds to three values, for instance, hydrophobicity might be polar, neutral, or hydrophobic, for more details see Chapter 3. After applying both composition and translation, 42-D descriptor values are extracted against each protein sequence, 21-D each.

7.3.3. Hybrid Model

The large explosion of protein sequences in protein databanks has increased the significance of computational models. Before developing computational models, it is necessary to select a proper feature extraction strategy, which can explore all the required information and further help the classification algorithm during the identification of
protein. Despite, choosing best feature extraction strategy, sometimes an individual feature extraction strategy has no ability to accurately predict a protein sequence due to the lack of essential information [192]. In that situation the concept of hybrid is used, which has the ability to enhance the discrimination power of classifiers. Therefore, we have integrated the SAAC and physicochemical properties to utilize the discrimination power of both the methods for the prediction of membrane protein types. The advantage of hybrid model is that it exploits different kinds of information. The dimension of the hybrid model is 102-D (60+42). Looking at the phenomenon of curse of dimensionality and computational cost; a most powerful feature selection technique mRMR is applied to reduce the dimensionality of the feature space. After applying mRMR, the dimension of the hybrid model is reduced to 80-D.

7.3. Web Server (Mem-PHybrid Predictor)

Unfortunately, protein sequences are exceeding in number rapidly, hence, difficult to be identified through experimental approaches. In order to facilitate scientific and student community, we have developed a two-layer user-friendly web predictor (Mem-PHybrid Predictor) for the classification of membrane proteins. In the first layer, it identifies the protein query as a membrane or a non-membrane protein. If the protein query is classified as a membrane protein then further, it determines the type of membrane protein. A procedure that elaborates the use of our web server is as follows:

Step1.
Open the web page at http://111.68.99.218/mem-PHybrid and the top page of the Mem-PHybrid predictor will be displayed on your computer screen as illustrated in (Figure 7.1a).

Step2.
An “Example” link demonstrates the format of a query sequence. Datasets of membrane and non-membrane proteins are attached under the “Data” link. Either paste or type a protein query into a text box provided in the center of the page. The Mem-PHybrid Predictor accepts only simple text format where minimum length of the protein query is 50 residues. After typing or pasting a protein query as shown in (Figure 7.1b), click on the “Submit” button to see the predicted output.
Mem-PHybrid Predictor

Prediction System for Membrane Protein Types

Please Enter Protein Sequence:  Example  Data

Submit  Clear

The Method of membrane protein types prediction provided in this website is based on the SVM incorporated with Hybrid features.

maqrood.hayat@gmail.com, asi@pieas.edu.pk
DCIS, Pakistan Institute of Engineering & Applied Science, Nishtar, Islamabad, Pakistan.

(a)

(b)
Figure 7.1. Screenshot (a) Display illustrates the Main page of Mem-PHybrid predictor (b) Display showing the input query of protein sequence (c) Display depicting the input protein query as a membrane or non-membrane (d) Display showing the predicted type of membrane protein.
Step 3.
After clicking on the Submit button, the Mem-PHybrid Predictor starts preprocessing before executing the results. It first confirms the length of the protein query. Then, it consequently verifies it for illegal characters. After verifying the constraints, the Mem-PHybrid Predictor first determines the query as a membrane or a non-membrane protein, as illustrated in Figure 7.1c. If the query is a membrane protein then, it further identifies the type of membrane protein, as illustrated in (7.1d).

7.4. Experimental Results and Discussion

In this study, the performance of the classifiers is evaluated through two statistical cross validation tests such as jackknife and independent dataset tests. Accuracy, sensitivity, specificity, MCC, F-measure, and ROC curve are used as performance measures.

7.4.1. Jackknife Test

The success rate of the classifiers using jackknife test is shown as follows

7.4.1.1. Membrane vs. Non-membrane Proteins

Mem-PHybrid Predictor is a two-layer predictor, where the first layer differentiates between a membrane and a non-membrane protein. The second layer totally depends on the prediction of the first layer. So, the accurate prediction of the first layer is indispensable. For this purpose, it is very necessary to select the best classification algorithm for the development of the first layer of the predictor. In this regard, we have evaluated the discrimination power of the three classification algorithms including RF, ET-KNN, and SVM in conjunction with the hybrid model for the prediction of membrane and non-membrane proteins. The success rates of SVM, ET-KNN, and RF using both the datasets are reported in Table 7.1. In case of dataset1, the overall accuracy of all the three classifiers is more or less similar. On the other hand, the accuracies of RF for membrane and non-membrane proteins have large differences, while the other two classifiers have comparable results. While using dataset2, the maximum accuracy is obtained by SVM while the minimum by ET-KNN. The overall accuracy of SVM is about 0.87 and 1.34% higher than that of the RF and ET-KNN, respectively. However, we need a classifier,
which has promising results for both membrane and non-membrane proteins. Therefore, we have selected SVM for the first layer of the predictor, which has presented outstanding results on both the datasets. The performance comparison of classifiers is presented in Figure 7.2.

Table 7.1. Performance of classifiers for membrane and non-membrane protein using Jackknife test on dataset1 and dataset2.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Dataset1</th>
<th></th>
<th>Dataset2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Membrane</td>
<td>Non-Membrane</td>
<td>Accuracy</td>
</tr>
<tr>
<td>SVM</td>
<td>89.40</td>
<td>89.40</td>
<td>89.35</td>
<td>95.17</td>
</tr>
<tr>
<td>ET-KNN</td>
<td>89.05</td>
<td>88.40</td>
<td>89.70</td>
<td>93.83</td>
</tr>
<tr>
<td>RF</td>
<td>89.43</td>
<td>86.88</td>
<td>92.01</td>
<td>94.30</td>
</tr>
</tbody>
</table>

Figure 7.2. Performance of various classifiers for membrane and non-membrane proteins using Jackknife test on dataset1 and dataset2.

7.4.1.2. Membrane Protein Types

The second layer of the predictor classifies the type of membrane proteins. For this layer to select better classifier among the three, we have investigated the performance of all the three classifiers. The predicted outcomes of ET-KNN, RF, and SVM are reported in Table 7.2. Among the three classifiers, SVM has yielded the highest overall success rates. SVM has obtained the accuracy of 89.6%, sensitivity of 90.7%, specificity of 89.1%,
MCC of 0.72, and F-measure of 0.78 on the parameters of $C=4$ and $\gamma=0.00025$. On the other hand, the obtained accuracies of RF and ET-KNN are 86.9 and 86.0%, respectively. RF has used 500 iterations and 200 trees. The ROC curve is drawn among the classifiers to show the performances on different thresholds depicted in Figure 7.3. The predicted outcome of the classifier for each membrane protein type is reported in Table 7.3. The performance of SVM for each membrane protein type is higher compared to that of the rest of classifiers. The highest accuracy is yielded for type-I and multipass transmembrane proteins, which are 89.9 and 95.3%, respectively. However, it has yielded for type-II transmembrane 63.8%, lipid chain-anchored 58.8%, and GPI-anchored membrane proteins 71.8% accuracies. The performance of ET-KNN is also good in case of individual membrane protein types, which is 80.1% for type-I, 63.1% for type-II, 93.6% for multipass transmembrane, 54.9% for lipid chain-anchored, and 65.4% for GPI-anchored membrane proteins. In contrast, the performance of RF is worse except for multipass and type-I transmembrane proteins.

Table 7.2. Success rates of classifiers using Jackknife test on dataset1 and dataset2

<table>
<thead>
<tr>
<th></th>
<th>Dataset1</th>
<th></th>
<th>Dataset2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Methods</td>
<td>Accuracy</td>
<td>Sensitivity</td>
</tr>
<tr>
<td></td>
<td>ET-KNN</td>
<td>86.0</td>
<td>85.7</td>
</tr>
<tr>
<td></td>
<td>RF</td>
<td>86.9</td>
<td>86.0</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>89.6</td>
<td>90.7</td>
</tr>
</tbody>
</table>

In addition, we have evaluated the performance of the classifiers by using another dataset. All the obtained outcomes of ET-KNN, RF, and SVM are listed in Table 7.2. The overall accuracy of all the three classifiers is comparatively good. Again, SVM achieves better results compared to ET-KNN and RF. The obtained results of SVM with the parameters of $C=15$ and $\gamma=0.00025$ are 91.5, 91.4, 91.4%, 0.74, and 0.79 of accuracy, specificity, specificity, MCC, and F-measure, respectively. RF has achieved an accuracy of 89.6% by using 500 iterations and 150 trees. The comparison among SVM, RF, and ET-KNN on dataset2 is illustrated in Figure 7.4 through ROC curve.
The predicted accuracy of the classifiers for the each membrane protein type is shown in Table 7.3. Again the performance of SVM is prominent for each type of membrane proteins; it yields better results than that of other classifiers. The performance of ET-KNN is also well except for GPI-anchored membrane proteins.

Table 7.3. Success rates of classifiers for each type of membrane proteins using Jackknife test on both datasets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Type I</th>
<th>Type II</th>
<th>Multipass</th>
<th>Lipid</th>
<th>GPI</th>
<th>Peripheral</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ET-KNN</td>
<td>80.0</td>
<td>63.1</td>
<td>93.6</td>
<td>54.9</td>
<td>65.4</td>
<td>Nil</td>
</tr>
<tr>
<td>RF</td>
<td>84.6</td>
<td>43.4</td>
<td>96.2</td>
<td>51.0</td>
<td>62.7</td>
<td>Nil</td>
</tr>
<tr>
<td>SVM</td>
<td>89.9</td>
<td>63.8</td>
<td>95.3</td>
<td>58.8</td>
<td>71.8</td>
<td>Nil</td>
</tr>
<tr>
<td>Dataset2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ET-KNN</td>
<td>91.6</td>
<td>60.3</td>
<td>97.2</td>
<td>75.0</td>
<td>39.7</td>
<td>30.3</td>
</tr>
<tr>
<td>RF</td>
<td>89.0</td>
<td>46.3</td>
<td>97.6</td>
<td>72.1</td>
<td>32.4</td>
<td>20.5</td>
</tr>
<tr>
<td>SVM</td>
<td>93.0</td>
<td>60.2</td>
<td>97.3</td>
<td>73.1</td>
<td>55.9</td>
<td>30.3</td>
</tr>
</tbody>
</table>

In contrast, RF yields better accuracy merely for type-I, multipass transmembrane, and lipid chain-anchored membrane proteins. Due to the uneven nature of the datasets, the performance of the classifiers is affected for minority classes. Overall, on both datasets, SVM performs well compared to RF and ET-KNN.
7.4.2. Independent Dataset Test

The predicted outcomes of the classifiers using hybrid model are listed in Table 7.4. The predicted overall accuracies of ET-KNN and SVM are almost similar. SVM has yielded 97.3% accuracy with the parameters of $C=20$ and $\gamma=0.00040$, while ET-KNN has achieved 97.1% accuracy. In contrast, RF has obtained 95.0% accuracy using 300 iterations and 100 trees. Additionally, the results of SVM, ET-KNN, and RF for each membrane protein type are shown in Table 7.5. SVM yields above 92% accuracy for each membrane protein type. ET-KNN achieves well accuracy for type-I, type-II, and multipass transmembrane while worse for lipid chain-anchored and GPI-anchored membrane proteins. On the other hand, the predicted accuracy of RF is better only for type-I and multipass transmembrane whereas poor for the rest of classes.

Furthermore, the classification algorithms are investigated through dataset2. The simulated results are reported in Table 7.4. In this case, the highest accuracy of 95.5% is obtained by SVM, which is 1.3 and 2.2% higher than that of ET-KNN and RF, respectively. SVM has used $C=4$ and $\gamma=0.00025$ whereas RF has used 300 iterations and 200 trees. Other performance measures of SVM are 98.0% sensitivity, 95.7% specificity, 0.84 MCC, and 0.88 F-measure. In Table 7.5 shows the predicted accuracy of the classifiers for each membrane protein type. Among the classification results, ET-KNN
has yielded relatively well accuracies for all the types of membrane proteins, while RF has obtained good accuracies only for type-I and multipass transmembrane whereas worse for the rest of membrane protein types.

Table 7.4. Success rates of classifiers using Independent dataset test on both datasets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET-KNN</td>
<td>97.1</td>
<td>98.2</td>
<td>96.9</td>
<td>0.91</td>
<td>0.92</td>
</tr>
<tr>
<td>RF</td>
<td>95.0</td>
<td>94.6</td>
<td>94.8</td>
<td>0.85</td>
<td>0.88</td>
</tr>
<tr>
<td>SVM</td>
<td>97.3</td>
<td>97.7</td>
<td>97.0</td>
<td>0.91</td>
<td>0.93</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>MCC</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET-KNN</td>
<td>94.3</td>
<td>91.1</td>
<td>94.3</td>
<td>0.82</td>
<td>0.83</td>
</tr>
<tr>
<td>RF</td>
<td>93.4</td>
<td>92.6</td>
<td>93.7</td>
<td>0.79</td>
<td>0.82</td>
</tr>
<tr>
<td>SVM</td>
<td>95.5</td>
<td>98.0</td>
<td>95.7</td>
<td>0.84</td>
<td>0.88</td>
</tr>
</tbody>
</table>

Table 7.5. Success rates of classifiers for each type of membrane proteins using Independent dataset test on both datasets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Type I</th>
<th>Type II</th>
<th>Multipass</th>
<th>Lipid</th>
<th>GPI</th>
<th>Peripheral</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET-KNN</td>
<td>98.3</td>
<td>93.9</td>
<td>98.6</td>
<td>64.3</td>
<td>77.9</td>
<td>Nil</td>
</tr>
<tr>
<td>RF</td>
<td>92.9</td>
<td>78.3</td>
<td>98.5</td>
<td>71.4</td>
<td>68.6</td>
<td>Nil</td>
</tr>
<tr>
<td>SVM</td>
<td>96.2</td>
<td>93.9</td>
<td>98.0</td>
<td>92.8</td>
<td>94.2</td>
<td>Nil</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>Type I</th>
<th>Type II</th>
<th>Multipass</th>
<th>Lipid</th>
<th>GPI</th>
<th>Peripheral</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET-KNN</td>
<td>82.7</td>
<td>89.6</td>
<td>98.0</td>
<td>73.1</td>
<td>77.1</td>
<td>50.0</td>
</tr>
<tr>
<td>RF</td>
<td>90.9</td>
<td>57.6</td>
<td>98.1</td>
<td>52.2</td>
<td>68.7</td>
<td>50.0</td>
</tr>
<tr>
<td>SVM</td>
<td>85.9</td>
<td>90.3</td>
<td>99.0</td>
<td>73.1</td>
<td>78.3</td>
<td>50.0</td>
</tr>
</tbody>
</table>

In contrast, SVM has achieved better result for each membrane protein type compared to ET-KNN and RF. For peripheral membrane protein, all the three classifiers have yielded the same results. It is anticipated that overall, in both statistical tests type-I and multipass transmembrane proteins have achieved the highest accuracies compared to the rest of types. This might be due to the maximum strength of both classes in the datasets. After the empirical analysis, finally, we have realized that SVM has more discriminative power and it might be conceived as useful tool for the prediction of membrane protein types.
8. RF-TMH: Prediction of Transmembrane Helix Using Hybrid Features

8.1. Introduction

In the first phase, the types of membrane proteins are targeted. This chapter starts with the second phase of this research that elaborates the proposed approach for the prediction of membrane protein structures.

Due to the vital role of transmembrane helix in living organisms, the prediction of transmembrane helix is indispensible. Transmembrane helix and topology provide imperative information regarding the function and structure of membrane proteins, despite, there are only 1% membrane protein structures identified. However, the prediction of transmembrane helix and topology becomes a focal problem in Computational Biology and Bioinformatics because of experimental intricacies and lack of structures. In the last few years, the location and orientation of transmembrane helices are identified using amino acid sequence information. In this regard, we have developed RF-TMH model to enhance true prediction of transmembrane helices. In this model, the protein sequences are represented by two feature extraction schemes of different nature including compositional index and physicochemical properties. Sometimes the feature space contains redundant information. So, suffering the classification algorithm performance, difficulty in protein motif discerning, and curse of dimensionality are the implications of redundant features. In order to thwart the model from redundant and irrelevant features as well as boost its learning power, we have applied SVD to eliminate such features. The selected features of both the feature extraction schemes are fused and provided to the classifier as input space. Weighted RF is used as a classification algorithm, because it has the ability of ensemble classification. In addition, it tries to handle the problem of bias by assigning different weight to each class. The performance
of the RF-TMH is assessed by applying 10-fold cross validation at various levels such as per protein, per segment, and per residue using two benchmark datasets.

8.2. Materials and Methods

In this section, datasets, feature extraction and feature selection technique, and RF-TMH system are presented.

8.2.1. Datasets

In this work, we have utilized two benchmark datasets. Both the datasets are passed through all the processes that are prerequisite for any standard dataset. The first benchmark dataset is low-resolution membrane protein dataset, which was constructed by Moller et al. [81]. It was annotated from SWISS-PROT release 49.0 [87] and consequently, redundancy and homology biases were removed. Initially, it consists of 145 membrane protein sequences, but two protein sequences have no annotation with membrane proteins, therefore, these two sequences were excluded from the dataset. As a result, the low-resolution dataset contains 143 membrane protein sequences that have 687 transmembrane helix segments. The second dataset is high-resolution membrane proteins dataset. It was constructed by merging two datasets of 3-D structure helix. It consists of 258 membrane protein sequences, of which 101 protein sequences are picked out from MPtopo database [88] and 231 protein sequences are selected from TMPDB database [89]. In order to remove the redundancy, 30% CD-Hit has been applied. The high-resolution dataset contains both single and multispansing transmembrane protein sequences that consists of 1,232 transmembrane helix segments.

8.2.2. Feature Extraction schemes

This section presents two different feature extraction schemes for generating relevant and valuable information from protein sequences.

8.2.2.1. Physicochemical properties

The primary structure of protein is a polymer of amino acids, where each amino acid plays distinct role in the formation of protein pattern. According to their behavior, amino
acids are categorized into different groups, which refer to physicochemical properties. The function and structure of proteins are identified by the roles of these physicochemical properties. Therefore, we have applied the prime physicochemical properties of amino acids namely, charge, polarity, aromaticity, size, and electronic for protein sequences representation in order to extract the vital information. Each physicochemical property is further categorized into sub-types based on its behavior as presented in Table 8.1.

In order to extract the information from protein sequences, first, we have created five replicas of each protein sequence, where one replica against one physicochemical property. Then each residue of replica is substituted by its corresponding property. For instance residue $r_i$ at position $i$ can be indicated as:

$$r_i = (C_i, P_i, A_i, S_i, E_i)$$

where $C_i$, $P_i$, $A_i$, $S_i$, and $E_i$ denote charge, polarity, aromaticity, size, and electronic, respectively.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Amino acids</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charge</td>
<td>Positive: H, K, R</td>
</tr>
<tr>
<td></td>
<td>Negative: D, E</td>
</tr>
<tr>
<td>Polarity</td>
<td>Polar: C, D, E, H, K, N, Q, R, S, T, Y</td>
</tr>
<tr>
<td>Aromaticity</td>
<td>Aliphatic: I, L, V</td>
</tr>
<tr>
<td></td>
<td>Aromatic: F, H, W, Y</td>
</tr>
<tr>
<td></td>
<td>Neutral: A, C, D, E, G, K, M, N, P, Q, R, S, T</td>
</tr>
<tr>
<td>Size</td>
<td>Small: A, G, P, S</td>
</tr>
<tr>
<td></td>
<td>Medium: D, N, T</td>
</tr>
<tr>
<td></td>
<td>Large: C, E, F, H, I, K, L, M, Q, R, V, W, Y</td>
</tr>
<tr>
<td>Electronic</td>
<td>Strong donor: A, D, E, P</td>
</tr>
<tr>
<td></td>
<td>Weak donor: I, L, V</td>
</tr>
<tr>
<td></td>
<td>Neutral: C, G, H, S, W</td>
</tr>
<tr>
<td></td>
<td>Strong acceptor: F, M, Q, T, Y</td>
</tr>
<tr>
<td></td>
<td>Weak acceptor: K, N, R</td>
</tr>
</tbody>
</table>

After substituting each residue with its corresponding value, the sliding window technique is applied to compute the relative frequency of each physicochemical property. It computes the features only at one position (one residue at a time) after that moves the sliding window to the next position. This process is reiterated up to the last residue of the
protein sequence. Consequently, 16 features are extracted against each position. The feature space can be formulated as:

\[ R_i = [C_{ij}]^{1*16} \]  

(8.2)

where \( C_{ij} \) is the relative frequency of property \( j \) in window \( i \). Finally, the acquired feature matrix is represented as:

\[ P = [R_1^T R_2^T \ldots R_{L-L+1}^T]_{16*L-L+1} \]  

(8.3)

where \( T \) is transpose, \( L \) is the length of a protein sequence and \( l \) is the size of window.

### 8.2.2.2. Compositional Index

In order to calculate the compositional index of each amino acid, transmembrane helix segments and non-transmembrane helix segments are separated from each protein sequence. Transmembrane helix segments are stored in \( T_1 \) dataset whereas non-transmembrane helix segments are placed in \( T_2 \) dataset. Next, the relative frequency of each amino acid is computed separately in \( T_1 \) and \( T_2 \). The computational index \( p_i \) for amino acid \( i \) is computed as:

\[ p_i = -\ln \left( \frac{f_{i \text{non-TM}}}{f_{i \text{TM}}} \right) \]  

(8.4)

where \( f_{i \text{non-TM}} \) represents the frequency of amino acid \( i \) in dataset \( T_2 \) and \( f_{i \text{TM}} \) indicates the frequency of amino acid \( i \) in dataset \( T_1 \). The negative value of \( p_i \) reveals that amino acid \( i \) has more occurrence in transmembrane helix segments compared to non-transmembrane helix segments. After computing compositional index of amino acids, each residue in protein sequence is substituted by its corresponding index value. Further, window slide is applied of different sizes. The compositional index for a protein sequence \( p \), with window size \( w \) can be calculated as:

\[
\begin{align*}
    m^*_j = & \begin{cases} 
        \frac{\sum_{i=0}^{j-(w-1)/2} p_i}{j + ((w-1)/2)} & 1 \leq j \leq (w-1)/2 \\
        \frac{\sum_{i=j-(w-1)/2}^{j+(w-1)/2} p_i}{\sum_{i=j-(w-1)/2}^{j+(w-1)/2} p_i} & (w-1)/2 < j \leq L - (w-1)/2 \\
        \frac{\sum_{i=L-j-(w-1)/2}^{L-1} p_i}{L - j + 1 + ((w-1)/2)} & L - ((w-1)/2) < j \leq L 
    \end{cases}
\end{align*}
\]  

(8.5)
We have selected only odd window size of \((w=7\text{ to } 25)\). Consequently, the obtained feature space is 10-D.

### 8.2.3. Singular Value Decomposition

SVD is a computational technique used for dimensionality reduction that plays a key role in many multivariate data analyses. SVD performs well in such situations where there is a high correlation among the data points. SVD transforms the correlated data into uncorrelated data. It is based on eigenvalues, because the eigenvalues exhibit variation in data points. The eigenvalues are sorted in descending order where the first data point has the highest variance; the second data point has second highest variance and so on. Details can be found in Chapter 3. In this work, we have selected the first five top ranked dimensions, where 83% variance is found among these dimensions.

### 8.2.4. RF-TMH Proposed System

In this work, we develop a promising model RF-TMH for the prediction of transmembrane helices. In RF-TMH model, the protein sequences are expressed by two feature extraction schemes namely, physicochemical properties and compositional index. Both methods explore the information from a protein sequence in different ways as depicted in Figure 8.1. In case of physicochemical properties, first each residue of protein sequence is substituted by the corresponding value of amino acid in that physicochemical property. Next, the occurrence frequency of each sub physicochemical property in the specified peptide is calculated. This procedure is repeated to the last residue of the protein sequence where each position in the protein sequence is described by 16 features. On the other hand, in case of compositional index, first, the transmembrane and non-transmembrane segments are separated from a protein sequence. Next, the relative frequency of each amino acid in transmembrane and non-transmembrane datasets is computed. Then compositional index of each amino acid is calculated. Finally, each residue in the protein sequence is replaced by its corresponding compositional index. Consequently, 10 features are extracted against each position by taking odd number of window size from 7 to 25. In order to exterminate the redundant and extraneous features
Figure 8.1. Framework of the proposed approach

from the feature space, we have employed SVD separately on the feature space of each feature extraction scheme. Finally, five features are excerpted from each feature space, which has high variation. To strengthen the learning power of the classifier, we have fused the selected features of both the feature extraction schemes. Furthermore, Weighted RF is adopted as a classifier, because it has the ability of ensemble classifier. The significance of ensemble classifier is that it makes the decision on majority voting where the probability of error is minimum compared to individual classifiers. On the other hand, transmembrane helix segments are less in strength compared to non-transmembrane segments. It is noticed that in such cases the prediction of classifier often favors the majority class. The aim of the proposed approach is to predict transmembrane helix segments more accurately. Therefore, weight is assigned to each class to control the bias,
where high weight is assigned to the minority class and low weight is assigned to the majority class.

8.3. Performance Measures

The performance of the RF-TMH model is measured in three ways: per protein, per segment, and per residue basis. However, per protein and per segment only evaluate helix residues while per residue evaluates both helix and non-helix residues.

\[
Q_{\text{obsd}}^{\text{htm}} = \frac{\text{number of correctly predicted TM in dataset}}{\text{number of TM observed in dataset}} \times 100 \tag{8.6}
\]

\(Q_{\text{obsd}}^{\text{htm}}\) represents the recall of transmembrane helix (TM). It shows the percentage of transmembrane helix segments correctly predicted in the entire pool of transmembrane helix segments in the dataset.

\[
Q_{\text{pred}}^{\text{htm}} = \frac{\text{number of correctly predicted TM in dataset}}{\text{number of TM predicted in dataset}} \times 100 \tag{8.7}
\]

where \(Q_{\text{pred}}^{\text{htm}}\) indicates the precision of transmembrane helix. It represents the percentage of transmembrane helix segments correctly predicted in the predicted transmembrane helix segments.

\[
Q_{\text{all}} = \frac{\sum_{i=1}^{N_{\text{prot}}} \delta_i}{N_{\text{prot}}} \times 100 \quad \delta_i = \begin{cases} 1, & \text{if } Q_{\text{obsd}}^{\text{htm}} \land Q_{\text{pred}}^{\text{htm}} = 100 \text{ for protein } i \\ 0, & \text{otherwise} \end{cases} \tag{8.8}
\]

\(Q_{\text{all}}\) represents the number of protein sequences in which all its transmembrane helix segments are correctly predicted.

\[
Q_2 = \frac{\sum_{i=1}^{N_{\text{prot}}} \text{number of residues predicted correctly in protein } i}{\text{number of residues in protein } i} \times 100 \tag{8.9}
\]

where \(Q_2\) determines the percentage of correctly predicted residues in both transmembrane helix and non-transmembrane helix segments. It is also called accuracy.

\[
Q_{2T}^{\text{obsd}} = \frac{\text{number of residues correctly predicted in TM helices}}{\text{number of residues observed in TM helices}} \times 100 \tag{8.10}
\]

where \(Q_{2T}^{\text{obsd}}\) measures that how many residues are correctly predicted in the observed residues.
\[ Q_{2T}^{\text{prd}} = \frac{\text{number of residues correctly predicted in TM helices}}{\text{number of residues predicted in TM helices}} \times 100 \]  

(8.11)

where \( Q_{2T}^{\text{prd}} \) measures the number of residues that are correctly predicted in the predicted residues.

where TP represents the number of correctly predicted helix residues, FP is the number of incorrectly predicted helix residues, TN is the number of correctly predicted non-helix residues, and FN is the number of incorrectly predicted non-helix residues.

8.4. Experimental Results and Discussion

Mostly, investigators have used jackknife test due to its special characteristics. However, the primary issue in jackknife test is computational cost. In order to reduce the computational cost along with utilizing the special attributes of jackknife test, we have applied 10 fold cross validation. The performance of the weighted RF is evaluated in terms of per protein, per segment, and per residue levels.

8.4.1. Performance analysis at Protein level

The success rates of RF-TMH as well as the already published methods at protein level are presented in Table 8.2. The performance of the RF-TMH at per protein level is evaluated as: the number of protein sequences, where all the transmembrane helix segments are correctly predicted divided by the total number of protein sequences. In case of low-resolution dataset, the accuracy of RF-TMH is 76.92%, which indicates that there are 110 out of 143 protein sequences whose all transmembrane helix segments are correctly predicted. In literature, the highest accuracy has been achieved by the method of Arai et al. which is 74.83 using the same dataset [193]. The second highest accuracy of 73.29% has been obtained by SVMtop [194]. Apart from these, other methods such as MEMSAT3, TMHMM2, HMMTOP2, PHDhtm v.1.96, Top-Pred2, SOSUI 1.1, SPLIT4, Phobius, and PolyPhobius are introduced for the prediction of transmembrane helices. So, RF-TMH has achieved the highest accuracy compared to the already published methods. The performance of RF-TMH is 2.09% higher than that of ConPred-II and 3.63% higher than that of SVMtop. In case of high-resolution dataset the predicted result of RF-TMH is 74% indicating that there are 191 protein sequences whose all transmembrane helix
segments are correctly predicted. In contrast, SVMtop has achieved the highest result of 72.09% [194] and ConPred-II has obtained 69.14% [193]. The predicted results of RF-TMH are 1.91 and 4.86% higher than that of SVMtop and ConPred-II, respectively and more advanced than the other published methods.

8.4.2. Performance analysis at Segment level

The performance of RF-TMH at per segment level is assessed by two well known measures namely, recall and precision. Recall determines the percentage of correctly predicted transmembrane helix segments in total transmembrane helix segments whereas precision computes the percentage of correctly predicted transmembrane helix segments in total predicted transmembrane helix segments. Column 3-4 of Table 8.2 shows the precision ($Q^{\text{Sprd}}$) and recall ($Q^{\text{Sobsd}}$) of RF-TMH and other state of the art methods. Like protein level, the performance of RF-TMH is also prominent at per segment level than that of the existing methods. Using low-resolution dataset, the RF-TMH has yielded 96.06 and 95.10% of recall and precision. In contrast, the recall and precision of SVMtop are 94.76 and 93.94%, respectively [194]. In addition, some methods have good recall but bad precision, while some have bad recall and good precision. However, overall the performance of ConPred-II is better compared to other methods in both recall and precision, which are 94.76 and 92.21%, respectively [193]. So, RF-TMH has obtained 1.84% higher precision than that of SVMtop and ConPred-II, while 1.16 and 2.89% higher recall than those of SVMtop and ConPred-II. The performance of RF-TMH is also better using the high-resolution dataset. It has yielded recall and precision of 93.26 and 95.45%, respectively. On the other hand, the recall and precision of SVMtop are 92.78 and 94.46%, respectively. The performance of the rest of existing methods is more or less similar on low-resolution dataset, where some methods have well recall; in contrast, some methods have good precision.
Table 8.2. Performance comparison of proposed method with the existing methods

<table>
<thead>
<tr>
<th></th>
<th>Per Protein (%)</th>
<th>Per segment (%)</th>
<th>Per residue (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Q_{obs}</td>
<td>Q^{pred}</td>
<td>Q_{obs}</td>
</tr>
<tr>
<td><strong>Low resolution</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF-TMH</td>
<td>76.92</td>
<td>96.06</td>
<td>95.10</td>
</tr>
<tr>
<td>SVMtop</td>
<td>73.29</td>
<td>94.76</td>
<td>93.94</td>
</tr>
<tr>
<td>TMHMM2</td>
<td>68.53</td>
<td>90.39</td>
<td>93.52</td>
</tr>
<tr>
<td>HMMTOP2</td>
<td>64.34</td>
<td>89.96</td>
<td>93.78</td>
</tr>
<tr>
<td>PHDhtm v.1.96</td>
<td>39.86</td>
<td>76.27</td>
<td>85.76</td>
</tr>
<tr>
<td>MEMSAT3</td>
<td>70.63</td>
<td>91.56</td>
<td>90.24</td>
</tr>
<tr>
<td>TopPred2</td>
<td>57.34</td>
<td>86.75</td>
<td>91.13</td>
</tr>
<tr>
<td>SOSUI 1.1</td>
<td>63.64</td>
<td>88.36</td>
<td>91.55</td>
</tr>
<tr>
<td>SPLIT4</td>
<td>72.73</td>
<td>93.45</td>
<td>91.32</td>
</tr>
<tr>
<td>ConPred-II</td>
<td>74.83</td>
<td>94.76</td>
<td>92.21</td>
</tr>
<tr>
<td>Phobius</td>
<td>72.03</td>
<td>92.87</td>
<td>93.14</td>
</tr>
<tr>
<td>PolyPhobius</td>
<td>71.33</td>
<td>94.47</td>
<td>91.54</td>
</tr>
<tr>
<td><strong>High resolution</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF-TMH</td>
<td>74.00</td>
<td>93.26</td>
<td>95.45</td>
</tr>
<tr>
<td>SVMtop</td>
<td>72.09</td>
<td>92.78</td>
<td>94.46</td>
</tr>
<tr>
<td>TMHMM2</td>
<td>59.30</td>
<td>86.93</td>
<td>93.78</td>
</tr>
<tr>
<td>HMMTOP2</td>
<td>65.89</td>
<td>90.34</td>
<td>89.98</td>
</tr>
<tr>
<td>PHDhtm v.1.96</td>
<td>38.37</td>
<td>74.43</td>
<td>84.59</td>
</tr>
<tr>
<td>MEMSAT3</td>
<td>64.84</td>
<td>87.67</td>
<td>91.09</td>
</tr>
<tr>
<td>TopPred2</td>
<td>50.39</td>
<td>84.50</td>
<td>90.05</td>
</tr>
<tr>
<td>SOSUI 1.1</td>
<td>56.98</td>
<td>85.06</td>
<td>92.17</td>
</tr>
<tr>
<td>SPLIT4</td>
<td>65.12</td>
<td>89.77</td>
<td>91.56</td>
</tr>
<tr>
<td>ConPred-II</td>
<td>69.14</td>
<td>90.94</td>
<td>91.31</td>
</tr>
<tr>
<td>Phobius</td>
<td>67.05</td>
<td>88.72</td>
<td>93.58</td>
</tr>
<tr>
<td>PolyPhobius</td>
<td>67.44</td>
<td>90.91</td>
<td>91.28</td>
</tr>
</tbody>
</table>

8.4.3. Performance analysis at residue level

The performance of RF-TMH is also evaluated at per residue level. At residue level both transmembrane and non-transmembrane segments are considered, while in protein and segment level only transmembrane helix segments are measured. The performance of RF-TMH is analyzed by four measures including accuracy (Q_{2}), recall (Q^{obs}), precision (Q^{pred}), and MCC. The success rates of RF-TMH using low-resolution dataset are 90.84, 87.81, 81.11%, and 0.78, accuracy, recall, precision, and MCC, respectively. On the same dataset, Le et al. have introduced SVMtop that achieved 89.23% of accuracy, 87.50% of recall, 80.35% of precision, and 0.77 of MCC [194]. TMHMM2 has been developed by Krogh et al. yielded 89.23, 82.82, 83.03%, and 0.76, accuracy, recall, precision, and MCC, respectively [2]. The predicted outputs of ConPred-II are 90.07% of accuracy,
84.37% of recall, 84.13% of precision, and 0.78 of MCC [193]. In case of high-resolution dataset, the success rates of our proposed method are 92.13% accuracy, 89.27% recall, 93.33% precision, and 0.84 MCC. In contrast, among the existing methods the highest results are obtained by SVMtop, which is 90.90% accuracy, 87.84% recall, 84.36% precision, and 0.81 MCC. The next highest results have been achieved by PolyPhobius that is 88.79% accuracy and 0.77 MCC, while SPLIT4 83.84% recall and ConPred-II 84.17% precision. So the performance of RF-TMH is better than all the previously state of the art methods in the literature. The importance of RF-TMH is that it not only yielded good results compared to existing methods but also increased the length of overlap segments, which are 11 residues long. In earlier methods, Jayasinghe et al. have considered the length of helix segment residue equals 3 [65] then further increased it in another results can be found in the literature 9 [79]. Moller et al. have used an overlap of 9 residues [81] numerous similar.

The empirical results reveal that RF-TMH model has achieved promising results at all levels. These achievements have been possible because of merging of the two powerful representation methods of amino acids and ensemble based classifier weighted RF. It is anticipated that RF-TMH model might play a significant role and will provide vital information for further structural and functional studies on membrane proteins.

9.1. Introduction

In this chapter, we have considered the outer membrane proteins (OMPs). OMPs perform physiological role by enabling cell membranes through specific topology, pore formation, and transportation. However, the prediction of OMPs from protein sequences is complicated due to their vague patterns and limited number of structures. Looking at the significance of OMPs; it is essential to develop a robust and high precision prediction model. For this purpose, we introduce a promising and high quality prediction model for discriminating OMPs. The proposed approach is based on topogenic sequence representation and machine learning. The protein sequences are expressed through three discrete sequence representation methods namely, AA composition, PseAA composition, and SAAC. Further, feature extraction schemes are concatenated in different combinations to investigate their discrimination capabilities. Three different natures of classification algorithms such as GP, KNN, and fuzzy KNN are adopted to explore the weaknesses and strengths of these algorithms. Among the utilized classifiers, GP is an evolutionary algorithm based on the principle of Darwinian natural selection. Where KNN is an instance based learner. It is also called voting classifier. Its decision, drawn by calculating the distance between instances, is dependent on the neighboring samples. On the other hand, Fuzzy KNN is emerged from the combination of fuzzy theory and KNN classifier where instances are classified by considering the distance between the test instance and each individual neighbor. The classification algorithms are evaluated by 5-fold cross validation. The performance measures such as accuracy, sensitivity, MCC, precision, and ROC curve are used.
9.2. Materials and Methods

Datasets are discussed as materials and feature extraction schemes and proposed system are explained as methods in this section.

9.2.1. Datasets

In order to develop a predictor for the discrimination of OMPs, we have used two benchmark datasets. The first un-redundant dataset utilized was constructed by Park and Coworkers [115]. The original dataset was developed by Gromiha and Suwa in 2005, consisting of 377 OMPs, 268 alpha-helical membrane proteins, and 674 globular proteins [111]. The dataset contains redundancy and homologous bias between OMPs and alpha-helical membrane proteins. In order to completely remove or at least reduce the redundancy between protein sequences, 40% CD-HIT is applied. Consequently, all those sequences are discarded from the dataset, which have more than 40% similarity with other sequences. In addition, the homologous bias is eliminated or decreased using 25% cutoff threshold. After applying 25% cutoff threshold, all those sequences are excluded from the dataset, which have more than 25% sequence identity. As a result, the obtained dataset (DS1) contains 1,087 protein sequences of which 208 are OMPs, 673 globular proteins, and 206 are alpha-helical membrane proteins. In order to further decrease the redundancy from the dataset 25% similarity is applied. The second benchmark dataset (DS2) is derived from DS1 consisting of only those sequences, which have less than 25% similarity. DS2 contains 963 protein sequences of which 112 are OMPs, 673 are globular proteins, and 178 are alpha-helical membrane proteins.

9.2.2. Feature Extraction Schemes

In this study, three biochemical methods namely, AA composition, PseAA composition, and SAAC are used for protein sequence representation. AA composition only represents the occurrence frequency of each amino acid in a protein sequence. PseAA composition is a pool of two kinds of information: frequency of each amino acid and order effects of amino acid residues. SAAC extracts information from different parts of protein sequence. More details about AA composition, PseAA composition, and SAAC are provided in
Chapter 3. Hybrid versions are formed by the different combinations of AA composition, PseAA composition, and SAAC. For instance hybrid model1 is developed by concatenating AA composition and PseAA composition, whereas hybrid model2 is formed by fusing AA composition and SAAC, and so on.

9.2.3. Proposed System

Due to the vital role of OMPs in a cell, it is essential to predict OMPs with high accuracy. In order to choose the best classification algorithm for the discrimination of OMPs, we have investigated the performance of three different sorts of Machine learning methods including GP, KNN, and fuzzy KNN. GP is an evolutionary algorithm based on the principal of Darwinian natural selection and inspired by evolutionary biology. GP solves the problem by creating a computer program and finds the best one among those programs. KNN is an instance based learner, makes the decision on proximity by calculating the distance between instances. Fuzzy KNN is the combination of fuzzy theory and KNN classifier where instances are classified by considering the distance between the test instance and each individual neighbor. Fuzzy KNN classifier also defines a new class label whenever the maximum membership value is less than a predefined threshold $\theta$. The defined instances are included in the training data for future references. The framework of the proposed system is depicted in Figure 9.1. In the first phase, all the three different types of classification algorithms in conjunction with individual and hybrid versions of the feature extraction schemes are evaluated for discriminating OMPs from non-OMPs. After empirical analysis, the best feature extraction scheme SAAC and classification algorithm fuzzy KNN are selected for further investigation. In the second phase, only fuzzy KNN coupled with SAAC is evaluated for discriminating OMPs from alpha-helix membrane and Globular proteins. The performance of the classification algorithms is assessed by 5-fold cross validation.
9.3. Experimental Results and Discussion

In the research community, mostly, three cross-validation tests such as sub-sampling, jackknife, and independent dataset tests are adopted in order to evaluate the performance of the prediction models. Among the three cross-validation tests, jackknife test is conceived the most effective and least arbitrary, therefore, many investigators have used this to test the power of prediction models. However, jackknife test is time consuming. Therefore, we have used 5-fold cross-validation test. It divides the data randomly into five mutually exclusive folds of almost, equal size. One fold is single out for testing and the remaining folds are used for training. The whole process is reiterated five times where each fold takes place as testing. The performance of the classification algorithms is measured through accuracy, sensitivity, specificity, MCC, and ROC curve.
9.3.1. OMPs vs. non-OMPs

In this section, the performance of GP, KNN and fuzzy KNN using individual and hybrid models of AA composition, PseAA composition, and SAAC for discrimination of OMPs and non-OMPs is discussed.

9.3.1.1. GP

Table 9.1 shows the performances of GP using alone and different combination of AA composition, PseAA composition, and SAAC on both datasets. In case of individually feature extraction schemes, GP has yielded the highest success rates using SAAC based features on DS1 whereas in case of DS2, the best results have been achieved using AA composition. The predicted results of GP on both datasets using SAAC are 92.27% of accuracy with MCC of 0.73 on DS1 and 93.94% of accuracy with MCC of 0.71 on DS2. On the other hand, the results of GP using AA composition are 91.90 and 94.33% of accuracy with MCC of 0.72 on both datasets. In contrast, the performance of PseAA composition is poor than that of AA composition and SAAC on both datasets. In order to more strengthen the discrimination capabilities of the feature space, the feature extraction schemes are concatenated in different combinations. First, AA composition and PseAA composition based features are merged and hybrid model1 of dimension 86-D is developed. The success rates of GP using hybrid model1 are 90.14 and 92.39% of accuracy while 0.65 and 0.63 of MCC on DS1 and DS2, respectively. Next, the AA composition is combined with SAAC and formed the hybrid model2 of 80-D. The performance of GP using hybrid model2 is boosted compared to hybrid model1. The obtained results of GP are 92.64% of accuracy and 0.75 MCC on DS1 whereas 94.84% of accuracy and 0.76 MCC on DS2. Further, PseAA composition based features are fused with SAAC based features and developed hybrid model3 of 126-D. In case of DS1, the accuracy of GP is not good but MCC is better, on the other hand, in case of DS2, the accuracy is good but MCC is poor. Finally, all the three feature extraction schemes are concatenated and formed hybrid model4 of 146-D. The obtained results of GP using hybrid model4 are 93.56 and 94.45% of accuracy whereas 0.70 and 0.72 MCC on DS1 and DS2, respectively. After the analysis of experimental results, we have noticed that GP has achieved the highest accuracy using hybrid model2, and the highest MCC using
hybrid model3 in case of DS1. On the other hand, the highest accuracy and MCC are yielded using hybrid model2 in case of DS2. Overall the performance of GP for discriminating OMPs and non-OMPs is not remarkable.

9.3.1.2. KNN

The performance of KNN is also investigated in conjunction with alone and hybrid models. The success rates of KNN with varying number of neighbors denoted by K are reported in Table 9.1. KNN has discriminated OMPs and non-OMPs in a better way and yielded better results using all the feature extraction schemes on both datasets compared to GP. Using the first discrete method AA composition, KNN has achieved an accuracy of 93.92% and MCC of 0.80 on DS1, and 96.36% of accuracy with MCC of 0.83 on DS2. In case of PseAA composition, KNN has yielded an accuracy of 94.94 and 96.39% with MCC of 0.83 on DS1 and DS2, respectively. Using the third method SAAC, the performance of KNN is better than using AA composition and PseAA composition. The obtained accuracies and MCC of KNN are 95.49, 96.78%, and 0.85 on DS1 and DS2, respectively.

Table 9.1. Success rates of GP, KNN, and Fuzzy KNN using 5-fold cross-validation

<table>
<thead>
<tr>
<th>Methods</th>
<th>GP Accuracy</th>
<th>MCC K</th>
<th>KNN Accuracy</th>
<th>MCC K</th>
<th>Fuzzy KNN Accuracy</th>
<th>MCC K</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AAC</td>
<td>91.90</td>
<td>0.72</td>
<td>3</td>
<td>93.92</td>
<td>0.80</td>
<td>7</td>
</tr>
<tr>
<td>PseAAC</td>
<td>90.15</td>
<td>0.66</td>
<td>5</td>
<td>94.94</td>
<td>0.83</td>
<td>7</td>
</tr>
<tr>
<td>SAAC</td>
<td>92.27</td>
<td>0.73</td>
<td>9</td>
<td>95.49</td>
<td>0.85</td>
<td>9</td>
</tr>
<tr>
<td>AAC+ PseAAC</td>
<td>90.14</td>
<td>0.65</td>
<td>3</td>
<td>94.75</td>
<td>0.82</td>
<td>7</td>
</tr>
<tr>
<td>AAC+SAAC</td>
<td>92.64</td>
<td>0.75</td>
<td>3</td>
<td>95.21</td>
<td>0.84</td>
<td>5</td>
</tr>
<tr>
<td>PseAAC+SAAC</td>
<td>91.25</td>
<td>0.79</td>
<td>3</td>
<td>95.30</td>
<td>0.84</td>
<td>5</td>
</tr>
<tr>
<td>AAC+ PseAAC +SAAC</td>
<td>93.56</td>
<td>0.70</td>
<td>1</td>
<td>95.12</td>
<td>0.84</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>DS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AAC</td>
<td>94.33</td>
<td>0.72</td>
<td>3</td>
<td>96.36</td>
<td>0.83</td>
<td>5</td>
</tr>
<tr>
<td>PseAAC</td>
<td>92.01</td>
<td>0.60</td>
<td>3</td>
<td>96.39</td>
<td>0.83</td>
<td>5</td>
</tr>
<tr>
<td>SAAC</td>
<td>93.94</td>
<td>0.71</td>
<td>9</td>
<td>96.78</td>
<td>0.85</td>
<td>5</td>
</tr>
<tr>
<td>AAC+ PseAAC</td>
<td>92.39</td>
<td>0.63</td>
<td>3</td>
<td>96.26</td>
<td>0.82</td>
<td>5</td>
</tr>
<tr>
<td>AAC+SAAC</td>
<td>94.84</td>
<td>0.76</td>
<td>3</td>
<td>96.13</td>
<td>0.81</td>
<td>5</td>
</tr>
<tr>
<td>PseAAC+SAAC</td>
<td>93.04</td>
<td>0.66</td>
<td>3</td>
<td>96.65</td>
<td>0.84</td>
<td>5</td>
</tr>
<tr>
<td>AAC+ PseAAC +SAAC</td>
<td>94.45</td>
<td>0.72</td>
<td>3</td>
<td>96.13</td>
<td>0.81</td>
<td>5</td>
</tr>
</tbody>
</table>
In case of hybrid models, the performance of KNN is better than using AA composition and PseAA composition, but still is not considerable. It has yielded the highest results using hybrid model3 on both datasets, which are 95.30 and 96.65% of accuracies compared to the rest of hybrid models. Finally, the experimental results reveal that KNN has obtained good results using SAAC based feature compared to the rest of individual and hybrid based features.

9.3.1.3. Fuzzy KNN

The classification power of fuzzy KNN is examined using all the three feature extraction schemes alone and hybrid models. The predicted results of Fuzzy KNN with varying number of nearest neighbors denoted as K are reported in Table 9.1. The performance of Fuzzy KNN is outstanding on each feature extraction scheme and performed the discrimination between OMPs and non-OMPs with high precision compared to GP and simple KNN on both datasets. Using AA composition, Fuzzy KNN has achieved an accuracy of 98.90% with 0.94 of MCC on DS1 and 99.39% of accuracy with MCC of 0.95 on DS2. In PseAA composition, Fuzzy KNN has yielded 98.80% accuracy and 0.94 MCC on DS1 and 99.35% accuracy and 0.94 MCC on DS2. The performance of AA composition is slightly better compared to PseAA composition. In contrast, using SAAC, the highest accuracy is reported on both datasets. It has yielded accuracy of 99.00 and 99.55% as well as MCC of 0.95 and 0.96 on DS1 and DS2, respectively. In case of hybrid models, fuzzy KNN has also obtained prominent results compared to GP and simple KNN. Fuzzy KNN has generated more or less similar results on all the above mentioned feature extraction schemes on both the datasets. Finally, the empirical results reveal that fuzzy KNN has performed well using SAAC based features compared to the rest of feature extraction schemes. On the other hand, performance comparison between simple KNN and fuzzy KNN on the basis of number of nearest neighbors is depicted in Figure 9.2 and 9.3. ROC curve shows the tradeoff between true positive rate and false positive rate on various values of K from 1 to 10 for simple KNN and fuzzy KNN.
Fuzzy KNN performed well due to some reasons, the first one is that fuzzy KNN makes the prediction by considering both the number of closest neighbors and the distance between the test instance and its neighbors while KNN only looks at number of closest neighbors. The second one is that as the value of K increases, the training instances that are nearer to the test instance contribute more to the prediction result comparing with the training instances that are far away. Therefore, the success rate of fuzzy KNN is higher compared to simple KNN.

9.3.2. Discrimination of OMPs from Alpha-helix membrane and Globular proteins

Owing to the special attributes of Fuzzy KNN along with outstanding performance for discriminating OMPs and non-OMPs, we have selected fuzzy KNN and SAAC based features for further discrimination of OMPs from alpha-helix membrane and globular proteins. Among all the feature extraction schemes, SAAC based features have efficiently discerned the patterns of OMPs and non-OMPs, which have enhanced the discrimination power of the fuzzy KNN. The success rates of fuzzy KNN coupled with SAAC on both datasets are listed in Table 9.2.
Figure 9.3. ROC Curve for KNN and Fuzzy KNN using SAAC on DS2

Table 9.2. Prediction performance of Fuzzy KNN

<table>
<thead>
<tr>
<th>Measures</th>
<th>OMPs vs. Globular K=9</th>
<th>OMPs vs. alpha-helix K=3</th>
<th>OMPs vs. Non-OMPs K=9</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS1</td>
<td>Accuracy</td>
<td>Sensitivity</td>
<td>Specificity</td>
</tr>
<tr>
<td></td>
<td>98.77</td>
<td>98.28</td>
<td>99.81</td>
</tr>
<tr>
<td></td>
<td>Sensitivity</td>
<td>93.57</td>
<td>98.86</td>
</tr>
<tr>
<td></td>
<td>Specificity</td>
<td>99.81</td>
<td>97.63</td>
</tr>
<tr>
<td></td>
<td>MCC</td>
<td>0.95</td>
<td>0.96</td>
</tr>
</tbody>
</table>

| DS2               | Accuracy              | Sensitivity              | Specificity           | MCC                   |
|                   | 99.81                 | 99.90                    | 100                   | 99.55                 |
|                   | Sensitivity           | 99.79                    | 99.88                 | 99.84                 |
|                   | Specificity           | 100                      | 100                   | 95.55                 |
|                   | MCC                   | 0.98                     | 0.99                  | 0.96                  |

In case of DS1, the predicted results of fuzzy KNN for OMPs and globular proteins are 98.77% accuracy, 93.57% sensitivity, 99.81% specificity, and 0.95 MCC, whereas in case of DS2, 99.81% of accuracy, 99.79% of sensitivity, 100 of specificity, and 0.98 of MCC are obtained. These performances are recorded on the value of K=9. In contrast, the success rates of fuzzy KNN for discriminating OMPs from alpha-helical membrane proteins are 98.28% of accuracy, 98.86% of sensitivity, 97.63% specificity, and 0.96 of MCC on DS1 whereas, 99.90, 99.88, 100%, and 0.99 are accuracy, sensitivity,
specificity, and MCC, respectively on DS2. Fuzzy KNN has yielded these results on the value of K=3.

9.3.3. Comparison with the Existing approaches on DS1

We have compared the proposed approach with already existing methods using 5-fold cross validation test. Performance comparison of our proposed approach with other existing methods is shown in Table 9.3. First, the discrimination of OMPs and globular proteins are analyzed. In this regard, Park et al. have introduced SVM and obtained 94.4% accuracy, 88.0% sensitivity, 96.4% specificity, and 0.84 MCC [115]. Yan et al. have utilized KNN, which has yielded 96.0% accuracy, 87.5% sensitivity, 98.7% specificity, and 0.88 MCC [117]. Furthermore, SVM adopted by Goa et al. has obtained better results than that of Park and Yan Method’s [5]. The predicted results are 98.2% accuracy, 93.8% sensitivity, 99.6% specificity, and 0.94 MCC. On the other hand, our proposed approach has yielded an accuracy of 98.7%, sensitivity of 93.5%, specificity of 99.8%, and MCC of 0.95. The overall accuracy of our proposed approach is 0.57% higher than the highest previous accuracy. Subsequently, we have also compared the performance of several methods for OMPs and alpha-helix membrane proteins. Park et al. predicted model has yielded 95.9% accuracy and 0.92 MCC for discriminating OMPs from alpha-helix membrane proteins [115]. The predicted results of Yan et al. model are 94.7% accuracy and 0.89 MCC [117]. The success rates of Goa et al. are 96.4% accuracy and 0.92 MCC. In contrast, the predicted results of our proposed approach are 98.3% accuracy with MCC of 0.96. Thus, our proposed approach has achieved 1.9% higher accuracy than that of Goa et al. method. Finally, the outcomes related to the discrimination of OMPs from non-OMPs protein are compared. Park et al. predicted model has yielded an accuracy of 95.2% with MCC of 0.84 for discriminating OMPs and non-OMPs [115]. The success rates of Yan et al. method are 96.1% accuracy and 0.87 MCC, while the predicted results of Goa et al. method are 97.8% accuracy and MCC 0.92 [5; 117].
Table 9.3. Comparative analysis between the proposed approach and existing approaches on DS1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OMPs vs. Globular</td>
<td>OMPs vs. alpha-helix</td>
<td>OMPs vs. Non-OMPs</td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>94.4</td>
<td>95.9</td>
<td>95.2</td>
<td>98.7</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>88.0</td>
<td>99.0</td>
<td>79.3</td>
<td>95.9</td>
</tr>
<tr>
<td>Specificity</td>
<td>96.4</td>
<td>92.7</td>
<td>99.0</td>
<td>99.8</td>
</tr>
<tr>
<td>MCC</td>
<td>0.84</td>
<td>0.92</td>
<td>0.84</td>
<td>0.92</td>
</tr>
</tbody>
</table>

On the other hand, our proposed approach has yielded the highest success rates of 99.0% accuracy and 0.94 MCC. The predicted accuracy of our proposed approach is 1.2% higher than that of Goa et al. proposed approach. The comparison between several already existing classification models with proposed approach is illustrated in Figure 9.4.

![Figure 9.4. Performance comparison of our proposed approach with already existing models on DS1](image-url)
9.3.4. Comparison with the Existing approaches on DS2

The success rates of fuzzy KNN on DS2 compared with state of the arts in the literature are listed in Table 9.4. DS2 has been considered for the first time by Yan et al. who have used single sequence and homologous sequences for discriminating OMPs and non-OMPs [110]. The predicted results using single sequence are 91.8% accuracy, 66.1% sensitivity, 95.2% specificity, and 0.60 MCC. In contrast, the performance of homologous sequences approach is better compared to single sequence approach. It has yielded an accuracy of 95.3%, sensitivity of 72.3%, specificity of 98.2%, and MCC 0.76. Recently, Mizianty et al. have developed OMBBpred, which has obtained 98.5% accuracy, 88.2% sensitivity, 99.9% specificity, and 0.93% MCC [119].

Table 9.4. Comparative analysis between the proposed approach and existing approaches on DS2

<table>
<thead>
<tr>
<th>Measures</th>
<th>Yan et al. [110]</th>
<th>Yan et al. [110]</th>
<th>Mizianty et al. [119]</th>
<th>Proposed Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OMPs vs. Non-OMP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>91.8</td>
<td>95.3</td>
<td>98.5</td>
<td><strong>99.5</strong></td>
</tr>
<tr>
<td>Sensitivity</td>
<td>66.1</td>
<td>72.3</td>
<td>88.2</td>
<td>99.8</td>
</tr>
<tr>
<td>Specificity</td>
<td>95.2</td>
<td>98.2</td>
<td>99.9</td>
<td>95.5</td>
</tr>
<tr>
<td>MCC</td>
<td>0.60</td>
<td>0.76</td>
<td>0.93</td>
<td>0.96</td>
</tr>
</tbody>
</table>

On the other hand, our proposed approach has yielded 99.55% accuracy, 99.8% sensitivity, 95.5% specificity, and 0.96 MCC, which are better than already published methods. The MCC of our proposed approach is 0.3 higher than that of Mizianty et al. method [119]. Finally, we conclude that the remarkable achievement of our proposed approach is because of the discriminative capabilities of the difference in the amino acid composition at the N- and C-terminus through SAAC and special characteristics of Fuzzy KNN.
10. Conclusions and Future Directions

This thesis contributes in the field of Computational Biology and Bioinformatics by applying various intelligent techniques in order to enhance the probability of true prediction of membrane proteins. The prediction of membrane proteins has been carried out from protein sequences. Biochemical, physicochemical, and evolutionary profiles based methods are used for protein sequence representation. Several feature selection techniques are utilized to eliminate irrelevant features from the feature space. Machine learning, neural networks, and evolutionary programming techniques are applied in order to obtain advanced results compared to the contemporary techniques. Self-consistency, jackknife, independent dataset, 5-fold, and 10 fold cross validation tests are adopted to assess the performance of the classification algorithms. A number of performance measures such as accuracy, sensitivity, specificity, MCC, F-measure, Q-statistics, ROC curve are used to analyze the performance of the classification algorithms. The research is conducted into two phases: in the first phase, we predict the types of membrane proteins and their structure in the second phase.

10.1. Membrane Protein Types

The prediction of membrane protein types are carried out in four parts. Chapter 4 presents the first part of phase-I. In the first part, we have developed a prediction model for membrane protein types using CPSR and individual classifier. PCA is applied as feature selection technique and SVM, PNN, and GRNN are used as classification algorithms. The importance of this model is CPSR, because CPSR is the composition of seven different attributes. In chapter 5, an ensemble classification model is developed for the prediction of membrane protein types, which is the contribution of part 2. Protein
sequences are expressed by three biochemical methods including DWT, PseAA composition, and SAAC. Hybrid models are also formed by the combination of biochemical methods. A comparison has been shown between hybrid models of full feature space and reduced feature space where the features are reduced using NPE. Ensemble classification model is developed in two ways: simple majority voting based ensemble and GA based ensemble. Promising results have been achieved by GA based ensemble classification in conjunction with SAAC. The third part of phase-I is presented in chapter 6. In this chapter, SAAC and evolutionary profiles PSSM are combined in order to reveal all the important features. Different kernels of SVM are investigated as classifiers. In addition, error correction code BCH is incorporated with SVM to correct the misclassification instances of the simple SVM. In the final part of phase-I, we have developed a two layer web predictor presented in chapter 7. The first layer is developed for discrimination between a membrane and a non-membrane protein while the second layer classifies only the type of membrane protein. It is developed by fusing physicochemical properties of amino acids and SAAC coupled with SVM. The drawback in a single layer predictor is that it also identifies a non-membrane protein as a type of membrane protein. This problem is tackled by our web predictor, which first determines a protein query as a membrane or a non-membrane proteins and then its further classifies the type of membrane protein only if it is a membrane.

10.2. Membrane Protein Structures

In the second phase of this work, we have predicted the structure of membrane proteins. The structure of membrane proteins is predicted as transmembrane helix and outer membrane proteins. In chapter 8, RF-TMH system is proposed for the prediction of transmembrane helix. The model is developed using hybrid features of physicochemical properties and compositional index of amino acids coupled with weighted RF. SVD is applied to find out variation among features. Consequently, only those features are selected from the feature space, which have above 83% variance. Transmembrane helices are not only predicted with high accuracy but also more overlap with observed helix than that of already published methods. In chapter 9, outer membrane proteins are predicted. In this chapter, biochemical methods namely, AA composition, PseAA composition, and
SAAC are used as feature extraction schemes. Hybrid versions of these methods are also developed to investigate the discrimination power of different combinations. KNN, fuzzy KNN, and GP are adopted as classification algorithms. Among classification algorithms, the performance of fuzzy KNN in combination with SAAC has obtained outstanding results than that of simple KNN, GP and existing methods. It has effectively discriminated outer membrane proteins from alpha-helix membrane and Globular protein.

Finally, we have reckoned that our proposed approach for membrane proteins might play a significant role not only in Computational Biology, Molecular Biology, and Bioinformatics, but also in pharmaceutical industries. In addition, our web predictor provides sufficient information to the researchers and academicians in future research.

10.3. Future Directions

Due to the large scale of innovation of protein sequences, the number of protein sequences in data bank is consistently growing. Currently, the prime challenges in the area of Computational Biology and Bioinformatics are to store, analyze, and annotate unprocessed data. However, manual identification of the membrane proteins in some situations is very difficult or even impossible. In this regard, some computational methods have been developed. Several online web servers are also lunched. These methods have provided user friendly tools in common use for identifying the structure and types of membrane protein in Biochemistry, Biomedical, and Molecular Biology research. It is also useful in applications related to drug discovery and pharmacy. However, in all these methods, the major target is to enhance the accuracy because the aim of the computational methods is to predict membrane proteins with high accuracy. Likewise, the principle objective of our research was to predict the types and structure of membrane protein with high precision. Due to the complex nature and limited availability of membrane protein structure; it should be the spotlight of future research work in the Machine learning community. However, other related but important issues such as computational cost, complexity as well as reliability and availability of web predictors should also be considered in future research.
References


