2017

Multiple Kernel Learning And Its Application In Bioinformatics

Hasan, Md. Al Mehedi

University of Rajshahi

MULTIPLE KERNEL LEARNING AND ITS APPLICATION IN BIOINFORMATICS

A Thesis Submitted to the Department of Computer Science and Engineering,
Faculty of Engineering of the University of Rajshahi for the Degree of
Doctor of Philosophy

Submitted
By
Md. Al Mehedi Hasan

Under the Supervision of

Professor Dr. Shamim Ahmad

Department of Computer Science And Engineering
Faculty of Engineering
University of Rajshahi, Rajshahi-6205
Bangladesh

April, 2017

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

Md. Al Mehedi Hasan
Ph. D. Fellow
Dept. of Computer science and Engineering
Faculty of Engineering
University of Rajshahi
Rajshahi-6205, Bangladesh.
&
Assistant Professor
Dept. of Computer Science and Engineering
Rajshahi University of Engineering and Technology
Rajshahi-6204, Bangladesh.

Certificate

This is to certify that Md. Al Mehedi Hasan, has carried out a research work entitled "**Multiple Kernel Learning And Its Application In Bioinformatics**" for the fulfillment of Doctor of Philosophy (Ph. D.) under my supervision and guidance in the Department of Computer Science and Engineering, Faculty of Engineering, University of Rajshahi, Bangladesh. I also certify that this work has not been presented for any degree or award elsewhere.

Prof. Dr. Shamim Ahmad
Supervisor
Dept. of Computer Science and Engg.
Faculty of Engineering
University of Rajshahi
Rajshahi-6205, Bangladesh

Prof. Dr. Md. Khademul Islam Molla
Co-supervisor
Dept. of Computer Science and Engg.
Faculty of Engineering
University of Rajshahi
Rajshahi-6205, Bangladesh

Acknowledgments

I am deeply grateful for the opportunity to have worked with so many brilliant and creative minds over the last few years. First of all, I would like to express my deep sense of gratitude to my reverend principal supervisor Dr. Shamim Ahmad, Professor, Dept. of Computer Science and Engineering, University of Rajshahi, Bangladesh, for his abundant encouragement, constant supervision, guidance, active help and cooperation and valuable suggestions, and providing financial support in completing this thesis work successfully.

Likewise, I am very grateful to my another co-supervisors Professor Md. Khademul Islam Molla, Dept. of Computer Science and Engineering, University of Rajshahi, Bangladesh, for giving advise, cooperation and for providing support in hosting all of our developed tools.

Of course, I owe the utmost gratitude to late Professor Mohammad Naser, Department of Statistics, University of Rajshahi, Bangladesh, who initially introduced me to the topic of multiple kernel learning (MKL) in 2011 and gave me advice and support in various respects from the very beginning of my PhD studies. I am also very grateful to Prof. Naser to take me as a member of his statistical learning group (SLG) and for spending of his valuable time for our extensive and fruitful discussions on various mathematical and statistical problems, sharing his immense knowledge and deep insights with me. It is rare that a professor gives so much of his time in mentoring a student. I wish my deep condolence to his family and pray Almighty to leave his soul at a heavenly place.

I am also thankful to Prof. Dr. Jahanur Rahman, Mr. Md. Rabiul Haque, Mr. Masum Murshed, Mr. Rajandra Bhowmik and all other members of SLG for their help and inspiration to continue my work.

I also express my gratitude and thanks to my honorable teachers Prof. Dr. Abu Raihan Shoyeb Ahmed Siddique, Prof. Dr. A K M Akhtar Hossain, Prof. Dr. Md. Ekramul Hamid, Prof. Dr. Somlal Das, Sangeeta Biswas and all other teachers of this department for their valuable suggestions and constant encouragement during the research work.

I am much more grateful to my family members especially to my parents, wife, all of my friends and well wishers for their inspiration and best wishes to move forward.

Finally, I express my gratitude to the Almighty for giving me strength and courage to complete my thesis work.

Dedication

I would like to dedicate this thesis to my loving parents and dear wife.

Abstract

During the last decades, the support vector machine (SVM) has been applied broadly within the field of computational biology or bioinformatics to answer biological questions and to reach valid biological conclusions. However, a successful application of SVM depends heavily on the determination of the right type and suitable parameter settings of kernel functions. The selection of the appropriate kernel and kernel parameters are both considered as the choice of kernel problem. Therefore, kernel learning becomes a crucial problem for all kernel-based methods like the SVM. Recently, the multiple kernel learning (MKL) has been developed to tackle the kernel learning problem efficiently and gives some scopes to improve the performance of a system.

On the other hand, sometimes it is desirable to handle multiple data sources for pattern recognition in the field of bioinformatics. In this context, if these data sources are combined appropriately as one data source, it is then possible to provide a more "complete" representation of an entity which in turns, enhances the performance of a pattern recognition system. In this case, MKL also provides a way to combine features from various data sources, where each kernel will be dedicated to a particular type of data source.

In order to use the above two advantages of MKL, we have applied MKL in two challenging problems in bioinformatics: protein subcellular localization prediction and protein post-translational modifications (PTMs) prediction. The knowledge of the subcellular localization and PTMs of proteins are important for both basic research and drug development. Recently various types of computational tools have been developed to predict the subcellular localization and PTMs or PTMs site of a protein through different types of machine learning algorithms. However, in order to meet the current demand of drug development and basic research, both of the above prediction systems require additional effort to produce efficient high-throughput tools.

In our thesis work, we have applied MKL in order to give potential solution for the choice of kernel problem in one of the two mentioned applications of bioinformatics. In this case, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) has been considered as the search space of the choice of kernel problem. Moreover, since both applications can be solved from various data sources, features from various sources are fused using multiple kernel learning with the expectation of better improvements. The experimental results show that the prediction systems using MKL based SVM provide better performance than other top existing systems in both applications. We have completed nine experiments throughout this thesis work. Where, four of those show the capability of single kernel based SVM, one shows the effects of the choice of kernel problem, one provides potential solution to the choice of kernel problem using MKL, finally, rest three show the application of MKL in handling multiple data sources. In addition to it, we have developed six user-friendly web servers for six specific prediction purposes as a product of these experiments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

1.1.1 Multiple Kernel Learning

In machine learning, kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. Algorithms capable of operating with kernels include the support vector machines (SVM), Gaussian processes, kernel principal components analysis, kernel canonical correlation analysis, kernel Fisher discriminant analysis, kernel independent component analysis, kernel ridge regression, spectral clustering, linear adaptive filters, extreme learning machine and many others. Among these kernel methods, support vector machine (SVM) is the best known member which a class of algorithm for pattern analysis (classification, regression).

SVM has been successfully applied to a number of real-world problems and is now considered state-of-the-art in various domains. Recently. support vector machine (SVM) is increasingly used to solve various problems in computational biology. It offers versatile tools to process, analyze, and compare many types of data, and offer state-of-the-art performance in many cases [1].

With the rapid increase in size of the biological databanks, understanding the data has become critical. Such an understanding could lead us to the elucidation of the secrets of life or ways to prevent certain currently non-curable diseases such as HIV [2]. Although laboratory experiment is the most effective method for investigating the data, it is very financially and labour expensive. As a result, computational algorithms and tools have therefore been widely used in classification, regression and cluster analysis of biological data. At present, support vector machine (SVM) has been applied to give the best prediction performance in many areas of bioinformatics including protein function prediction, protease functional site recognition, transcription initiation site prediction and gene expression data classification. This is because SVM is designed to maximize the margin to separate two classes so that the trained model generalises well

on unseen data [2]. Most other computer programs implement a classifier through the minimization of error occurred in training, which leads to poorer generalization.

However, a successful application of SVM depends heavily on the determination of the right type and suitable parameter settings of kernel functions. The selection of the appropriate kernel and kernel parameters are both considered as the choice of kernel problem [3, 4]. After more than a decade of research it still remains an unsolved problem to find the best kernel for a task at hand [5]. Most frequently, the kernel is selected from a candidate set according to its generalization performance on a validation set, which is held back at training time. Clearly, the performance of such an algorithm is limited by the performance of the best kernel in the set and can be arbitrarily bad if the kernel does not match the underlying learning task. Therefore, kernel learning becomes a crucial problem for all kernel-based methods, in which the central question is how to choose an appropriate kernel. The Multiple Kernel Learning (MKL) framework has been developed recently to tackle efficiently the kernel learning problem [5].

Multiple kernel learning (MKL) provides a systematic approach to using data to learn the most suitable kernel function for a learning task [4]. These methods are usually designed to combine kernels from a given set. Since a kernel function corresponds to a notion of similarity between data instances, combining multiple kernels can be interpreted as combining different notions of similarity [4].

Simplest way to do this is taking the unweighted sum of all kernels. This gives equal priority to every kernel but this method may not be an ideal because some kernels may not be as good as other kernels. One better method is taking the weighted sum (linear combination) of the given kernels [6, 7, 8] takes the convex combination of kernels. Again, some methods [9] took the nonlinear combination of kernels. These methods gives fixed weights to the kernels over whole input space. But, using different weights for different points may produce a better classifier overall. With this idea, Localized MKL [10] method is proposed which combines the Kernels linearly by giving different weights at each data point.

On the other hand, in bioinformatics research, we are not only faced with the massiveness of data but also with multiple sources of the data that often provide complementary views about a subject [11]. Moreover, the multiple sources of the data can also creates various types of data formats. For example, genomic data can come in a wide variety of data formats[12]: expression data are expressed as vectors or time series; protein sequence data as strings from a 20-symbol alphabet; gene sequences are strings from a different (4-symbol) alphabet; protein-protein interactions are best expressed as graphs, and so on.

However, the multiple sources of data of a subject may result due to various reasons [11]. It may be due to difference in data extraction environment or methods and what perspective of the subject is being studied. Difference in the data extraction environment includes difference in the data sources, and the extraction methods. Each of these distinct data types of a subject provides one view of the molecular machinery

of the cell [12]. In the near future, research in bioinformatics will focus more and more heavily on methods of data fusion. Different data sources are likely to contain different and thus partly independent information about the task at hand. Combining those complementary pieces of information can be expected to enhance the total information about the problem at hand.

Multiple kernel learning (MKL) provides a potential solution of the above problem. It can combine different types of features that are represented in different data formats or same data formats. For the purpose of integrative analysis of data, in MKL, difference kernel are created for each data types and combined to be used with various kernel based learning methods such as SVM.

1.1.2 Protein Subcellular Localization Prediction

A biological cell is made up of many different compartments or organelles, these compartments are closed to each other and they have different functions as well. The proteins in the cell are responsible for most those functions required for a cell's survival. A typical cell contains approximately one billion protein molecules that reside in many different compartments or organelles, usually termed "subcellular locations" [13]. This is very interesting that a protein can perform their appropriate functions when they are located in the right subcellular locations. The knowledge of the subcellular localization of proteins is important because it (a) provides useful insights on their functions, (b) indicates how and in which kind of cellular environments they interact with each other and with other molecules, (c) helps in understanding the intricate pathways that regulate biological processes at the cellular level and (d) helps in identifying and prioritizing drug targets during the process of drug development [14].

Although various experimental approaches have been developed for determining the protein subcellular locations, but unfortunately, most of those approaches are costly and time consuming as well [15]. However, since number of newly discovered proteins has been growing exponentially, which in turns, makes the subcellular localization prediction by purely laboratory tests prohibitively expensive [16]. In this context, computational methods are developed to help biologists in selecting target proteins and designing related experiments. Moreover, the computational methods are fast and can potentially predict locations for proteins whose actual locations have not yet been experimentally determined. Various methods for predicting subcellular localization of protein sequences have been extensively studied in the last decades, and researchers have developed increasingly more new models to acquire better prediction performance.

Conventional methods for subcellular localization prediction can be roughly divided into sequence-based methods and annotation-based methods [14, 17]. Sequence-based predictors make use of (I) sequence-coded sorting signals such as WoLF PSORT [18], (II) amino acid composition information [19, 20], (III) both information sources [18, 21]. It should be noted that sequence based methods are general in that they can be applied to any newly discovered proteins [22]. However, their performance is

usually poor, especially for datasets containing sequences with low-similarity.

Annotation-based predictors use information about functional domains and motifs [23], protein-protein interaction [24], homologous proteins such as KnowPredsite [25], annotated Gene Ontology (GO) terms [26] such as Euk-OET-PLoc [27], Euk-mPLoc [28], iLoc-Gneg [29] and textual information from Swiss-Prot keywords or PubMed abstracts. The annotation-based predictors often show higher accuracies than pure sequence-based predictors although they are less robust when the protein is newly discovered one and even if its close homologues are unknown [30]. In fact, when the protein to be predicted is a newly discovered one, there is no existing annotation in the database. As a result, the prediction performance of annotation-based methods will be degraded. However, at least annotation of the close homologues of a novel protein is expected to be available since coverage of public annotation databases is increasing rapidly, which finally reduce the limitation of this method as mentioned above.

In addition to the above approaches, some researchers have developed hybrid prediction approaches such as BNCs [17], MDLoc [31], YLoc+ [30], etc. which includes the both sequence-based methods and annotation-based methods as well [17, 31, 30, 32, 28].

Since feature representations of protein are different due to their different extraction techniques, so there are some scopes to improve performance of protein subcellular localization prediction by combining multiple feature representations. MKL can be a good techinque to combine multiple features.

On the other hand, not only protein sequence information but also prediction algorithms could affect the accuracy of the subcellular localization prediction [33]. Many computational techniques, such as the neural network [34], K-nearest neighbor (KNN) [35, 36, 37] and a few ensemble classifiers [38, 39] have been introduced for the prediction of protein subcellular localization.

However, in recent days, the support vector machine (SVM) [14, 21, 33, 40] has also been extensively applied to provide potential solutions for the subcellular localization prediction. But, the selection of an appropriate kernel and its parameters for a certain classification problem influence the performance of the SVM. Here, both problems (selection of an appropriate kernel and its parameters) are considered as choice of kernel problem. However, the literature survey have showed that most of the researchers applied radial basis function (RBF) kernel to build SVM based subcellular localization prediction [14, 16, 39] and have found the value of its parameter by using different techniques such as trial and error, heuristics or grid search procedure, unfortunately, these approaches are time consuming [41].

Therefore, to reduce the complexity in finding kernel parameters such as sigma for RBF (choice of kernel from the set of RBF kernels), multiple kernel learning approach can be adopted [41]. It should be mentioned here that different values of σ of the RBF kernel parameter create different types of RBF kernel [3]. Several researchers have proposed various multiple kernel learning approach in which multiple kernels are used to construct a combined kernel [41, 42, 43, 7, 8, 9, 44, 45, 46, 47, 3]. In order to

avoid the complexity in finding kernel parameters, in MKL, the kernel parameter space are discretized into r values and created a set of r kernels using each of the r values. Finally, construct a combined kernel by combining each of the r kernels using specific rule of MKL and use this combined kernel in SVM classifier.

1.1.3 Post-translational Modifications (PTMs) Prediction

Post-translational modifications (PTMs) are widely used to modulate protein function in the cell. PTMs can be viewed as covalent modifications that increase the structural and biophysical diversity of proteins and thus enrich the information stored in the genomes. There are several different PTMs that are incorporated by the cell. To achieve the required effect, a protein may undergo a single PTM or several PTMs that may engage in cross-talk. In many cases, a single position on the protein can be altered by different modifications so that switching between several effects (or functions) can be regulated by the identity of the PTM at that position.

The major types of protein covalent modifications, such as phosphorylation, acetylation, glycosylation, methylation, and ubiquitylation, can be classified according to the type of amino acid side chain modified, the category of the modifying enzyme, and the extent of reversibility. An understanding of the scope and pattern of the these post-translational modifications in cells provides insight into the function and dynamics of proteome compositions. Moreover, various types of major human diseases including Alzheimer's disease, diabetes, Parkinson's disease, chronic renal failure, chronic lung disease, sepsis are associated with protein PTMs [48].

As a result, it requires an easiest way to detect modification in proteins. Although various experimental approaches have been developed for identifying PTMs of proteins, but unfortunately, most of those approaches are costly and time consuming as well [49, 50]. However, since number of newly discovered proteins has been growing exponentially, which in turns, makes the PTMs prediction by purely laboratory tests prohibitively expensive [16]. In this context, computational methods are developed to help biologists in selecting target proteins and designing related experiments.

There are two types of prediction systems available in the field of PTMs prediction. One is to predict the sites of PTMs of a known modified protein [51, 49, 50, 52, 53]. Another one is to predict whether an uncharacterized protein can be modified or not [54, 55]. Although various types of computational classifiers have been developed for identifying the PTM sites in protein [51, 52] or identifying PTM of protein [54, 55], in order to meet the current demand to produce efficient high-throughput tools, additional effort are required to enrich the prediction quality [49, 55].

However, one of the most important problems for these kinds of prediction is how to formulate a biological sequence with a discrete model or a vector by keeping its sequence order information or essential feature. This is because all the existing machine learning algorithms can only handle vector but not sequence samples. In case of PTMs site prediction or PTMs prediction, various types of feature representation of proteins or peptide samples is possible [51, 49, 50, 52, 53, 55]. It should be mentioned that

in the case of PTMs site prediction, protein sequences are segmented, called peptide samples, according to a specific window size. In each peptide sample, the modified residue take the middle position of the peptide segment. So, post-translational modification prediction is also a problem to handle multiple data sources. Although various researcher used single feature to predict post-translational modification prediction, combining these features can improve the performance of the existing system. MKL can be a good technique to combine multiple features.

1.2 Problem Statement

Multiple kernel learning (MKL) is very popular in machine learning domain mainly for the following two reasons:

1. the ability to give a potential solution of the choice of kernel problem (selection of an appropriate kernel and kernel parameters).
2. the ability to combine kernels for integrating data from different sources where a specific kernel will be used for each individual data source.

Considering these two uses of MKL, we will apply this technique in two well known bioinformatics problems to improve their performance than existing system. In both applications, SVM will be used as base learner for MKL.

1.2.1 Problem Statement for Protein Subcellular Localization Prediction

Most of the protein subcellular localization prediction system has been developed by using sequence information. Since most of the machine learning algorithm can handle vector data (i.e. real valued data), various scientist invent various types of feature extraction techniques to extract real valued feature from protein sequence. In this context, some prediction systems use one feature representation to develop protein subcellular localization prediction, and some other use a combination of these feature representations. However, in order to meet the current demand to produce efficient high-throughput tools for protein subcellular localization prediction, additional effort are required to enrich the prediction quality. In this thesis, we will develop protein subcellular localization prediction using MKL based SVM in the following two situations in order to improve the performance than other existing system.

1. When we will use single source of information, in that case, choice of kernel problem will be solved by MKL. Herein, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) will considered as the search space of the choice of kernel problem.
2. MKL will be used to fuse different sources of information.

Moreover, we will also implement some prediction systems to show the power of the single kernel based SVM.

1.2.2 Problem Statement for Post-translational Modifications Prediction

There are various types of post-translational modifications exist in protein. Among them, carbonylation, succinylatyion and phosphorylation PTM are very important in controlling various functions of protein. In this thesis, we will focus on these PTMs. Here, we will try to develop two different types of prediction systems. One is for post-transtional modification sites prediction where predictor will predict in which site can be modified. In this type of prediction, protein will be fragmented with a predefined window size by keeping the modified residue at the center position. After that, these fragments, called peptide samples, will be used to extract features in order to train and test the predictor. Another one is for predicting whether an uncharacterized protein can be modified or not. If yes, then which types of modification will occur in that protein. In this type of prediction, the whole protein sequence will be used to extract feature in order to train and test the predictor.

However, both problem will handle sequence information in order to predict post-translational modifications. Like, protein subcellular localization prediction, in this case, various feature extraction techniques are available. Our hypothesis is by fusing various features using MKL will improve the performance of the prediction system than other existing system. Moreover, we will also implement some prediction systems to show the power or capability of the single kernel based SVM.

1.3 Contributions

1.3.1 Core Contributions

The core contribution of this these will be shown from four different aspects or categories. We will perform nine experiments in these four aspects. These aspects are defined and discussed below-

1.3.1.1 Category 1: Show the Power or capability of Single Kernel Based SVM

In this aspect or category, we will show the power of SVM with single kernel by developing some prediction systems in the field of protein subcellular localization prediction and post-translational modification prediction. We will perform four experiments or develop four prediction systems. However to develop all of the prediction systems in these category, we will use grid search method to find the kernel parameter (choice of kernel among the set of RBF kernels) of the predictors. In addition, we will use heuristic approach to choose RBF kernel in these prediction systems.

Experiment No 1:- predMultiLoc-Gneg: Predicting Subcellular Localization of Gram-Negative Bacterial Proteins Using Feature Selection in Gene Ontology Space and Support Vector Machine with Resolving the Data Imbalanced Issue

Contribution

- Show the power of SVM with single kernel in protein subcellular localization prediction.
- Providing better accuracy by solving imbalance dataset issue.
- Develop prediction system considering multi-label issue.
- Establish a web server for public use.

Experiment No 2:- predCar-Site: Carbonylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

Contribution

- Show the power of SVM with single kernel in carbonylation sites prediction.
- Providing better accuracy by solving imbalance dataset issue.
- Establish a web server for public use.

Experiment No 3:- predSucc-Site: Lysine Succinylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

Contribution

- Show the power of SVM with single kernel in succinylation sites prediction.
- Providing better accuracy by solving imbalance dataset issue.
- Establish a web server for public use.

Experiment No 4:- mLysPTMpred: Multiple Lysine PTM Site Prediction Using Combination of SVM Classifier with Resolving Data Imbalanced Issue

Contributions

- Show the power of SVM with single kernel in multiple lysine PTM site prediction.
- Providing better accuracy by solving imbalance dataset issue.
- Solve multi-label prediction issue using combination of SVM.
- Establish a web server for public use.

1.3.1.2 Category 2: Show Choice of Kernel Effects for Single Kernel Based SVM

In this aspect, we will show how proper kernel selection affects the performance of a prediction system. We will perform one experiment here. In this experiment, we consider four different types of kernels.

Experiment No 5:- Protein Subcellular Localization Prediction using Support Vector Machine with the Choice of Proper Kernel

Contributions

- Show how choice of kernel problem affects the performance of a single kernel based SVM in protein subcellular localization prediction.
- Develop prediction system considering multi-label issue.

1.3.1.3 Category 3: Use MKL as a Solution for Choice of Kernel Problem

In this aspect, we will use multiple kernel learning (MKL) to solve kernel selection problem. We will also compare the runtime performance of single kernel based SVM and MKL based SVM. We will perform one experiment here. In this experiment, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) will be considered as the search space of the choice of kernel problem.

Experiment No 6:- Protein Subcellular Localization Prediction Using Multiple Kernel Learning Based Support Vector Machine

Contributions

- Show how choice of kernel problem will be solved using MKL in protein subcellular localization prediction.
- Providing better performance over single kernel based SVM and other top systems.
- Providing less run time than single kernel based SVM and other top systems.

1.3.1.4 Category 4: Use MKL to Fuse Multiple Data Sources

Finally, we will use MKL in order to fuse multiple data sources. We will do three experiments in this aspect. One in the field of protein subcellular localization prediction. Other two in the field of protein post-translational modification prediction.

Experiment No 7:- Protein Subcellular Localization Prediction Using Kernel Based Feature Fusion

Contributions

- MKL will be used to integrate multiple data sources in predicting protein subcellular localization.
- Providing better performance than existing systems.
- Develop prediction system considering multi-label issue.
- Providing better accuracy by solving imbalance dataset issue.

Experiment No 8:- predHumPhos: Predicting Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

Contributions

- MKL will be used to integrate multiple data sources in predicting human phosphorylated proteins. Here, single label issue (binary classification) will be considered, i.e., the prediction will be whether a protein can be phosphorylated or not.
- Providing better performance than existing systems.
- Providing better accuracy by solving imbalance dataset issue.
- Establish a web server for public use.

Experiment No 9:- iMulti-HumPhos: A Multi-Label Classifier For Identifying Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

Contributions

- MKL will be used to integrate multiple data sources in predicting human multi-label phosphorylated proteins.
- Providing better performance than existing systems.
- Providing better accuracy by solving imbalance dataset issue.
- Develop prediction system considering multi-label issue.
- Establish a web server for public use.

1.3.2 Related Contributions

Not all of the my works has been included in this thesis during my PhD time. In this thesis, we mainly focus on the application of MKL in bioinformatics. Beside the core applications area, We have done some experiments in some other areas, namely intrusion detection. Intrusion detection is an important task in computer network security. We have implemented some intrusion detection system using support vector machine. In those systems, we have also shown that kernel selection is a critical question for kernel methods like SVM. Only list of publications of these systems have been included in this thesis in the publications section.

1.4 Organization of the Dissertation

This work is oriented forwards diverse audiences in machine learning and data mining, especially for those who are interested in kernel methods, kernel learning and its applications in bioinformatics. In the following we will outline the thesis and give an overview of the content of the individual chapters and their contributions. This thesis is divided into two parts. The first part is on the topic of machine learning and more specifically about multiple kernel learning algorithms. The second part of the thesis

demonstrate how the MKL described in the first part can be put into practice for some applications in bioinformatics. Hence the first part is more focused on algorithmic and theoretical questions, while the second part is concerned with practical problems, and as such, contains a larger experimental part.

Chapter 2 will introduce the basic concepts of claissification problem, kernel methods, SVM and MKL. This will lay the foundation for chapter 4 in which we will describe ways of how to use kernel methods in experiments or applications. We start in section 2.1 and 2.2 with a basic introduction of classification problems and kernels methods especially support vector machine. In Section 2.3 we state various types of MKL algorithm. For more details, please refer to references therein.

Chapter 3 will introduce basic concepts of bioinformatics. In section 3.2, some basic concepts of molecular biology will be described. After that, some problems of bioinformatics will be listed and then where MKL can be applied will also be discussed. Section 3.6 will be dedicated to a specific problem of bioinformatics named protein subcellular localization prediction. Another problem termed protein post-translational modification prediction will described in section 3.7.

Chapter 4 will discuss the implementation and analysis of our systems. It will discuss nine types of prediction systems in the field of protein subcellular localization prediction and post-translational modifications prediction by considering various objectives.

Chapter 5 will present conclusion and future work of this thesis work.

Finally, list of publications as the outcome so far found from this thesis work will be enclosed.

Chapter 2

Kernel Methods and Multiple Kernel Learning

2.1 Learning Functions from Data

The goal of a machine learning algorithm is to synthesize functional relationships on the basis of the data. A learning algorithm is a rule that associates a dataset $\mathcal{D}$ with a function $g : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are called input set and output set, respectively. Depending on the structure of the dataset, learning problems can be classified as being supervised, unsupervised, or semi-supervised [56].

2.1.1 Supervised Learning

The dataset for a supervised learning problem is also called training set, and is made of a finite-number of input-output pairs $(x_i, y_i) \in (\mathcal{X} \times \mathcal{Y})$:

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_l, y_l)\}$$

The framework of supervised learning describes a scenario in which a supervisor shows a set of examples to a learner. There's a training phase in which the learner infers a functional relationship between inputs and outputs using a supervised learning algorithm. The learned relationship can be used during a test phase to make predictions over new inputs, possibly not present in the training set.

Depending on the structure of the output set Y, one can distinguish between two types of supervised learning problems: classification and regression.

2.1.2 Unsupervised Learning

Diffeerently from supervised learning problems, where samples of both inputs and outputs are given, datasets for unsupervised learning problems are made only of inputs $x_i \in \mathcal{X}$:

$$\mathcal{D} = \{x_1, \ldots, x_l\}$$

The goal is to learn a functional relationship $g : \mathcal{X} \to \mathcal{Y}$, where even the structure of the output set Y may possibly be learned from the data. Examples of unsupervised learning problems are clustering and dimensionality reduction.

2.1.3 Semi-supervised Learning

Semi-supervised learning problems falls in between supervised and unsupervised learning problems, and are characterized by the presence of both labeled and unlabeled examples in the dataset, namely:

$$\mathcal{D} = \mathcal{D}_L \cup \mathcal{D}_U, \qquad \mathcal{D}_L = \{(x_1, y_1), \ldots, (x_l, y_l)\}, \qquad \mathcal{D}_U = \{x_{l+1}, \ldots, x_{l+l_U}\}$$

Semi-supervised problems are motivated by situations in which obtaining output labels is costly, whereas unlabeled examples abound. Similarly to the supervised case, $\mathcal{D}$ is called training set, and the goal is to solve classification or regression problems. Under certain conditions, the availability of unlabeled examples can bring considerable benefit to the learning performances.

2.2 Kernel Methods

Kernel-based learning methods (in short, kernel methods) use kernel or kernel matrix as their input in the learning process, not the values of input instances themselves. They work by embedding the data into a Hilbert space and searching for linear relations in such a space [57]. The embedding space is usually characterized via a kernel function, or in short, kernel.

While SVMs play a central role in kernel methods, there are several other popular kernel methods which are also widely used [58], such as Kernel Target Alignment, kernelized version of several methods such as Kernel Fisher Discriminant analysis, kernelized metric learning algorithms [45]. In principle, any learning algorithms can become kernel methods, i.e being kernelized, if they can be extended to work in some kernel space; their optimization problems can be formulated in a way that only the inner product of instances are required but not the explicit feature representations of instances.

2.2.1 Suport Vector Machine (SVM)

Consider the problem of separating the set of training vectors belong to two linear separate classes, $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where $x_i \in R^n$, $y_i \in \{-1, +1\}$ with a hyperplane $w^T x + b = 0$. The SVM modeling algorithm finds an optimal hyperplane with the maximal margin to separate two classes, which requires solving the following constraint problem [59, 60]:

find w and b such that
$w^T x_i + b \geq 1 \; if \; y_i = +1$
$w^T x_i + b \leq -1 \; if \; y_i = -1$

Considering the maximum margin classifier, there is hard margin SVM, applicable to a linearly separable dataset, and then modifies it to handle non-separable data. This leads to the following constrained optimization problem:

$$\begin{aligned} \underset{w,b}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1, \; i = 1, \ldots, n \end{aligned} \tag{2.1}$$

The constraints in this formulation ensure that the maximum margin classifier classifies each example correctly, which is possible since we assumed that the data is linearly separable. In practice, data is often not linearly separable and in that case, a greater margin can be achieved by allowing the classifier to misclassify some points. To allow errors, the optimization problem now becomes:

$$\begin{aligned} \underset{w,b}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 + C\sum_{1=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i, \; i = 1, \ldots, n \\ & \xi_i \geq 0, \; i = 1, \ldots, n \end{aligned} \tag{2.2}$$

The constant $C > 0$ sets the relative importance of maximizing the margin and minimizing the amount of slack. This formulation is called the soft-margin SVM [59, 60]. Using the method of Lagrange multipliers, we can obtain the dual formulation which is expressed in terms of variables α_i [59, 60]:

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{subject to} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C, \; i = 1, \ldots, n \end{aligned} \tag{2.3}$$

The dual formulation leads to an expansion of the weight vector in terms of the input examples:

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

Finally, the linear classifier based on a linear discriminant function takes the following form

$$f(x) = \sum_{i}^{n} \alpha_i y_i x_i^T x + b$$

In many applications a non-linear classifier provides better accuracy. The naive way of making a non-linear classifier out of a linear classifier is to map our data from the input space X to a feature space F using a non-linear function $\Phi : X \rightarrow F$. In the space F, the discriminant function is:

$$f(x) = w^T \Phi(x) + b$$

Now, examine what happens when the nonlinear mapping is introduced into equation 2.3. We have to optimize

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \Phi(x_i)^T \Phi(x_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C, \; i = 1, \ldots, n \end{aligned} \tag{2.4}$$

Notice that the mapped data only occurs as an inner product in the objectives. Now, we can apply a little mathematically rigorous magic known as kernels. By Mercer's theorem, we know that for certain mapping $\Phi(x)$ and any two points x_i and x_j, the inner product of the mapped points can be evaluated using the kernel function without ever explicitly knowing the mapping [57]. The kernel function can be defined as

$$k(x_i, x_j) = \Phi(x_i)^T \Phi(x_j)$$

Substituting the kernel in the equation 2.4, the optimization takes the following form:

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq < C, \; i = 1, \ldots, n \end{aligned} \tag{2.5}$$

Finally, in terms of the kernel function the discriminant function takes the following form:

$$f(x) = \sum_{i}^{n} \alpha_i y_i k(x, x_i) + b$$

2.2.1.1 Some Popular Kernel Functions

The definitions of some popular kernel functions are given below:

1. Linear kernel: $K(x_i, x_j) =< x_i, x_j >$
2. Polynomial kernel: $K(x_i, x_j) = (< x_i, x_j > +1)^d$, d is the degree of polynomial.
3. Gaussian kernel: $K(x_i, x_j) = exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$, σ is the width of the function.
4. Laplace kernel: $K(x_i, x_j) = exp(-\frac{\|x_i - x_j\|}{\sigma})$, σ is the width of the function.

2.2.1.2 Multiclass Support Vector Machine

Support vector machines are formulated for two class problems. But because support vector machines employ direct decision functions, an extension to multiclass problems is not straightforward [61, 60]. There are several types of methods available for SVM to handle multiclass problems. Two commonly used methods are One-Vs-All and One-Versus-One approach.

- The One-Vs-All technique is extended from the binary two-class problem to perform classification tasks with $k > 2$ classes. In this approach, the base classifier (in our case - SVM) is trained on K copies of the K-class original training set, with each copy having the K-th label as the positive label, and all other labels as the negative label (combined class). We denote the optimal separating hyperplane discriminating the class j and the combined class as

  $$g^j = x^T \hat{w}^j + \hat{b}^j,\ j = 1, 2, 3, \ldots, k$$

  where the superscript in $\hat{w}^j$ stands for the class which should be separated from the other observations. After finding the all k optimal separating hyperplanes, the final classifier has been defined by

  $$f_k(x) = \underset{j}{\text{argmax}} \left(g^j(x)\right)$$

  In this approach the index of the largest component of the discriminant vector $\left(g^1(x), g^2(x), \ldots, g^k(x)\right)$ is assigned to the vector x. In other words, each input is classified by all k models, and the output is chosen by the model with the highest degree of confidence.

- Another classical approach for multi-class classification is the one-versus-one (1V1) or pairwise decomposition. It evaluates all possible pairwise classifiers and thus induces $n(n-1)/2$ individual binary classifiers.

  Let the decision function for class i against class j, with the maximum margin, be

  $$D_{ij}(x) = w_{ij}^T x + b_{ij}$$

  where w_{ij} is the m-dimensional vector, b_{ij} is a scalar, and $D_{ij}(x) = D_{ji}(x)$.

  For the input vector x, we calculate

  $$D_i(x) = \sum_{j \neq i, j=1}^{n} sign(D_{ij}(x)) \tag{2.6}$$

  and classify x into the class

$$arg \max_{i=1,\ldots,n} D_i(x) \tag{2.7}$$

But if 2.7 is satisfied for plural i's, x is unclassifiable.

2.2.1.3 Multiclass Multi-label Support Vector Machine

Support vector machines are formulated for two class single label problems. An extension to multiclass multi-label problems for SVM is not straightforward [62, 63]. We have followed Binary relevance method (BR) [62] to solve multiclass multi-label problem. Binary relevance method (BR) [62] uses the one-against rest strategy to convert a multi-label problem into several binary classification problems. Given a multi-label dataset with N class labels, BR method trains one classifier for each class label. When training one classifier for each class label, BR method annotates all of the training examples associated with that label as positive examples while all remaining examples are regarded as negative examples [63]. Given a test example, each classifier in BR will output a prediction score and BR will combine these scores into an N-dimensional score vector, where each score corresponds to a specific class label. The value of the score has two conditions, positive and negative, positive means the binary classifier predicts the test example belongs to the corresponding class label, negative means it do not belong to the class label. Note that if all N scores are negative, the class label with the maximum score is assigned to the test example.

In accordance with the method discussed above, in order to predict class-label(s) of datasets containing both single-label and multi-label subject, N independent binary SVMs are trained, one for each class. Then the class-label(s) of the i-th query subject will be predicted as:

$$M^*(x_i) = \cup_{j=1}^{N} \{j : f_j(x_i) > 0\} \tag{2.8}$$

Here, $M^*(x_i)$ is a predicted set that may have one, or more elements, even it can be empty too, which enables us to make multi-label prediction. However, if Eq. 2.8 provides an empty class label, i.e., $M^*(x_i) = \emptyset$, in that case, the number of class-label will be single one and that class-label will be given by

$$M^*(x_i) = arg\ max_{j=1} f_j(x_i) \tag{2.9}$$

2.2.1.4 Parameter Setting

In order to use the SVM for classification, two kinds of parameters have to be determined:

- The regularization parameter C of the SVM
- The kernel and its parameters

A proper choice of these parameters is crucial to the good performance of the algorithm. A standard way to fix parameters is to use cross-validation. Let us denote by a parameter of a kernel to be set, for instance, the width of the Gaussian RBF kernel, and C the parameter of the algorithm. Given specific values of C and γ the k-fold cross-validation error is calculated as follows: first of all, the training set $\mathcal{Z} = \{x_i, y_i\}_{i=1}^n$ is randomly divided into k subsets $\mathcal{Z}_1, \ldots, \mathcal{Z}_k$ of approximately equal size. The SVM is trained on $k-1$ subsets and its error rate on the remaining subset is computed. Repeating this process k times such that each subset is tested once, the cross-validation error is determined by the average of the test errors. When $k = n$, the cross-validation error is especially called the leave-one-out error.

In this scheme C and γ are determined so as to minimize the cross-validation error. This goal is approximately achieved by a Grid search. A set of candidate values are chosen both for C and γ and the cross-validation error is computed for every possible combination of them. If nc and n are the number of candidate values for the of C and σ respectively, then the cross-validation error is computed $nc * n$ times, which means that the SVM is trained $k * nc * n$ times in total. Typically, users do not have any idea about the optimal values for C and γ , so the candidate values must cover a very large domain. Hsu et al. [64] suggest the candidate values be determined as an exponentially growing sequence (e.g., $C = 2^{-8}, 2^{-7}, \ldots, 2^8, \gamma = 2^{-8}, \ldots, 2^8$). When there are more than two parameters, the grid search becomes difficult as the number of grid points grows exponentially. In such cases, one can use a gradient search to minimize an upper bound on the leave-one-out error [65]. However, this process can be easily parallelized, which alleviates the burden of cross-validation.

On the other hand, a question frequently posed by practitioners is "which kernel should I use for my data?". There are several answers to this question. The first is that it is, like most practical questions in machine learning, data-dependent, so several kernels should be tried. That being said, we typically follow the following procedure: Try a linear kernel first, and then see if we can improve on its performance using a non-linear kernel [66].

2.2.2 Advantages of SVM and Other Kernel Methods

Kernel methods for machine learning have become an attractive alternative to traditional methods in machine learning for four main reasons: First, if the feature space is rich enough, then simple linear estimators with decision functions such as hyperplanes and half-spaces in feature space may be sufficient. Second, kernels allow us to construct machine learning algorithms in Hilbert space H without explicitly computing the mapping of the input vectors. This makes it possible to kernelize linear algorithms provided that they can be expressed in terms of dot products between the data. Third, there is no need to make any assumptions about the input space X other than for it to be a set. We can construct kernels from any kinds of data and then apply the kernel to the learning algorithm [58]. Kernels act like a transparent bridge between data and learning algorithm. This makes it possible to compute similarity between discrete

objects such as strings, trees and graphs. Four, We can change the kernel or learning algorithm separately. Moreover with the kernel-based method, we can avoid the curse of dimensionality, that means all that we need in kernel-based learning is the kernel matrix, which does not depend on the dimensionality of the instances.

2.2.3 Disadvantages and Limitations of SVM and Other Kernel Methods

Kernel-based methods have a big disadvantage: the efficiency of learning process depends much on the kernels [58]. If we choose appropriate kernel and its parameters, we can get high accuracy, but if we choose bad kernel and its parameters, the accuracy can be cut down dramatically. Choosing an appropriate kernel or a group of appropriate kernels becomes a major problem in kernel-based learning. The traditional approach to choosing the kernel function is to select the best kernel function among a set of candidates, $\{k_1, \ldots, k_r\}$, according to their prediction accuracy on a validation set. This approach is very time consuming. Thus learning kernels comes to the scene and plays an important role in kernel methods.

2.2.4 Kernel Learning

By choosing the kernel and function kernel parameter, we have a rather flexible way to embed prior knowledge into a machine learning algorithm [58]. However, the available prior knowledge might not be sufficient to uniquely determine the best kernel for a given problem among the set of all possible kernels on a given domain $\mathcal{X}$. This observation leads to the idea of learning the kernel function itself from the data, which is motivating a considerable amount of research in recent years.

Kernel learning tries to address the problem of selecting or learning the appropriate kernel(s) for a given problem. Work in kernel learning falls roughly in one of the following categories according to the type of basic kernels that takes place:

1. Multiple Kernel Learning (MKL): methods that learn a combination, usually convex, of kernels from some given finite set of kernels [43, 44, 7, 8]. MKL can also be extended to use in the following two situations:

   (a) Kernel Parameter Learning: methods that learn the parameters of some parametrized kernel family, e.g the degree of polynomial kernels, the width of Gaussian kernels, the parameters of hyperkernels [67, 65, 3, 4].

   (b) Kernel Learning with infinite number of basic kernels: methods that learn a kernel combination from an infinite set of basic kernels or kernel matrices [68].

2. Nonparametric Kernel Learning: methods that learn directly the kernel matrix, without requiring any given basic kernels [69, 70].

Most of the work on kernel learning falls into the first category which is MKL. In the following section, some basic concept and theory of MKL has been discussed.

2.3 Multiple Kernel Learning

The goal of machine learning is to learn unknown concepts from data. But the success of a learning machine crucially depends on the quality of the data representation. At this point, the paradigm of kernel-based learning [57] offers an elegant way for decoupling the learning and data representation processes in a modular fashion. This allows to obtain complex learning machines from simple linear ones in a canonical way. Nowadays, kernel machines are frequently employed in modern application domains that are characterized by vast amounts of data along with highly non-trivial learning tasks such as bioinformatics or computer vision, for their favorable generalization performance while maintaining computational feasibility [5].

However, after more than a decade of research it still remains an unsolved problem for kernel methods to find the best kernel for a task at hand. Most frequently, the kernel is selected from a candidate set according to its generalization performance on a validation set, which is held back at training time. Clearly, the performance of such an algorithm is limited by the performance of the best kernel in the set and can be arbitrarily bad if the kernel does not match the underlying learning task. Unfortunately, in the current state of research, there is little hope that in the near future a machine will be able to automatically engineer the perfect kernel for a particular problem at hand [71].

A first step towards a more realistic model of learning the kernel was achieved in Lanckriet et al. [43], who showed that, given a candidate set of kernels, it is computationally feasible to simultaneously learn a support vector machine and a linear kernel combination at the same time, if the so-formed kernel combinations are required to be positive-definite and trace-norm normalized. This framework was entitled multiple kernel learning (MKL).

Multiple kernel learning (MKL) provides a systematic approach to using data to learn the most suitable kernel function for a learning task [4]. These methods are usually designed to combine kernels from a given set. Since a kernel function corresponds to a notion of similarity between data instances, combining multiple kernels can be interpreted as combining different notions of similarity.

The more common implementation of combined kernel for MKL is

$$k_\eta(x_i, x_j) = f_\eta(\{k_m(x_i^m, x_j^m)\}_{m=1}^P|\eta)$$

where the combination function, $f_\eta : R^P \to R$, can be a linear or a nonlinear function, $\{k_m : R^{D_m} \times R^{D_m} \to R\}_{m=1}^P$ take P feature representations (not necessarily different) of data instances: $x_i = \{x_i^m\}_{m=1}^P$ where $x_i^m \in R^{D_m}$, and D_m is the dimensionality of the corresponding feature representation, finally, η parameterizes the combination function. For example, we can linearly parameterize the combination function as

$$k_\eta(x_i, x_j) = f_\eta\left(k_m\{x_i^m, x_j^m\}_{m=1}^P|\eta\right) = \sum_{m=1}^P \eta_m k_m(x_i^m, x_j^m)$$

2.3.1 Key Properties of Multiple Kernel Learning

We identify and explain nine key properties of the existing MKL algorithms in order to obtain a meaningful categorization. We can think of these nine dimensions (though not necessarily orthogonal) defining a space in which we can situate the existing MKL algorithms and search for structure (i.e., groups) to better see the similarities and differences between them.

2.3.1.1 The Learning Method

The existing MKL algorithms use different learning methods for determining the kernel combination function. We basically divide them into five major categories [10]:

1. Fixed rules
2. Heuristic approaches
3. Optimization approaches
4. Bayesian approaches
5. Boosting approaches

2.3.1.2 The Functional Form

There are different ways in which the combination can be done and each has its own combination parameter characteristics. We group functional forms of the existing MKL algorithms into three basic categories [10]:

1. Linear combination
2. Nonlinear combination
3. Data-dependent combination

2.3.1.3 The Target Function

We can optimize different target functions when selecting the combination function parameters. We group the existing target functions into three basic categories [10]:

1. Similarity-based functions calculate a similarity metric between the combined kernel matrix and an optimum kernel matrix calculated from the training data and select the combination function parameters that maximize the similarity. The similarity between two kernel matrices can be calculated using kernel alignment, Euclidean distance, Kullback-Leibler (KL) divergence, or any other similarity measure.

2. Structural risk functions follow the structural risk minimization framework and try to minimize the sum of a regularization term that corresponds to the model complexity and an error term that corresponds to the system performance.

3. Bayesian functions measure the quality of the resulting kernel function constructed from candidate kernels using a Bayesian formulation.

2.3.1.4 The Training Method

We can divide the existing MKL algorithms into two main groups in terms of their training methodology [10]:

1. One-step methods calculate both the combination function parameters and the parameters of the combined base learner in a single pass. One can use a sequential approach or a simultaneous approach.

2. Two-step methods use an iterative approach where each iteration, first we update the combination function parameters while fixing the base learner parameters, and then we update the base learner parameters while fixing the combination function parameters. These two steps are repeated until convergence.

2.3.1.5 The Base Learner

There are many kernel-based learning algorithms proposed in the literature and all of them can be transformed into an MKL algorithm, in one way or another [10]. The most commonly used base learners are SVM and support vector regression (SVR), due to their empirical success, their ease of applicability as a building block in two-step methods, and their ease of transformation to other optimization problems as a one-step training method using the simultaneous approach. Kernel Fisher discriminant analysis (KFDA), regularized kernel discriminant analysis (RKDA), and kernel ridge regression (KRR) are three other popular methods used in MKL. Multinomial probit and Gaussian process (GP) are generally used in Bayesian approaches.

2.3.1.6 The Computational Complexity

The computational complexity of an MKL algorithm mainly depends on its training method (i.e., whether it is one-step or two-step) and the computational complexity of its base learner [10].

Some of the important modeling algorithm for one step method are semidefinite programming (SDP) problem, a quadratically constrained quadratic programming (QCQP) problem, and second-order cone programming (SOCP) problem. Two-step methods can be modeled as a semi-infinite linear programming (SILP) problem, which uses a generic linear programming (LP) solver and a canonical SVM solver in the inner loop.

2.3.1.7 Set of kernels

Instead of selecting kernels from a predefined finite set, we can increase the number of candidate kernels in an iterative manner. We can basically select kernels from an uncountably infinite set constructed by considering base kernels with different kernel parameters [72]. Gehler and Nowozin [72] propose a forward selection algorithm that finds the kernel weights for a fixed size of candidate kernels using one of the methods described above, then adds a new kernel to the set of candidate kernels, until convergence.

2.3.1.8 Constraints on Kernel Combination Weights

L_1-norm of the kernel weights, also known as the simplex constraint, is mostly used in MKL methods [73]. The advantage of the simplex constraint is that it leads to a sparse solution, i.e., only a few base kernels among manys carry significant weights. However, as argued in [74], the simplex constraint may discard complementary information when base kernels encodes orthogonal information, leading to suboptimal performance. To improve the accuracy in this scenario, an L_2-norm of the kernel weights, known as a ball constraint, is introduced in their work. A natural extension to the L_2-norm is the Lp-norm, which is approximated by the second order Taylor expansion and therefore leads to a convex optimization problem [75].

2.3.1.9 Generalized Performance Measure

Multiple kernel learning searches for a combination of base kernel functions/matrices that maximizes a generalized performance measure [73]. Typical measures studied for multiple kernel learning, include maximum margin classification errors [43, 6, 76, 77], kernel-target alignment [45], Fisher discriminative analysis [78], etc.

MKL is an important extension of support vector machines (SVM) for handling multiple information sources [79]. By predefining one (or multiple in general) "base" kernel function for each source, MKL aims to find the optimal linear combination weights of these kernels by maximizing classification-performance-related criteria such as the margin of two classes. One of the representative algorithms is SimpleMKL [8].

Recent research [80] proposes to use more sophisticated criteria to optimize the kernel weights. In addition to the margin of two classes, these criteria consider the radius of minimum enclosing ball (MEB) of training data. The logic lies at that the radius affects the generalization performance of SVM and it varies with the kernel weights. Hence, this radius shall be considered when seeking the optimal weight values. One of the representative algorithm termed 'radius-incorporated MKL' has been developed by considering both margin and radius [80].

2.3.2 Multiple Kernel Learning Algorithm

Most of the existing MKL methods use the SVM objective functions and try to find a linear combination of basic kernels such that the separating margin between the classes is maximized. There are some other base learners available for MKL besides SVM that are dicussed in section 2.3.1.5.

In this work, we have considered SVM objective functions as base learner for MKL.

2.3.2.1 Fixed Rules

Fixed rules obtain $k_\eta(\cdot,\cdot)$ using $f_\eta(\cdot)$ and then train a canonical kernel machine with the kernel matrix calculated using $k_\eta(\cdot,\cdot)$. For example, we can obtain a valid kernel by taking the summation or multiplication of two valid kernels [81]:

$$k_\eta(x_i, y_i) = k_1(x_i^1, x_j^1) + k_2(x_i^2, x_j^2)$$
$$k_\eta(x_i, y_i) = k_1(x_i^1, x_j^1) k_2(x_i^2, x_j^2)$$

We can apply the rules of the above equations recursively to obtain the rules for more than two kernels. For example, the summation or multiplication of P kernels is also a valid kernel:

$$k_\eta(x_i, y_i) = \sum_{m=1}^{P} k_m(x_i^m, x_j^m)$$
$$k_\eta(x_i, y_i) = \prod_{m=1}^{P} k_m(x_i^m, x_j^m)$$

Pavlidis et al. [82] report that on a gene functional classification task, training an SVM with an unweighted sum of heterogeneous kernels gives better results than the combination of multiple SVMs each trained with one of these kernels.

2.3.2.2 Heuristic Approaches

De Diego et al. [83] propose a matrix functional form of combining kernels:

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m(x_i, x_j) k_m(x_i^m, x_j^m)$$

where $\eta_m(\cdot,\cdot)$ assigns a weight to $k_m(\cdot,\cdot)$ according to x_i and x_j. They propose different heuristics to estimate the weighing function values using conditional class probabilities, $Pr(y_i = y_j|x_i)$ and $Pr(y_j = y_i|x_j)$, calculated with a nearest-neighbor approach [10]. However, each kernel function corresponds to a different neighborhood and $\eta_m(\cdot,\cdot)$ is calculated on the neighborhood induced by $k_m(\cdot,\cdot)$. For an unlabeled data instance x, they take its class label once as $+1$ and once as -1, calculate the discriminant values $f(x|y = +1)$ and $f(x|y = -1)$, and assign it to the class that has more confidence in its decision (i.e., by selecting the class label with greater $yf(x|y)$ value).

We can also use a linear combination instead of a data-dependent combination and formulate the combined kernel function as follows:

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i^m, x_j^m)$$

where we select the kernel weights by looking at the performance values obtained by each kernel separately. For example, Tanabe et al. [84] propose the following rule in order to choose the kernel weights for classification problems:

$$\eta_m = \frac{\pi_m - \delta}{\sum_{h=1}^{P}(\pi_h - \delta)}$$

where π_m is the accuracy obtained using only K_m, and δ is the threshold that should be less than or equal to the minimum of the accuracies obtained from single-kernel learners. Qiu and Lane [85] propose two simple heuristics to select the kernel weights for regression problems:

$$\eta_m = \frac{R_m}{\sum_{h=1}^{P} R_h} \qquad \forall m$$

$$\eta_m = \frac{\sum_{h=1}^{P} M_h - M_m}{(P-1)\sum_{h=1}^{P} M_h} \qquad \forall m$$

where R_m is the Pearson correlation coefficient between the true outputs and the predicted labels generated by the regressor using the kernel matrix K_m, and M_m is the mean square error generated by the regressor using the kernel matrix K_m.

Cristianini et al. [45] define a notion of similarity between two kernels called kernel alignment. The empirical alignment of two kernels is calculated as follows:

$$A(K_1, K_2) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}}$$

where $\langle K_1, K_2 \rangle_F = \sum_{i=1}^{N}\sum_{j=1}^{N} k_1(x_i^1, x_j^1) k_2(x_i^2, x_j^2)$. This similarity measure can be seen as the cosine of the angle between K_1 and K_2. yy^T can be defined as ideal kernel for a binary classification task, and the alignment between a kernel and the ideal kernel becomes

$$A(K, yy^T) = \frac{\langle K, yy^T \rangle_F}{\sqrt{\langle K, K \rangle_F \langle yy^T, yy^T \rangle_F}} = \frac{\langle K, yy^T \rangle_F}{N\sqrt{\langle K, K \rangle_F}}$$

Kernel alignment has one key property due to concentration (i.e., the probability of deviation from the mean decays exponentially), which enables us to keep high alignment on a test set when we optimize it on a training set [10].

Qiu and Lane [85] propose the following simple heuristic for classification problems to select the kernel weights using kernel alignment:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P}(K_h, yy^T)} \qquad \forall m$$

where we obtain the combined kernel as a convex combination of the input kernels.

2.3.2.3 Similarity Measure Optimizing Approaches

Linear Approaches with Arbitrary Kernel Weights

Lanckriet et al. [43] propose to optimize the kernel alignment as follows:

$$\begin{array}{ll} \text{maximize} & A(K_\eta^{tra}, yy^T) \\ \text{with respect to} & K_\eta \in S^N \\ \text{subject to} & tr(K_\eta) = 1 \\ & K_\eta \succeq 0. \end{array} \tag{2.10}$$

where the trace of the combined kernel matrix is arbitrarily set to 1, S^N means real symmetric $N \times N$ matrices.

Cortes et al. [44] give a different kernel alignment definition, which they call centered-kernel alignment. The empirical centered-alignment of two kernels is calculated as follows:

$$CA(K_1, K_2) = \frac{\langle K_1^c, K_2^c \rangle_F}{\sqrt{\langle K_1^c, K_1^c \rangle_F \langle K_2^c, K_2^c \rangle_F}}$$

where K^c is the centered version of K and can be calculated as

$$K^c = K - \frac{1}{N}\mathbf{1}\mathbf{1}^T K - \frac{1}{N} K \mathbf{1}\mathbf{1}^T + \frac{1}{N^2}(\mathbf{1}^T K \mathbf{1})\mathbf{1}\mathbf{1}^T$$

where $\mathbf{1}$ is the vector of ones with proper dimension. Cortes et al. [44] also propose to optimize the centered-kernel alignment as follows:

$$\begin{array}{ll} \text{maximize} & CA(K_\eta, yy^T) \\ \text{with respect to} & \eta \in \mathcal{M} \end{array} \tag{2.11}$$

where $\mathcal{M} = \{\eta : \|\eta\|_2 = 1\}$. This optimization problem 2.11 has an analytical solution:

$$\eta = \frac{M^{-1}a}{\|M^{-1}a\|_2}$$

where $M = \{\langle K_m^c, K_h^c \rangle_F\}_{m,h=1}^P$ and $a = \{\langle K_m^c, yy^T \rangle_F\}_{m=1}^P$

Linear Approaches with Nonnegative Kernel Weights

Kandola et al. [86] propose to maximize the alignment between a nonnegative linear combination of kernels and the ideal kernel. The alignment can be calculated as follows:

$$A(K_\eta, yy^T) = \frac{\sum_{m=1}^P \eta_m \langle K_m, yy^T \rangle_F}{N\sqrt{\sum_{m=1}^P \sum_{h=1}^P \eta_m \eta_h \langle K_m, K_h \rangle_F}}$$

We should choose the kernel weights that maximize the alignment and this idea can be cast into the following optimization problem:

$$\begin{aligned} &\text{maximize} && A(K_\eta, yy^T) \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \end{aligned} \tag{2.12}$$

where $\mathbb{R}_+$ means non-negative real numbers.
and the above problem is equivalent to

$$\begin{aligned} &\text{maximize} && \sum_{m=1}^{P} \eta_m \langle K_m, yy^T \rangle_F \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \\ &\text{subject to} && \sum_{m=1}^{P} \sum_{m=1}^{P} \eta_m \eta_h \langle K_m, K_h \rangle_F = c \end{aligned} \tag{2.13}$$

Lanckriet et al. [43] restrict the kernel weights to be nonnegative and their SDP formulation reduces to the following QCQP problem:

$$\begin{aligned} &\text{maximize} && \sum_{m=1}^{P} \eta_m \langle K_m^{tra}, yy^T \rangle_F \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \\ &\text{subject to} && \sum_{m=1}^{P} \sum_{m=1}^{P} \eta_m \eta_h \langle K_m, K_h \rangle_F \leq 1 \end{aligned} \tag{2.14}$$

Cortes et al. [44] also restrict the kernel weights to be nonnegative by changing the definition of $\mathcal{M}$ in 2.11 to $\{\eta : \|\eta\|_2 = 1, \eta \in R_+^P\}$ and obtain the following QP:

$$\begin{aligned} &\text{minimize} && \nu^T M \nu - 2\nu^T a \\ &\text{with respect to} && \nu \in \mathbb{R}_+^P \end{aligned} \tag{2.15}$$

where the kernel weights are given by $\eta = \frac{\nu}{\|v\|_2}$.

Linear Approaches with Kernel Weights on a Simplex

He et al. [87] choose to optimize the distance between the combined kernel matrix and the ideal kernel, instead of optimizing the kernel alignment measure, using the following optimization problem:

$$\begin{aligned} &\text{minimize} && \langle K_\eta - yy^T, K_\eta - yy^T \rangle_F^2 \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \\ &\text{subject to} && \sum_{m=1}^{P} \eta_m = 1 \end{aligned} \tag{2.16}$$

where $\mathbb{R}_+$ means non-negative real numbers.

Nguyen and Ho [88] propose another quality measure called feature space-based kernel matrix evaluation measure (FSM) defined as

$$FSM(K, y) = \frac{s_+ + s_-}{\|m_+ - m_-\|_2}$$

where $\{s_+, s_-\}$ are the standard deviations of the positive and negative classes, and $\{m_+, m_-\}$ are the class centers in the feature space.

Tanabe et al. [84] optimize the kernel weights for the convex combination of kernels by minimizing this measure:

$$\begin{aligned} &\text{minimize} && FSM(K_\eta, y) \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \\ &\text{subject to} && \sum_{m=1}^{P} \eta_m = 1 \end{aligned} \tag{2.17}$$

Again, Ying et al. [89] follow an information-theoretic approach based on the KL divergence between the combined kernel matrix and the optimal kernel matrix:

$$\begin{aligned} &\text{minimize} && KL(N(\mathbf{0}, K_\eta)||N(\mathbf{0}, yy^T)) \\ &\text{with respect to} && \eta \in \mathbb{R}_+^P \\ &\text{subject to} && \sum_{m=1}^{P} \eta_m = 1 \end{aligned} \tag{2.18}$$

where $\mathbf{0}$ is the vector of zeros with proper dimension. The kernel combinations weights can be optimized using a projected gradient-descent method.

2.3.2.4 Structural Risk Optimizing Approaches

Structural risk functions follow the structural risk minimization framework and try to minimize the sum of a regularization term that corresponds to the model complexity and an error term that corresponds to the system performance. Like similarity based approach, deepening on various types of kernel weight, three approaches are popular which are listed below [10]:

1. Linear Approaches with Arbitrary Kernel Weights
2. Linear Approaches with Nonnegative Kernel Weights
3. Linear Approaches with Kernel Weights on a Simplex

2.3.2.5 Nonlinear Approaches to Combine Kernel Functions

Varma and Babu [90] propose a generalized formulation called generalized multiple kernel learning (GMKL) that contains two regularization terms and a loss function in the objective function which uses nonlinear approach to combine kernel functions. This formulation regularizes both the hyperplane weights and the kernel combination weights. At the same time, Cortes et al. [9] develop another type of nonlinear kernel combination method based on KRR and polynomial combination of kernels.

2.3.2.6 Data-Dependent Approaches to Combine Kernel Function

Gönen and Alpaydin [47] propose a data-dependent formulation called localized multiple kernel learning (LMKL) that combines kernels using weights calculated from a gating model. Inspired from LMKL, two methods that learn a data-dependent kernel function are used for image recognition applications in the work of Yang et al. [91, 92]; they differ in their gating models that are constants rather than functions of the input.

2.3.2.7 Radius-Margin based Multiple Kernel Learning

The MKL methods that use the SVM objective function do not exploit the fact that the error bound of SVM depends not only on the separating margin, but also on the radius of the smallest sphere that encloses the data. In fact even the standard SVM algorithms do not exploit the latter, because for a given feature space the radius is fixed. However in the context of MKL the radius is not fixed but is a function of the weights of the basic kernels. Huyen Do et al. [80] propose a novel MKL method that takes account of both radius and margin to optimize the error bound.

2.3.2.8 MKL with Many Kernels

In general MKL algorithm, the number of base kernels K is fixed before runing the optimization program. The size of the number of base kernel is very important in MKL. Afkanpour [3, 4] has developed MKL with many kernels. On the other hand, according to [72], it is an unnecessary restriction. Gehler [72] proposes infinite kernel learning (IKL) by using infinite set of base-kernels. It is an extension of MKL where the number of kernel is infinite. On some datasets, fixed set of base kernel in MKL can decrease the accuracy due to the possibility of using a largely increased kernel set. On some datasets IKL yields massive increases in accuracy over SVM/MKL due to the possibility of using a largely increased kernel set.

2.3.2.9 Bayesian Approaches

Girolami and Rogers [93] formulate a Bayesian hierarchical model and derive variational Bayes estimators for classification and regression problems. The proposed decision function can be formulated as

$$f(x) = \sum_{i=0}^{N} \alpha_i \sum_{m=1}^{P} \eta_m k_m(x_i^m, x^m)$$

where η is modeled with a Dirichlet prior and α is modeled with a zero-mean Gaussian with an inverse gamma variance prior.

2.3.2.10 Boosting Approaches

Inspired from ensemble and boosting methods, Bennett et al. [94] modify the decision function in order to use multiple kernels:

$$f(x) = \sum_{i=1}^{N} \sum_{m=1}^{P} \alpha_i^m k_m(x_i^m, x^m) + b$$

The parameters $\{\alpha_m\}_{m=1}^{P}$ and b of the KRR model are learned using gradient-descent in the function space. The columns of the combined kernel matrix are generated on the fly from the heterogeneous kernels. Bi et al. [95] develop column generation boosting methods for binary classification and regression problems. At each iteration, the proposed methods solve an LP or a QP on a working set depending on the regularization term used [10].

2.4 MKL with Some Other Directions

In theses previuos discussion, we only consider supervised case of MKL. Besides these, some researcher are also developed MKL for unsupervised case [96]. In multiple kernel learning algorithm, we only consider SVM as base leaner for MKL. We can also consider other base learner such as Kernel Fisher discriminant analysis (KFDA), regularized kernel discriminant analysis (RKDA), and kernel ridge regression (KRR)for MKL [10]. In some applications, multilabel muclass classification is required. To handle these problems, [97] developed MKL to solve multilabel problems.

Chapter 3

Applications of MKL in Bioinformatics

3.1 Introduction

Over the past few decades, major advances in the field of molecular biology, coupled with advances in genomic technologies, have led to an explosive growth in the biological information generated by the scientific community. This deluge of genomic information has, in turn, led to an absolute requirement for computerized databases to store, organize, and index the data, and for specialized tools to view and analyze the data [98].

Bioinformatics involve the creation and advancement of algorithms using techniques including computational intelligence, applied mathematics and statistics, informatics, and biochemistry to solve biological problems usually on the molecular level [99]. It is concerned with the use of computation to answer biological questions and to acquire and exploit large-scale biological data. Answering biological questions requires that investigators take advantage of large, complex data sets (both public and private) in a rigorous fashion to reach valid biological conclusions. The potential of such an approach is beginning to change the fundamental way in which basic science is done, helping to more efficiently guide experimental design in the laboratory.

3.2 Basic Concept in Molecular Biology

In this section, we will introduce some basic concepts in molecular biology which are the prerequisites to understand various problems of bioinformatics.

3.2.1 Deoxyribonucleic Acid (DNA)

Deoxyribonucleic acid (DNA) and proteins are biological macromolecules built as long linear chains of chemical components [98]. A DNA strand consists of a large sequence of nucleotides, or bases. For example, there are more than three billion bases in human

DNA sequences. The units of DNA are called nucleotides. One nucleotide consists of one nitrogen base, one sugar molecule (deoxyribose), and one phosphate. Four nitrogen bases are denoted by one of the letters A (adenine), C (cytosine), G (guanine), and T (thymine). A linear chain of DNA is paired to a complementary strand. The complementary property stems from the ability of the nucleotides to establish specific pairs (A-T and G-C). The resulting rope ladder is twisted around an imaginary central axis to form the famous DNA double helix (see Figure 3.1).

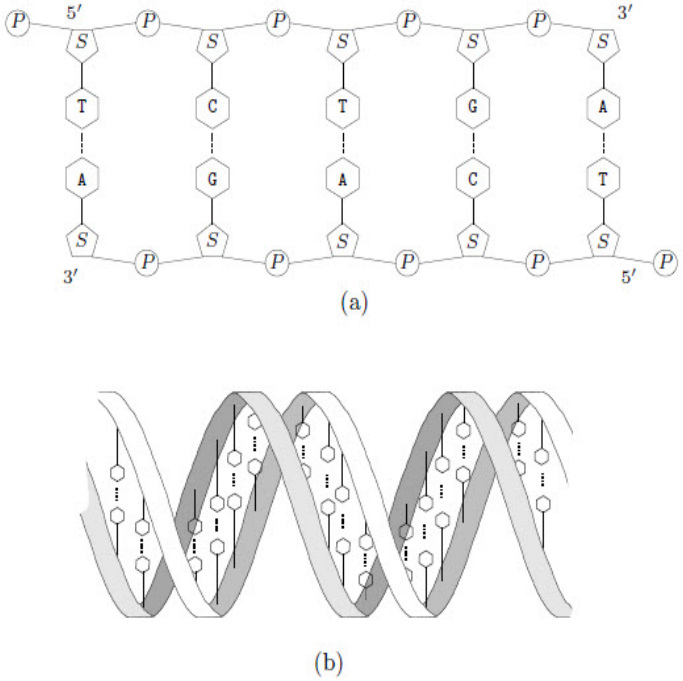

Figure 3.1: Schematic view of a DNA double strand (a) and a DNA double helix (b)
Source: This figure is taken from [100]

3.2.2 Ribonucleic Acid (RNA)

A large variety of different RNA species exist within any cell. All RNAs are transcribed from RNA-encoding genes. The various types of RNA are constructed from the same building blocks but perform different roles in the cell. In light of these different roles, RNAs are divided into two general categories: messenger RNA and functional RNA [101].

Genes transcribing messenger RNA (mRNA) are protein-producing genes, and their transcripts direct protein synthesis by the process of translation. Messenger RNA is the short-lived intermediary form of RNA that conveys the genetic message of DNA to ribosomes for translation. Messenger RNA is the only form of RNA that undergoes translation.

Functional RNAs perform a variety of specialized roles in the cell. The functional RNAs carry out their activities in nucleic acid form and are not translated. Two major categories of functional RNA are active in bacterial and eukaryotic translation: transfer RNA (tRNA) and ribosomal RNA (rRNA).

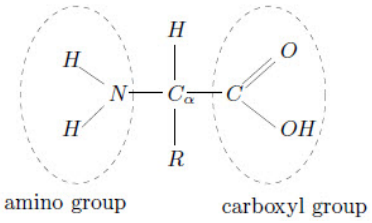

Figure 3.2: Structure of an amino acid consisting of a central carbon C_α together with an amino and a carboxyl group, and a specific side chain R
Source: This figure is taken from [100]

3.2.3 Genomes and Genes

The entire DNA sequence that codes for a living thing is called its genome. The genome doesn't function as one long sequence, however. It is divided into individual genes. A gene is a small defined section of the entire genomic sequence, and each gene has a specific unique purpose [102].

There are three classes of genes [102]. Protein-coding genes are templates for generating molecules called proteins. Each protein encoded by the genome is a chemical machine with a distinct purpose in the organism. RNA-specifying genes are also templates for chemical machines, but the building blocks of RNA machines are different from those that make up proteins. Finally, untranscribed genes are regions of genomic DNA that have some functional purpose but don't achieve that purpose by being transcribed or translated to create another molecule.

3.2.4 Proteins

Proteins represent one of the most important of the molecule classes in living organisms [100]. Their functions include the catalysis of metabolic processes in the form of enzymes; they play an important role in signal transmission, defense mechanisms, and molecule transportation; and they are used as building material, for example in hair. They also function as structural supports, motors that drive movement, pumps, gene regulators and many other things.

Proteins are chains of smaller molecular entities, so-called amino acids, which consist of a central carbon atom, denoted as C_α, connected to an amino group (NH_2), a carboxyl group (COOH), and a side chain (R), which is specific for the particular amino acid. The fourth free binding site of C_α is saturated by a single hydrogen (H) atom. In Figure 3.2, this general structure is shown. According to this, the particular amino acids only differ with respect to their side chains, which also determine their chemical characteristics. We will, however, not consider the detailed chemical structure of the side chains any further.

In nature, there are several known amino acids, but only twenty of them serve as standard building blocks of proteins; these are given in Table 3.1.

Table 3.1: The 20 standard amino acids and their single letter code

name	Alanine	Valine	Leucine	Isoleucine	Phenylalanine
code	A	V	L	I	F
name	Proline	Methionine	Serine	Threonine	Cysteine
code	P	M	S	T	C
name	Tryptophan	Tyrosine	Asparagine	Glutamine	Aspartic acid
code	W	Y	N	Q	D
name	Glutamic acid	Lysine	Arginine	Histidine	Glycine
code	E	K	R	H	G

3.2.5 Molecular Biology's Central Dogma

The central dogma of molecular biology states that [102]:

"DNA acts as a template to replicate itself, DNA is also transcribed into RNA, and RNA is translated into protein".

As it is observed that, the central dogma sums up the function of the genome in terms of information. Genetic information is conserved and passed on to progeny through the process of replication. Genetic information is also used by the individual organism through the processes of transcription and translation. There are many layers of function, at the structural, biochemical, and cellular levels, built on top of genomic information. But in the end, all of life's functions come back to the information content of the genome.

DNA makes RNA and RNA makes proteins, stated by the Central Dogma of Molecular Biology, is explained in Figure 3.3.

The DNA mainly occurs in the nucleus and is not able to leave it. On the other hand, the "manufacturing halls" of proteins, the ribosomes, are located outside the nucleus. The process of transforming genetic information encoded in a sequence of nucleotides into a sequence of amino acids mainly consists of two steps, as shown schematically in Figure 3.3.

3.2.6 Molecular Evolution

Molecular evolution is the process of change in the sequence composition of cellular molecules such as DNA, RNA, and proteins across generations. The field of molecular evolution uses principles of evolutionary biology and population genetics to explain patterns in these changes. Major topics in molecular evolution concern the rates and impacts of single nucleotide changes, neutral evolution vs. natural selection, origins of new genes, the genetic nature of complex traits, evolution of development, and ways that evolutionary forces influence genomic and phenotypic changes.

The study of molecular evolution–how gene sequences change and evolve over time–is an important part of bioinformatics, the study of DNA sequence information.

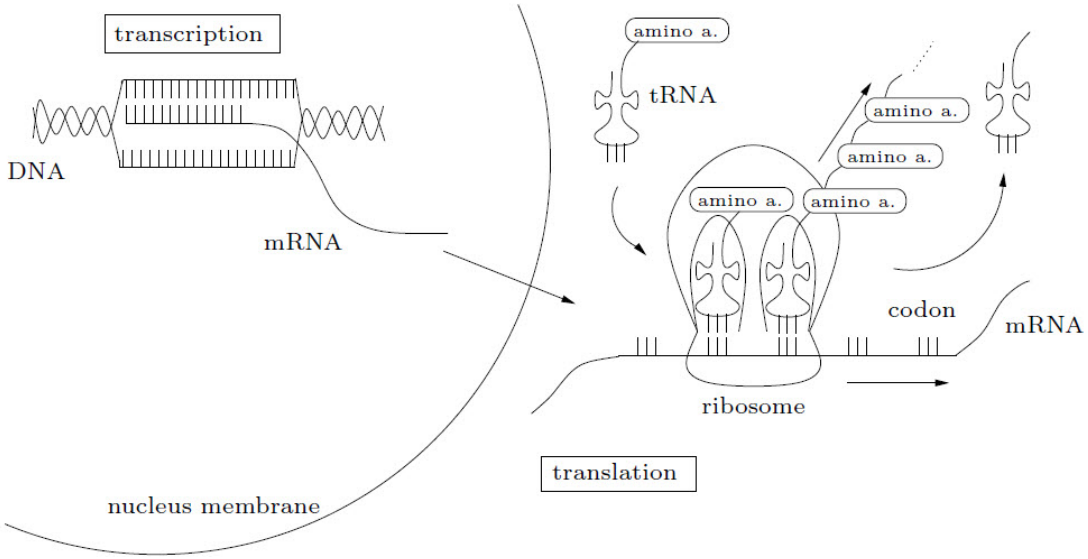

Figure 3.3: Schematic description of transcription and translation
Source: This figure is taken from [100]

Genes that have diverged by mutation from a common ancestor are called homologs.

3.3 Bioinformatics Tasks

Different biological problems can be considered within the scope of bioinformatics. In order to get better understanding, some specific tasks or problems of bioinformatics are listed in below:

- Alignment and comparison of DNA, RNA, and protein sequences
- Gene finding and promoter identification from DNA sequences
- Gene regulatory network identification
- Gene Selection
- Protein Structure Prediction (PSP)
- Microarray Classification
- Protein Subcellular Localization Prediction
- Post-translation Modification Prediction
- Protein-protein interaction prediction
- DNA Fragment Assembly (FA)
- Modeling Biological Systems
- High-throughput Image Analysis

3.4 Aims of Bioinformatics

The increasing availability of annotated genomic sequences has resulted in the introduction of computational genomics and proteomics, large-scale analysis of complete genomes. The primary goal of bioinformatics is to increase the understanding of biological processes. What sets it apart from other approaches, however, is its focus on developing and applying computationally intensive techniques to achieve this goal. Major research efforts in the field include sequence alignment, gene finding, genome assembly, drug design, drug discovery, protein structure alignment, protein structure prediction, prediction of gene expression and protein-protein interactions, genome-wide association studies, the modeling of evolution and cell division/mitosis. Bioinformatics now entails the creation and advancement of databases, algorithms, computational and statistical techniques, and theory to solve formal and practical problems arising from the management and analysis of biological data.

3.5 Uses of MKL in Bioinformatics

MKL has advanced rapidly since 2002 and now demonstrates state-of-the-art performance in various fields. MKL is familiar to do solve the following problem in data analysis:

1. Provide a potential solution to the choice of kernel problem for kernel methods.
2. Data fusion from multiple data sources with different format or same format.

Support vector machine is an important classification methods among all Kernel based methods. The main difficulties of SVM is to choose the proper kernel and its parameter for a given problem. MKL provides a potential solution in this context. This application of MKL can be used in every problems of bioinformatics where SVM as well as kernel methods are applicable.

On the other hand, the second application of MKL requires handling of multiple data sources of a problem. In bioinformatics, many problems require process of multiple data sources. In the following sections we will discuss some problems in bioinformatics that have multiple data sources for analysis:

- **Protein Function Prediction:**

  Protein function prediction can be predicted from various types of data sources [103]. Information may come from nucleic acid sequence homology, gene expression profiles, protein domain structures, text mining of publications, phylogenetic profiles, phenotypic profiles, and protein-protein interaction. Most of the researcher developed protein function prediction using single information sources such as using gene expression profile, or protein-protein interaction feature. Integrating of these features can be an improvement of the prediction process.

- **Protein Subcellular Localization Prediction (SCLP):**

  Protein subcellular localization prediction involves the prediction of where a protein resides in a cell, its subcellular localization. In general, prediction tools take the input information about a protein from its sequence of amino acids [19, 21]. Several different approaches are available to extract feature from protein sequences such as Amino Acid Composition (AAC), Dipeptide Composition (DC), Pseudo-Amino Acid Composition (PAAC), Amphiphilic Pseudo-Amino Acid Composition (APAAC), Physicochemical Properties Model, Amino Acid Index Distribution, GO-based representation, SeqEvo (Sequential Evolution) Formulation, FunD (Functional Domain) Representation, etc. Various types computational methods have been proposed using these feature representations. Each feature provides a different angle to view of a protein, therefore a combination of them can produce better performance.

- **Post-translation Modifications (PTMs) Prediction:**

  Most of PTMs sites predictors or PTM predictors use information from protein sequence [104, 105, 106, 107, 55]. Various types of feature extraction techniques from sequence are also available such as Binary encoding, CKSAAP encoding scheme, Position-specific amino acid propensity (PSAAP), AAIndex property, Amino Acid Occurrence Frequency (AAOF), K Nearest Neighbor Score (KNNS), Encoding based on attribute grouping, Position Weight Amino Acid Composition(PWAAC), Sequence Coupling Model, etc. So, handling different sources combinedly can be a good solution in the improvement of PTM site prediction as well as PTM prediction.

- **Protein-protein Interaction Prediction:**

  Prediction of protein-protein interaction (PPI) is very important for the investigation of intracellular signaling pathways, modelling of protein complex structures and for gaining insights into various biochemical processes. Prediction can be performed using different sources of evidence. Evidence for machine learning includes physical features (such as calculated statistics of hydrophobicity, hydrophilicity, polarizability, etc.) and non-physical features (such as gene co-expression, sequence similarity, function annotation enrichment, etc.). Each feature provides a different angle to view protein interactions and has the potential for uncovering a novel subset of the whole interactome. For this reason, integration of these evidence can strengthen and flourish the study of PPI prediction and also give a proper direction in accurate prediction of PPI prediction.

- **Drug-target Interaction Prediction:**

  Drug-target networks are receiving a lot of attention in late years, given its relevance for pharmaceutical innovation and drug lead discovery. This problem requires handling of multiple data sources such as chemical structures, side-effects, amino acid sequence, biological function, PPI interactions and network topology.

The data integration strategy can be able to improve the quality of the predicted interactions, and can speed up the identification of new drug-target interactions as well as identify relevant information for the task.

Although various problems of bioinformatis require the analysis of heterogeneous data sources. In this work, we have applied MKL in protein subcellular localization prediction (SCLP) and protein post-translational modification site or post-translational modification prediction. In the following section we will describe these problems in details.

3.6 Protein Subcellular Localization

According to the cellular anatomy, most of the functions, which are critical to a cell's survival, are performed by the proteins in the cell [13]. A typical cell contains approximately 1 billion (or 10^9) protein molecules that reside in many different compartments or organelles (an example of a cell is shown in Figure 3.4), usually termed 'subcellular locations'.

It is well established that there is a relation between the function and the subcellular location of a protein. Therefore, one of the fundamental goals in cell biology and proteomics is to identify the subcellular locations and functions of these proteins. Information of the subcellular locations of proteins can provide useful clues about their functions. For understanding the intricate pathways that regulate the biological processes at the cellular level, we also need to know the subcellular distributions of proteins.

3.6.1 Challenges for Prediction of Protein Subcellular Locations

Prediction of protein subcellular locations are very complex task due to the following challenges [15]:

- Informative representations of proteins.
- Performance measurement for multi-label system.
- Determining the number of subcellular locations of a query protein.

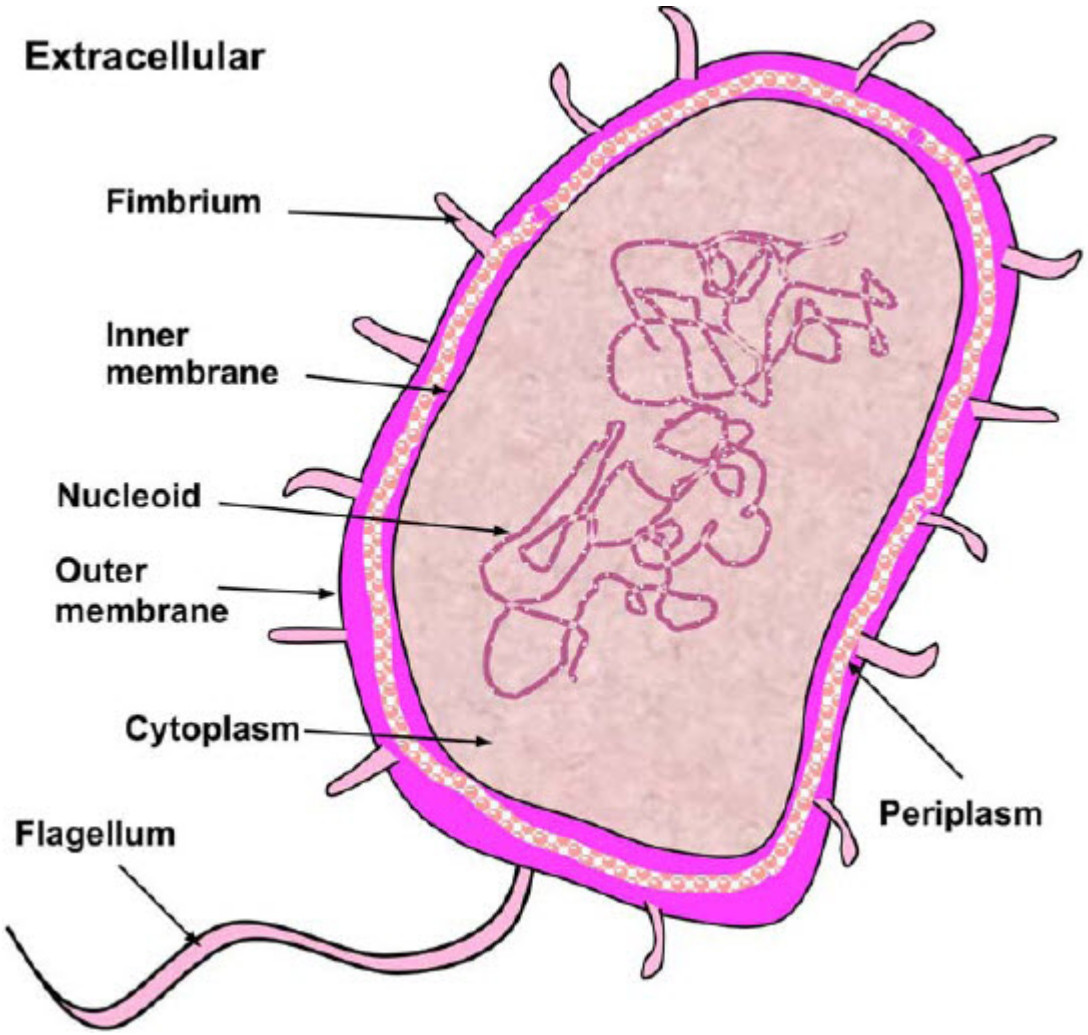

Figure 3.4: Schematic illustration showing many different components or organelles of Gram-negative bacterial cell.
Source: This figure is taken from [29]

3.6.2 Laboratory Experimental Approaches for Determining Protein Subcellular Localtion

A protein's subcellular location has been traditionally determined in the laboratory via techniques suitable for low throughput, single protein analysis only. However, more recently a number of high-throughput identification methods have become available. A list of common experimental approaches will be mentioned below [108]:

- Microscopy-based visualization techniques.
- PhoA fusion technique.
- Subcellular fractionation and two-dimensional (2D) gel electrophoresis.
- Mass spectrometry identification of subcellular fractions.

3.6.3 Importance of Protein Subcellular Localization Prediction (SCLP)

One of the goals of basic microbiology research is to figure out the functions of all genes within a genome, so we can better understand the organism. By determining the SCL of a protein, we can narrow down its potential set of functions. From a more practical point of view, SCL identification and prediction has many useful purposes.

Within microbiology research, subcellular location prediction is also helpful for researchers, who may be working with a gene of unknown function and would like to know its destined subcellular location to facilitate protein extraction or general experimental design and confirmation. Subcellular location prediction is now also routinely used as part of a microbial genome annotation pipeline. There is a growing trend in proteomics projects as well, where subcellular fractions of bacteria are submitted to high-throughput identification. Given there are always possible contaminations and experimental errors, high precision computational subcellular location prediction can serve as a valuable means for evaluation and confirmation purposes.

Knowledge of the subcellular localization of a protein can significantly improve target identification during the drug discovery process. For example, secreted proteins and plasma membrane proteins are easily accessible by drug molecules due to their localization in the extracellular space or on the cell surface.

3.6.4 Methods to Predict Subcellular Localizations and Dataset

Many computational techniques, such as the neural network [34], K-nearest neighbor (KNN) [27, 35, 36], naive Bayes [30] and a few ensemble classifiers [38, 39] have been introduced for the prediction of protein subcellular localization. Some methods use variations of k-NN to predict multiple locations for proteins such as WoLF PSORT [18] uses k-NN with a distance measure that combines Euclidean and Manhattan distances, Euk-mPLoc [28]uses an ensemble of k-NN. In recent days, the support vector machine (SVM) [14, 21, 33, 40] has also been extensively applied to provide potential solutions for the subcellular localization prediction.

Most of the existing computational method developed their system using some specific dataset. Some of the dataset and their location are listed in Table 3.2.

3.6.5 Feature Extraction Techniques for SCLP

A protein or peptide sequence with L amino acid residues could be generally represented as $\{R_1, R_2, \ldots, R_L\}$, where R_i represents the residue at the i-th position in the sequence [109]. However, various non-sequential or discrete models to represent protein samples have been proposed in hopes to establish some sort of correlation or cluster manner through which the prediction could be more effectively carried out. Some of the feature extraction approaches are discussed in the following sections.

Table 3.2: Dataset name and their location of source

Dataset Name	Location
Eukaryotic Proteins	http://www.csbio.sjtu.edu.cn/bioinf/euk-multi/
Plant protein	http://www.csbio.sjtu.edu.cn/bioinf/plant/
Gram-negative Bacteria Protein	http://www.csbio.sjtu.edu.cn/bioinf/Gneg/
Human Proteins	http://www.csbio.sjtu.edu.cn/bioinf/hum-multi/
Viral Proteins	http://www.csbio.sjtu.edu.cn/bioinf/virus/
HÃűglund Dataset	http://abi.inf.uni-tuebingen.de/Services/MultiLoc/multiloc_dataset
DBMLoc Dataset	http://www.bioinfo.tsinghua.edu.cn/DBMLoc/index.htm

3.6.5.1 Amino Acid Composition (AAC)

The Amino Acid Composition (AAC) is the fraction of each amino acid type within a protein [109]. The fractions of all 20 natural amino acids are calculated as:

$$f(r) = \frac{N_r}{N}, \qquad r = 1, 2, \ldots, 20 \tag{3.1}$$

where N_r is the number of the amino acid type r and N is the length of the sequence.

3.6.5.2 Dipeptide Composition (DC)

Dipeptide Composition (DC) represents the fraction of every two consecutive amino acid residues of a protein [109, 110]. The Dipeptide Composition (DC) gives 400 descriptors, defined as:

$$f(r, s) = \frac{N_{rs}}{N - 1} \qquad r, s = 1, 2, \ldots, 20 \tag{3.2}$$

where N_{rs} is the co-occurrence frequency of the dipeptide denoted by amino acid r and type s.

3.6.5.3 Pseudo-Amino Acid Composition (PAAC)

Pseudo-Amino Acid Composition feature extraction method not only based on sequence composition but also its order information [109, 110]. This descriptor is proposed in [110]. Let $H_1^0(i)$, $H_2^0(i)$, $M^0(i)$ $(i = 1, 2, \ldots, 20)$ be the original hydrophobicity values, the original hydrophilicity values and the original side chain masses of

the 20 natural amino acids, respectively. Before their use, they are all subjected to standard conversion as described by the following equation:

$$H_1(i) = \frac{H_1^0(i) - \frac{1}{20}\sum_{i=1}^{20} H_1^0(i)}{\sqrt{\frac{\sum_{i=1}^{20}\left[H_1^0(i) - \frac{1}{20}\sum_{i=1}^{20} H_1^0(i)\right]^2}{20}}} \tag{3.3}$$

$H_2^0(i)$ and $M^0(i)$ are normalized as $H_2(i)$ and $M(i)$ in the same way.

Consider a protein chain of L amino acid residues: $R_1R_2R_3R_4R_5R_6 \cdots\cdots R_L$. A set of descriptors called sequence order-correlated factors are defined as:

$$\begin{aligned}
\theta_1 &= \frac{1}{L-1}\sum_{i=1}^{L-1}\Theta(R_i, R_{i+1}) \\
\theta_2 &= \frac{1}{L-2}\sum_{i=1}^{L-2}\Theta(R_i, R_{i+2}) \\
\theta_3 &= \frac{1}{L-3}\sum_{i=1}^{L-3}\Theta(R_i, R_{i+3}) \\
&\vdots\ \vdots \\
\theta_\lambda &= \frac{1}{L-\lambda}\sum_{i=1}^{L-\lambda}\Theta(R_i, R_{i+\lambda})
\end{aligned} \tag{3.4}$$

$\lambda(\lambda < L)$ is a parameter to be chosen. Here, θ_1 is called the first-tier correlation factor that reflects the sequence order correlation between all the most contiguous residues along a protein chain, θ_2 the second-tier correlation factor that reflects the sequence order correlation between all the second most contiguous residues, and so forth. In Eq. 3.4 the correlation function is given by

$$\Theta(R_i, R_j) = \frac{1}{3}\left\{[H_1(R_i) - H_1(R_j)]^2 + [H_2(R_i) - H_2(R_j)]^2 + [M(R_i) - M(R_j)]^2\right\} \tag{3.5}$$

This correlation function is actually an averaged value for the three amino acid properties: hydrophobicity value, hydrophilicity value and side chain mass.

Let f_i be the normalized occurrence frequency of the 20 amino acids in the protein sequence. Now, the Pseudo-Amino Acid Composition of a protein can be expressed as:

$$X = [x_1, x_2, \ldots, x_{20}, x_{21}, \ldots, x_{20+\lambda}]$$

where

$$x_u = \begin{cases} \frac{f_u}{\sum_{r=1}^{20} f_r + w\sum_{j=1}^{\lambda}\theta_j} & (1 \le u \le 20) \\ \frac{w\theta_u - 20}{\sum_{r=1}^{20} f_r + w\sum_{j=1}^{\lambda}\theta_j} & (21 \le u \le 20+\lambda) \end{cases} \tag{3.6}$$

Here, w is the weighting factor for the sequence-order effect and is set as $w = 0.05$ in our work as suggested by Kuo-Chen Chou [110].

3.6.5.4 Amphiphilic Pseudo-Amino Acid Composition (APAAC)

Amphiphilic Pseudo-Amino Acid Composition (APAAC) was proposed in [110]. The definitions of these qualities are similar to the PAAC descriptors [109, 110]. Based on previous calculation, we have $H_1(i)$ and $H_2(j)$. Then the hydrophobicity and hydrophilicity correlation functions are defined as:

$$H_{ij}^1 = H_1(i)H_1(j)$$

$$H_{ij}^2 = H_2(i)H_2(j)$$

From these qualities, sequence order factors can be defines as:

$$\begin{aligned}
\tau_1 &= \frac{1}{L-1}\sum_{i=1}^{L-1} H_{i,i+1}^1 \\
\tau_2 &= \frac{1}{L-1}\sum_{i=1}^{L-1} H_{i,i+1}^2 \\
\tau_3 &= \frac{1}{L-2}\sum_{i=1}^{L-2} H_{i,i+2}^1 \\
\tau_4 &= \frac{1}{L-2}\sum_{i=1}^{L-2} H_{i,i+2}^2 \\
&\dots \\
\tau_{2\lambda-1} &= \frac{1}{L-\lambda}\sum_{i=1}^{L-\lambda} H_{i,i+\lambda}^1 \\
\tau_{2\lambda} &= \frac{1}{L-\lambda}\sum_{i=1}^{L-\lambda} H_{i,i+\lambda}^2
\end{aligned}$$

Then a set of descriptors called Amphiphilic Pseudo-Amino Acid Composition (APAAC) are defined as:

$$X = [x_1, x_2, \dots, x_{20}, x_{21}, \dots, x_{20+2\lambda}]$$

where

$$x_u = \begin{cases} \frac{f_u}{\sum_{r=1}^{20} f_r + w\sum_{j=1}^{2\lambda}\tau_j} & (1 < u < 20) \\ \frac{w\tau_u}{\sum_{r=1}^{20} f_r + w\sum_{j=1}^{2\lambda}\tau_j} & (21 \le u \le 20+2\lambda) \end{cases} \tag{3.7}$$

where w is the weighting factor and is taken as $w = 0.5$ in our work as suggested by Kuo-Chen Chou [110].

3.6.5.5 Physicochemical Properties Model (PPM)

In Physicochemical Properties Model, amino acid residues in all proteins are divided into neutral, hydrophobic and polar groups according to their seven physicochemical properties. The seven physicochemical properties are hydrophobicity, normalized vander Waals volume, polarity, polarizibility, charge, secondary structures and solvent accessibility. According to the seven properties of amino acids residues, compute occurrence frequency of the residues manifested as polar, neutral and hydrophobic in a protein sequence. The calculation formula is [111]:

$$f_{i,polar} = \frac{n_{polar}}{N} \tag{3.8}$$

$$f_{i,neutral} = \frac{n_{neutral}}{N} \tag{3.9}$$

$$f_{i,hydrophobic} = \frac{n_{hydrophobic}}{N} \tag{3.10}$$

$i = 1, 2, \ldots, 7$ (seven properties), $f_{i,*}$ is the frequency of amino acid characterized by polar / neutral / hydrophobic, n_* represents the total number of polar / neutral / hydrophobic characters present in protein sequence, N is the length of the protein sequence.

So according to this feature extraction model, it creates 21 dimension feature vector for each protein sequence. The distribution situation of amino acids properties are showed in Table 3.3.

In Table 3.3, the 20 capitals denote the 20 amino acids respectively, the left column

Table 3.3: The distribution situation of amino acids properties

Property	Polar	Neutral	Hydrophobic
hydrophobicity	RKEDQN	GASTPHY	CLVIMFW
normalized vander Waals	GASCTPD	NVEQIL	MHKFRYW
polarity	LIFWCMVY	PATGS	HQRKNED
polarizibility	GASDT	CPNVEQIL	KMHFRYW
charge	KR	ANCQGHILMFPSTWYN	DE
secondary structures	EALMQKRH	VIYCWFT	GNPSD
solvent accessibility	ALFCGINW	RKQEND	MPSTHY

shows the seven properties of the amino acids, and the line represents the distribution situation of the 20 amino acids under a property [111].

3.6.5.6 Amino Acid Index Distribution (AAID)

Amino Acid Index Distribution (AAID) considers the physicochemical value and order of amino acids appeared in the protein sequence to express the protein sequence [111]. In this kind of model, the feature vector of the protein sequence can be represented by the following formula [111]:

$$F_{AAID} = [x_1, x_2, \ldots, x_{20}; y_1, y_2, \ldots, y_{20}; z_1, z_2, \ldots, z_{20}] \tag{3.11}$$

Firstly here, the first 20 dimension of vector F_{AAID} is the combination of statistical information and physicochemical values. Let $R_1,\ R_2,\ \cdots, R_{20}$ represent the 20 natural amino acids A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y respectively and $J_i\ (i = 1, 2, \ldots, 20)$ be the amino acid index value of the 20 natural amino acids R_i. Here amino acid index is a set of 20 numerical values representing any of the different physicochemical properties of the 20 amino acids [111]. By considering the physicochemical property of the amino acid, we can define the feature x_i of amino acid

$$x_i = J_i f_i \qquad i = 1, 2, \ldots, 20 \tag{3.12}$$

where f_i is the frequency of the amino acids R_i in the protein sequence, J_i is the physicochemical values of the amino acids R_i which are as follows [111]: $J_1 = 0.486$, $J_2 = 0.2$, $J_3 = 0.288$, $J_4 = 0.538$, $J_5 = 0.318$, $J_6 = 0.12$, $J_7 = 0.4$, $J_8 = 0.37$, $J_9 = 0.402$, $J_{10} = 0.42$, $J_{11} = 0.417$, $J_{12} = 0.193$, $J_{13} = 0.208$, $J_{14} = 0.418$, $J_{15} = 0.262$, $J_{16} = 0.2$, $J_{17} = 0.272$, $J_{18} = 0.379$, $J_{19} = 0.462$, $J_{20} = 0.161$.

Secondly, the following 20 dimension feature vectors is 2-order center distance information, it does not only includes the statistical information and physicochemical values, but also contains position information. The formula is as follows:

$$y_i = \sum_{j=1}^{N_{R_i}} \left(\frac{K_{i,j} - \bar{K_i}}{L} * J_i \right)^2 \tag{3.13}$$

where N_{R_i} is the total number of amino acid R_i appearing in the protein sequence P, $K_{i,j}$ is the j^{th} position of the amino acid R_i in the sequence, and $\bar{K_i}$ is the mean of the position of amino acid R_i.

Now feature y_i contains the physicochemical information, statistical information and the sequence-order information of amino acid R_i, but it still does not distinguish the protein pairs in some cases [111]. To solve this problem, the 3^{rd} order center distance z_i of amino acid R_i was introduced, which is defined as

$$z_i = \sum_{j=1}^{N_{R_i}} \left(\frac{K_{i,j} - \bar{K_i}}{L} * J_i \right)^3 \tag{3.14}$$

The 3^{rd} order center distance information is almost same as 2^{nd} order center distance except the order number. In our work, we have used the first 40 dimension vectors.

3.6.5.7 GO-based Representation

The GO-based representation, which is a mode of general form pseudo-amino acid compositions, was widely applied in predicting multisite protein subcellular locations [15]. Let n be the total number of GO numbers that are considered in a study. Every given protein is represented as follows:

$$P_{GO} = [\triangle_1, \triangle_2, \ldots, \triangle_n]^T \tag{3.15}$$

where $\triangle_i, i = 1, 2, \ldots, n$, can be defined in different ways.

For example, in Euk-mPLoc2 [112], every protein sequence has been searched against the entire UniProtKB/SwissProt database using BLAST [113]. The sequences with similarity scoring higher than a threshold were collected to form a homology set. Let N be the total number of protein sequences in the homology set of a given protein, and $\delta(i, k)$ an indicator function as follows:

$$\delta(i,k) = \begin{cases} 1 & \text{if the } k\text{-th protein in the homology} \\ & \text{set hits the } i\text{-th GO number} \\ 0 & \text{Otherwise} \end{cases}$$

$\triangle_i$ was defined as:

$$\triangle_i = \delta(i,1) \vee \delta(i,2) \vee \ldots \vee \delta(i,N) \tag{3.16}$$

where $\vee$ is the disjunction operator in logical algebra. P_{GO} is a binary vector with the definition in equation 3.16, of which the i-th dimension indicates whether a hit can be found against the i-th GO number for any of the protein in the homology set. However, the importance of i-th GO number cannot be represented using only a binary form. Therefore, in iLoc-Animal [114] and iLoc-Euk [115], an improved $\triangle_i$ was proposed as follows:

$$\triangle_i = \frac{1}{N} \sum_{k=1}^{N} \delta(i,k) \tag{3.17}$$

This definition incorporates the enrichment information of a certain GO number in the homology set of a given protein, which can be used as the importance of GO number.

3.6.5.8 Sequential Evolution Information

Biology is a natural science with historic dimension. This is very interesting that a very limited number of ancestral species are responsible for biological species developed so far. The fact is also applicable for the protein sequences [115, 116]. However,

the evolution of protein sequences involves changes of single residues, insertions and deletions of several residues, gene doubling, and gene fusion [115, 116]. In spite of having long times for the above changes take place, the same biological function of proteins and their presence in a same subcellular location are still common but many similarities between initial and resultant corresponding proteins sequences got eliminated.

In this context, to incorporate the sequential evolution information, let us use the information of the PSSM (Position-Specific Scoring Matrix) [117], as described below:

1. Given a query protein sequence P as formulated by

$$P = R_1 R_2 R_3 R_4 R_5 R_6 \ldots\ldots R_L \tag{3.18}$$

According to [117], the sequential evolution information of protein P can be expressed by a $20 \times L$ matrix as given by

$$\begin{bmatrix} \acute{E}_{1\to 1} & \acute{E}_{2\to 1} & \ldots & \acute{E}_{L\to 1} \\ \acute{E}_{1\to 2} & \acute{E}_{2\to 2} & \ldots & \acute{E}_{L\to 2} \\ \vdots & \vdots & \ldots & \vdots \\ \acute{E}_{1\to 20} & \acute{E}_{2\to 20} & \ldots & \acute{E}_{L\to 20} \end{bmatrix} \tag{3.19}$$

where L is the length of P (counted in the total number of its constituent amino acids as shown in Eq. 3.18), $\acute{E}_{i\to j}$ represents the score of the amino acid residue in the i-th position of the protein sequence being changed to amino acid type j during the evolutionary process. Here, the numerical codes $1, 2, \ldots, 20$ are used to denote the 20 native amino acid types according to the alphabetical order of their single character codes. The $20 \times L$ scores in Eq. 3.19 were generated by using PSI-BLAST [117] to search the UniProtKB/Swiss-Prot database (In our work, File Name: uniprot_sprot.fasta, Download date: 14-01-2016) through three iterations with 0.001 as the E-value cutoff for multiple sequence alignment against the sequence of the protein P.

2. Use the elements in PSSM of equation 3.19 to define a new matrix M as formulated by

$$\begin{bmatrix} E_{1\to 1} & E_{2\to 1} & \ldots & E_{L\to 1} \\ E_{1\to 2} & E_{2\to 2} & \ldots & E_{L\to 2} \\ \vdots & \vdots & \ldots & \vdots \\ E_{1\to 20} & E_{2\to 20} & \ldots & E_{L\to 20} \end{bmatrix} \tag{3.20}$$

with

$$E_{i\to j} = \frac{\acute{E}_{i\to j} - \bar{\acute{E}}_j}{SD\left(\bar{\acute{E}}_j\right)} \qquad i = 1, 2, \ldots, L; \qquad j = 1, 2, \ldots, 20 \tag{3.21}$$

where

$$\bar{\acute{E}}_j = \frac{1}{L}\sum_{i=1}^{L} \acute{E}_{i\to j} \qquad j = 1, 2, \ldots, 20 \tag{3.22}$$

is the mean for $\acute{E}_{i\to j} \qquad (i = 1, 2, \ldots, L)$ and

$$SD\left(\bar{\acute{E}}_j\right) = \sqrt{\sum_{i=1}^{L}\left[\acute{E}_{i\to j} - \bar{\acute{E}}_j\right]^2 / L} \tag{3.23}$$

is the corresponding standard deviation.

3. Introduce a new matrix generated by multiplying M with its transpose matrix M^T, which contains $20 \times 20 = 400$ elements. Since MM^T is a symmetric matrix, we only need the information of its 210 elements, of which 20 are the diagonal elements and $(400 - 20)/2 = 190$ are the lower triangular elements, to formulate the protein P. Now the protein P can be formulated as

$$P_{Evo} = \left[\theta_1^E \theta_2^E \cdots \theta_u^E \cdots \theta_{210}^E\right]^T \tag{3.24}$$

where the components $\theta_u^E (u = 1, 2, \cdots, 210)$ are respectively taken from the diagonal and lower triangular elements of MM^T matrix by following a given order, say from left to right and from the 1st row to the last as illustrated by following equation

$$\begin{bmatrix} (1) & & & & \\ (2) & (3) & & & \\ (4) & (5) & (6) & & \\ \vdots & \vdots & \vdots & \ddots & \\ (191) & (192) & (193) & \ldots & (210) \end{bmatrix} \tag{3.25}$$

where the numbers in parentheses indicate the order of elements taken from the matrix MM^T.

It should be mentioned here that in this paper, the term 'PSSM' has been used to mean the name of the sequential evolution feature extraction approach.

3.7 Post-translational Modifications (PTMs)

There are two major mechanisms for expanding the coding capacity of the 6000 (yeast) to 30,000 (human) genes in eukaryotic genomes to generate diversity in the corresponding proteomes, the inventory of all proteins in a cell or organism [118]. The first route of diversification of proteins is at the transcriptional level, by mRNA splicing, including tissue-specific alternate splicing. The second route to proteome expansion is the post-translational modifications (PTMs) of proteins which are usually covalent modifications and occur after DNA has been transcribed into RNA and translated into proteins [119].

Post-translational modifications (PTMs) are widely used to modulate protein function in the cell. It increases the structural and biophysical diversity of proteins and thus enrich the information stored in the genomes. The accomplishment of Human Genomic Project is one of the greatest science and technology achievements in the twen-tieth century [120]. Upon close inspection of the first complete draft of the human genome it is surprisingly found out that only about 30000 to 50000 genes have been found [121], which is only about three or five times that in the eelworm or the drosophila. It is far from enough to regulate such a complex life process only depending on such a small number of genes. Therefore protein post-translational modification process is extremely important. It makes the protein obtain more complicated structures, perfect functions, more accurate regulations and more specific operations. Human complexity is not simply a result of the direct protein products of genes. It is the protein post-translational modification that allows one gene to correspond not only to one protein, which therefore grants, to some extent, our human life diversity.

There are various types of PTMs that are incorporated by the cell. To achieve the required effect, a protein may undergo a single PTM or several PTMs that may engage in cross-talk. In many cases, a single position on the protein can be altered by different modifications so that switching between several effects (or functions) can be regulated by the identity of the PTM at that position. PTMs are often classified according to the mechanisms involved: the addition of functional groups (e.g., phosphorylation and glycosylation); attachment of other polypeptides (e.g., ubiquitination and SUMOylation); changing of the chemical nature of amino acids (e.g., acetylation, deamidation and oxidation); and cleavage of the backbone by proteolysis. PTMs can also be categorized according to the conformational preference of the modification sites; namely, if the PTM occurs on a structured or disordered region. PTMs at structured domains are crucial, for example, for modifying enzymatic activities or stabilizing protein structure. PTMs at disordered regions are advantageous because of the high-exposure of these sites, which enables them to be accessed easily by the modifying enzymes through high-specificity and low-affinity interactions that also often involve a disorder-to-order transition.

Some Common Post-translational modifications (PTMs) are listed below:

Table 3.4: 10 most common experimentally found modifications.

Frequency	**Modification**
8383	Phosphorylation
6751	Acetylation
5526	N-linked glycosylation
2844	Amidation
1619	Hydroxylation
1523	Methylation
1133	O-linked glycosylation
878	Ubiquitylation
826	Pyrrolidone Carboxylic Acid
504	Sulfation

- Ubiquitylation
- Acetylation
- Acylation
- Glycosylation
- Hydroxylation
- Succinylation
- Phosphorylation
- Sulfation
- Methylation
- Lipodation
- Carbonylation

3.7.1 Statistics of PTMs of Proteins

In 2011, statistics of each post-translational modification experimentally and putatively detected have been compiled using proteome-wide information from the Swiss-Prot database [122]. The 10 most common experimentally found modifications has been shown in Table 3.4.

3.7.2 Laboratory Experimental Approaches for Determining PTMs in Proteins

Various types of laboratory experimental approaches have been developed to detect various types of PTMs sites [106, 123]. For example, liquid chromatography has been used to analyze carbonylated protein, optical detection methodologies, phosphor-specific antibody , dual-mode field-effect devices and nanoplasmonic sensors, and radioisotope labelling have been used to detect phosphorylated sites, affinity-tagged Ub, Ub antibodies and Ub-binding have been used to Ubiquitinated sites and finally high-throughput mass-spectrometry (MS) technique for almost all types of PTMs sites detection.

However, all the purely experimental techniques to determine PTM substrates are cost inefficient, time consuming, laborious, or sometimes it require the involvement of chemical reagents [106, 55]. For example, when mass spectrometry is used, it requires expertise and large investments [55].

3.7.3 Necessity of Computation Tools for PTMs Prediction

As protein post-translational modification is not directly decided by gene, studies on protein post-translational modification are of great significance for future researches of proteomics. Moreover, various types of major human diseases including Alzheimer's disease, diabetes, Parkinson's disease, chronic renal failure, chronic lung disease, sepsis are associated with protein carbolnylation [48]. Understanding what influences the post-translational modification will help to uncover cellular process and function of protein network in molecular level and finally help to design and develop drug more precisely[120].

However, the purely experimental techniques, define in section 3.7.2 to determine the exact modified sites of proteins is expensive as well as time-consuming, especially for large-scale datasets. In this context, it is highly demanded to use computational approaches to identify the PTM sites effectively and accurately [106].

3.7.4 Methods to Predict Post-translational Modification and Dataset

Various types of computational methods have developed to predict various types of post-translational modification. Some method have developed to predict whether a protein can be able to be modified by any PTM and some other methods have developed to predict PTM sites of a known modified proteins. Random forest and ensemble of random forest methods are very popular in the prediction of various types of PTM and PTM sites (succinylation, carbonylation, phophorylation, etc.) prediction [49, 51, 104, 105, 50, 55] . Another important used method in the prediction of PTM sites is SVM [124, 106, 107, 125, 37, 126]

Generally, in the field of PTM site prediction, researchers collect protein sequences containg PTM site from different sources such as UniProtKB/Swiss-Prot, CPLM databases, multiple published articles, etc. Most of the researcher made benchmark dataset

Table 3.5: Dataset name and their location of source

Dataset Name	Location
Carbonylation Sites	http://www.jci-bioinfo.cn/iCar-PseCp
Dephosphorylation Sites	http://genomics.fzu.edu.cn/dephossite/
Lysine Malonylation Sites	http://app.aporc.org/Mal-Lys/
S-sulfenylation Sites	http://app.aporc.org/iSulf-Cys/
Succinylation Sites	http://www.jci-bioinfo.cn/iSuc-PseOpt
Phophorylation sites	http://www.jci-bioinfo.cn/iPhos-PseEn
Phophorylation Proteins	http://www.jci-bioinfo.cn/Multi-iPPseEvo

set by collecting protein sequences from the web site at http://www.uniprot.org/ by giving various constrains such as i) experimental assertion for evidence, ii)consider only human protein sequences, and iii) use keywords of 'acetyllysine', 'crotonyllysine', 'methyllysine' , 'succinyllysine' or other name of PTM site in the advance search option.

Some of the benchmark dataset with their source of location and the name of the reference paper are listed in Table 3.5.

3.7.5 Feature Extraction Technique of PTMs Site Prediction

In the theme of using machine learning methods to predict post-translational modification sites (PTMs), the protein sequences are fragmented by considering a window size with keeping the modified residue in the central position. According to Chou's scheme, a potential PTM site containing peptide sample can be generally expressed by

$$P_{\xi}(\odot) = R_{-\xi}R_{-(\xi-1)}R_{-(\xi-2)}\cdots R_{-2}R_{-1} \odot R_{+1}R_{+2}\cdots R_{+(\xi-1)}R_{+\xi} \tag{3.26}$$

where the symbol $\odot$ denotes the single amino acid modified residue (such as K, S, T, Y, etc.) the subscript ξ is an integer, $R_{-\xi}$ represents the ξ-th upstream amino acid residue from the center, the $R_{+\xi}$ represents the ξ-th downstream amino acid residue, and so forth.

However, as most existing machine learning algorithm can handle only vector but not sequence sample, one of the critical problem in PTMs prediction is how to extract vector from peptide sample with keeping considerable sequence characteristics. The appropriate features of protein samples plays very important roles for the prediction

of PTM site in proteins. Some of the well known feature extraction technique in the field of PTM site prediction are listed below [106, 105]:

- Binary encoding
- CKSAAP encoding scheme
- Position-specific amino acid propensity (PSAAP)
- AAIndex property
- Amino Acid Occurrence Frequency (AAOF)
- K Nearest Neighbor Score (KNNS) Encoding based on attribute grouping
- Position Weight Amino Acid Composition (PWAAC)
- Sequence Coupling Model

To understand clearly of some experiments of this thesis, one of the feature extraction techniques has been discussed in the following section:

3.7.5.1 Sequence Coupling Model

The $(2\xi + 1)$-tuple peptide sample (defined in equation 3.26) $P_\xi(\odot)$ can be further classified into the following two categories:

$$P_\xi(\odot) \in \begin{cases} P_\xi^+(\odot) & \text{if its center is a modified site} \\ P_\xi^-(\odot) & \text{otherwise} \end{cases} \tag{3.27}$$

where P_ξ^+ denotes a true PTM segment with modified residue (such as K, S, T, Y, etc.) at its center, P_ξ^- denotes a corresponding false modified segment, and the symbol $\in$ means 'a member of' in the set theory.

Based on the general PseAAC's concept, the peptide sequence of equation 3.26 can be formulated as

$$P_\xi(\odot) = P_\xi^+(\odot) - P_\xi^-(\odot) \tag{3.28}$$

where

$$P_{\xi}^{+}(\odot) = \begin{bmatrix} p_{-\xi}^{+}\left(R_{-\xi}|R_{-(\xi-1)}\right) \\ p_{-(\xi-1)}^{+}\left(R_{-(\xi-1)}|R_{-(\xi-2)}\right) \\ \vdots \\ p_{-2}^{+}\left(R_{-2}|R_{-1}\right) \\ p_{-1}^{+}\left(R_{-1}\right) \\ p_{+1}^{+}\left(R_{+1}\right) \\ p_{+2}^{+}\left(R_{+2}|R_{+1}\right) \\ \vdots \\ p_{+(\xi-1)}^{+}\left(R_{+(\xi-1)}|R_{+(\xi-2)}\right) \\ p_{+\xi}^{+}\left(R_{+\xi}|R_{+(\xi-1)}\right) \end{bmatrix} \tag{3.29}$$

and

$$P_{\xi}^{-}(\odot) = \begin{bmatrix} p_{-\xi}^{-}\left(R_{-\xi}|R_{-(\xi-1)}\right) \\ p_{-(\xi-1)}^{-}\left(R_{-(\xi-1)}|R_{-(\xi-2)}\right) \\ \vdots \\ p_{-2}^{-}\left(R_{-2}|R_{-1}\right) \\ p_{-1}^{-}\left(R_{-1}\right) \\ p_{+1}^{-}\left(R_{+1}\right) \\ p_{+2}^{-}\left(R_{+2}|R_{+1}\right) \\ \vdots \\ p_{+(\xi-1)}^{-}\left(R_{+(\xi-1)}|R_{+(\xi-2)}\right) \\ p_{+\xi}^{-}\left(R_{+\xi}|R_{+(\xi-1)}\right) \end{bmatrix} \tag{3.30}$$

In Eq. 3.29 $p_{-\xi}^{-}\left(R_{-\xi}|R_{-(\xi-1)}\right)$ is the conditional probability of amino acid $R_{-\xi}$ occurring at the left 1^{st} position (see Eq. 3.26) given that its closest right neighbor is $R_{-(\xi-1)}$, $p_{-(\xi-1)}^{-}\left(R_{-(\xi-1)}|R_{-(\xi-2)}\right)$ is the conditional probability of amino acid $R_{-(\xi-1)}$ occurring at the left 2^{nd} position given that its closest right neighbor is $R_{-(\xi-2)}$, and so forth. Note that in Eq. 3.29, only $p_{-1}^{+}\left(R_{-1}\right)$ and $p_{+1}^{+}\left(R_{+1}\right)$ are of non-conditional probability since the right neighbor of R_{-1} and the left neighbor of R_{+1} are always $\odot$ (modified residue). All these probability values can be easily derived from the positive benchmark dataset. Likewise,the components in Eq. 3.30 are the same as those in Eq. 3.29 except for that they are derived from the negative benchmark dataset.

Chapter 4

Implementation of Experiments, Results and Discussion

4.1 Introduction

Multiple kernel learning uses a predefined set of kernels and learns an optimal linear or non-linear combination of kernels as part of the algorithm. Reasons to use multiple kernel learning include a) the ability to select the appropriate kernel and its parameters (both are considered as the choice of kernel problem) from a larger set of kernels, and b) combining data from different sources that have different notions of similarity. Considering these two uses of MKL, we have applied this technique in two well known bioinformatics problem to improve their performance than existing systems. One is protein subcellular localization prediction and another one is post-translation modifications (PTMs) prediction.

In the case of protein subcellular localization prediction, we have used MKL based SVM in the following two situations in order to improve the performance of our system than other existing systems.

1. When we will use single source of information, in that case, choice of kernel problem will be solved by MKL. Herein, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) will considered as the search space of the choice of kernel problem.

2. MKL has been used to fuse different source of information.

In case of PTM prediction, we have tried to developed two different prediction system.

1. One is post-transtional modifications sites prediction. For a given protein, predictor will prediction in which site or amino acid will be modified. In this type of prediction, protein will be fragmented with a predefined with window size where the modified residue will take the middle position. After that, those fragment, called peptide samples, will be used to train and test the predictor.

2. Another one is that for a given protein, predictor will predict whether this protein can be modified or not. If yes, then which types of modification will occur in this protein. But, where the modification will happen will not be predicted by this system.

Like, protein subcellular localization prediction, in this two cases of PTMs prediction, various feature extraction techniques are available. So, in this type of prediction, we have also used MKL to fuse different source of information and improve the performance than other existing system.

Table 4.1: Experiment name

No.	Experiment Name
1	predMultiLoc-Gneg: Predicting Subcellular Localization of Gram-Negative Bacterial Proteins Using Feature Selection in Gene Ontology Space and Support Vector Machine with Resolving the Data Imbalanced Issue
2	predCar-Site: Carbonylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue
3	predSucc-Site: Lysine Succinylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue
4	mLysPTMpred: Multiple Lysine PTM Site Prediction Using Combination of SVM Classifier with Resolving Data Imbalanced Issue
5	Protein Subcellular Localization Prediction using Support Vector Machine with the Choice of Proper Kernel
6	Protein Subcellular Localization Prediction Using Multiple Kernel Learning Based Support Vector Machine
7	Protein Subcellular Localization Prediction Using Kernel Based Feature Fusion
8	predHumPhos: Predicting Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine
9	iMulti-HumPhos: A Multi-Label Classifier for Identifying Human Phosphorylated Proteins Using Multiple Kernel Learning Based Support Vector Machine

However, we have done nine experiments in these two area of bioinformatics, as shown in Table 4.1. In these experiments, at first we shown the power of SVM with single kernel by developing four prediction systems (Experiments 1, 2, 3, 4 of Table 4.1). In these four prediction systems, RBF kernel has been used in SVM and its parameter

sigma has been found (choice of kernel from the set of RBF kernels) by using time consuming grid search technique. In the second steps, we have done one experiment (Experiment 5 of Table 4.1) to show how proper kernel selection affects the performance of a predictor. In the third step, multiple kernel learning has been used in our one experiment (Experiment 6 of Table 4.1) to solve the choice of kernel problem. In this case, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) has been considered as the search space of the choice of kernel problem. We have also compared the runtime performance of our system with the system developed using single kernel based SVM and other existing top systems. Finally, we have used MKL in order to fused multiple data sources. We have done three experiment (Experiments 7, 8, 9 of Table 4.1). One in the field of protein subcellular localization prediction. Other two in the field of protein post translational modifications prediction. All of the experiments and their results are discussed one-by-one in the following section of this chapter.

4.2 Common Topics for Most of the Experiments

4.2.1 Imbalance Data Management

Any data set that shows an unequal distribution between its classes can be considered imbalanced data set problem. The main challenge in imbalance problem is that the small classes are often more useful, but standard classifiers tend to be weighed down by the huge classes and ignore the tiny ones. Although SVMs work effectively with balanced datasets, they provide sub-optimal models with imbalanced datasets [127, 128]. The main reason for the SVM algorithm to be sensitive to class imbalance would be that the soft margin objective function given in Eq. 2.5 assigns the same cost (i.e., C) for both positive and negative misclassifications in the penalty term [129]. In this thesis, for most of experiments, we have managed imbalance dataset issue to produce better prediction result using the following techniques.

4.2.1.1 Techniques Used in Experiments 1, 2, 3, 4, 8, 9

In the experiments 1, 2, 3, 4, 8, and 9, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem. The Different Error Costs (DEC) method is a cost-sensitive learning solution proposed in [127] to overcome this problem in SVMs. In DEC method, the SVM soft margin objective function is modified to assign two misclassification costs, such that C^+ is the misclassification cost for positive class examples, while C^- is the misclassification cost for negative class examples. In our work, the following equations give the cost for the positive and negative classes

$$C^+ = C * N/(2 * N_1), \quad C^- = C * N/(2 * N_2) \tag{4.1}$$

where N is the total number of instances, N_1 is the number of instances for positive class, and N_2 is the number of negative class.

4.2.1.2 Techniques Used in Experiment 7

In the experiment 7, we have used a hybrid model using random oversampling and Different Error Costs (DEC) to handle imbalance dataset problem. At first we have applied random oversampling and after that we have applied Different Error Costs (DEC) in our implementation. Random oversampling is simple but well-known resampling methods applied to solve the problem of class imbalance. Oversampling balances the data set by replicating the examples of minority class. In experiment 7, random oversampling has been used to replicate minority class in such a way that the ratio among majority and minority class will be almost 2:1. After doing the oversampling, still the dataset suffers from data imbalance problem. To overcome this rest imbalance situation, we have used DEC method. In DEC method, the SVM soft margin objective function is modified to assign two misclassification costs, such that C^+ is the misclassification cost for positive class examples, while C^- is the misclassification cost for negative class examples, as given in Eq. 4.1.

4.2.2 Experimental Setting

In statistical prediction, there are three commonly used methods to derive the metric values for a predictor, these are, the independent dataset test, subsampling (e.g., K-fold cross validation) test, and jackknife test [105, 49, 130] . These methods are often used for testing the accuracy of a statistical prediction algorithm. However, among those three methods, the jackknife test is deemed the most objective because it can always yield a unique result for a given benchmark data set, as reported in a comprehensive review [131]. Although the jackknife test has been increasingly and widely adopted by investigators to examine the power of various prediction methods, it takes huge computational time for a larger dataset.

In most of the experiments (Experiments 2-7, 8 of Table 4.1) of this thesis, we have used K-fold cross validation (subsampling) method to save the computational time. As the information about the exact K-way (K can be 5 or 10) splits of dataset used in previous studies is not published, therefore, in order to validate the stability and the statistical significance of our results, we have repeated the K-fold cross validation for multiple times. In order to compare our results with other system, we have also used jackknife test to derive the value of a metric in the experiments 1 and 8.

It can be mentioned here that all the trains and tests in all experiments have been conducted on a standard machine of DELL Optiplex 390 with 8 GB RAM and Core-i3 processor running at 3.30 GHz.

4.2.3 Measuring Metrics

For measuring the predictive capability and reliability of our systems or predictors we have different types of evaluation metrics depending on the nature of experiments. Although different experiments use different evaluation metrics, some experiments use common metrics. Some experiments are binary class single-label problem and some

are multiclasss multi-label problem. The evaluation metrics for multi-label problems are different than single-label problems. There are various types of multi-label evaluation metrics are available. Herein, we have also used different types of evaluation metrics in different experiments depending on the existing top systems, so that we can make comparison with that systems.

4.2.3.1 Evaluation Metrics for the Experiments 1, 7

In the experiments 1, 7, we have used the (overall) locative and actual accuracy to measure the performance of multi-label predictors. The concept of locative proteins and actual proteins has been introduced in [14, 63, 29]. The overall locative and actual accuracy are defined as follows [14]:

$$overall\ locative\ accuracy = \frac{1}{N_{loc}} \sum_{i=1}^{N_{dif}} |Y_i \cap Z_i| \tag{4.2}$$

$$overall\ actual\ accuracy = \frac{1}{N_{dif}} \sum_{i=1}^{N_{dif}} 1(Y_i \equiv Z_i) \tag{4.3}$$

where Y_i is the set of true labels of each protein, Z_i the set of predicted labels of each one, N_{loc} the number of locative proteins, N_{dif} the number of different (actual) proteins, $|-|$ the operator acting on the set to count the number of its elements, $\cap$ the intersection of sets, $1(Y_i \equiv Z_i)$ equals 1 if true labels are entirely identical to predicted labels, 0 otherwise.

According to Eq. 4.2, a locative protein can be defined as correctly predicted if any of the predicted labels matches any labels in the true label set. On the other hand, Eq. 4.3 describes that an actual protein is considered to be correctly predicted when all of the predicted labels must match in the true label set exactly [14]. Therefore, the actual accuracy is stricter than the locative accuracy [14, 63].

4.2.3.2 Evaluation Metrics for the Experiments 2, 3, 8

In the experiments 2, 3 and 8, the problems which we have solved are binary single-label problems. To evaluate for this kind of systems, a set of four metric is usually used in the literature: (i) overall accuracy or Acc, (ii) Mathew's correlation coefficient or MCC, (iii) sensitivity or Sn, and (iv) specificity or Sp [106, 132, 133, 134].

$$\begin{aligned} Sn &= \frac{TP}{(TP+FN)} \\ Sp &= \frac{TN}{(TN+FP)} \\ Acc &= \frac{(TP+TN)}{(TP+TN+FP+FN)} \\ MCC &= \frac{(TP*TN)-(FP*FN))}{\sqrt{((TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \end{aligned} \tag{4.4}$$

where TP (true positive) denotes the number of modified (carbonylated, succinylated or phophorylated) peptides correctly predicted, TN (true negative) the numbers non-modified peptides correctly predicted, FP (false positive) the non-modified incorrectly predicted as the modified peptides, and FN (false negative) the modified peptides incorrectly predicted as the non-modified peptides.

4.2.3.3 Evaluation Metrics For the Experiments 5, 6

In the experiments 5 and 6, we have developed two multiclass multi-label systems in the field of protein subcellular localization prediction. In these experiments, we have used various types of adapted measures such as accuracy and F_1 score proposed by Tsoumakas et al. [62] for evaluating multi-label classification in our evaluation.

To formally define these evaluation measures, let D be a dataset containing m proteins and $S = \{s_1, s_2, \ldots, s_q\}$ be the set of q possible subcellular components in the cell. For a given protein P, let $M^P = \{s_i | l_i^P = 1, where\ 1 \leq i \leq q\}$ be the set of locations to which protein P localizes according to the dataset, and let $\hat{M}^P = \{s_i | \hat{l}_i^P = 1, where\ 1 \leq i \leq q\}$ be the set of locations that a classifier predicts for protein P, where $\hat{l}_i^P, l_i^P \in \{1, 0\}$. It should be noted here that l_i^P or $\hat{l}_i^P$ takes the value 1 if P actually localizes s_i or predicted to s_i respectively. The multi-label accuracy and the multi-label F_1 score are computed as [17]

$$Acc = \frac{1}{|D|} \sum_{P \in D} \frac{|M^P \cap \hat{M}^P|}{|M^P \cup \hat{M}^P|}$$

and

$$F_1 = \frac{1}{|D|} \sum_{P \in D} \frac{|M^P \cap \hat{M}^P|}{|M^P| + |\hat{M}^P|}$$

respectively.

In our evaluation, we have also used adapted measures of multi-label precision and recall denoted Pre_{s_i} and Rec_{s_i} to evaluate how well our system classifies proteins as localized or not localized to each individual location s_i, and defined as follows [39]:

$$Pre_{s_i} = \frac{1}{|\{P \in D | s_i \in \hat{M}^P\}|} \sum_{P \in D | s_i \in \hat{M}^P} \frac{|M^P \cap \hat{M}^P|}{|\hat{M}^P|}$$

$$Rec_{s_i} = \frac{1}{|\{P \in D | s_i \in \hat{M}^P\}|} \sum_{P \in D | s_i \in M^P} \frac{|M^P \cap \hat{M}^P|}{|M^P|}$$

We have used here the terms Multilabel-Precision and Multilabel-Recall to refer to Pre_{s_i} and Rec_{s_i} respectively. It should be mentioned that Pre_{s_i} represents the ratio of the number of correctly predicted multiple locations to the total number of multiple locations predicted, and Rec_{s_i} represents the ratio of the number of correctly predicted multiple locations to the number of original multiple locations, for all the proteins that co-localize to location s_i [17][8]. Therefore, high values of these measures for proteins

that co-localize to the location s_i can be used to indicate that the sets of predicted locations that include location s_i are predicted correctly [17].

Standard precision and recall measures, denoted by $Pre\_Std_{s_i}$ and $Rec\_Std_{s_i}$, are used in this paper to evaluate the correctness of predictions made for each location s_i and computed as:

$$Pre\_Std_{s_i} = \frac{TP}{TP + FP}$$

$$Rec\_Std_{s_i} = \frac{TP}{TP + FN}$$

where TP (true positives) denotes the number of proteins that localize to s_i and are predicted to localize to s_i, FP (false positives) denotes the number of proteins that do not localize to s_i but are predicted to localize to s_i, and FN (false negatives) denotes the number of proteins that localize to s_i but are not predicted to localize to s_i.

Additionally, the adapted measure of F_1-label score used by Briesemeister et al. [30] for evaluating the performance of multi-location predictors will be used in our evaluation and it is defined as:

$$F_1 - label = \frac{1}{|S|} \sum_{s_i \in S} \frac{2 \times Pre_{s_i} \times Rec_{s_i}}{Pre_{s_i} + Rec_{s_i}}$$

where S is the set of all locations.

4.2.3.4 Evaluation Metrics For the Experiments 4, 9

Evaluation metrics for experiments 4 and 9 are described in their respective material and methods section.

4.3 Experiment No 1:- predMultiLoc-Gneg: Predicting Subcellular Localization of Gram-Negative Bacterial Proteins Using Feature Selection in Gene Ontology Space and Support Vector Machine with Resolving the Data Imbalanced Issue

4.3.1 Motivation and Goals

With the rapid increase of protein sequences in the post-genomic age, the need for an automated and accurate tool to predict protein subcellular localization becomes increasingly important. In this context, several types of subcellular localization prediction methods have been proposed depending on various classification methods which produce different levels of accuracy. Most of them aim to find the optimal classification scheme and less of them take simplifying the complexity of biological system into consideration. There are two important issues that can take place to simplify

the complexity of prediction system before developing successful predictor: handling high dimension feature, and overcoming the challenge of large data imbalance in the training data. To the best of our knowledge, Gneg-mPLoc [135], iLoc-Gneg [29] and Gneg-ECC-mPLoc [63] are capable of predicting multi-label gram-negative bacterial proteins with highest accuracy so far. They have used GO approach to do prediction which increases the dimension of input features. However, they have not reduced the dimension of the input feature and have not solved the issue of dataset imbalance problem to develop their systems. In this context, in order to meet the current demand to produce efficient high-throughput tools, additional effort are required to further improve the prediction quality

Therefore, in this experiment, a novel computational tool termed predMultiLoc-Gneg has been developed to predict the subcellular localization of gram-negative bacterial proteins by (1) selecting relevant GO (Gene Ontology) terms in order to create a GO subspace which reduces feature dimension in contrast to the whole GO space and extracting GO-based features for a protein by considering only these relevant GO terms, (2) constructing a multi-label predictor using support vector machine (SVM) with resolving data imbalance issue. Moreover, a user-friendly web server for the predMultiLoc-Gneg has also been established at http://research.ru.ac.bd/predMultiLoc-Gneg/

4.3.2 Materials and Methods

4.3.2.1 Short Description of Dataset and Working Procedure of the Proposed System

In this work, the gram-negative bacterial benchmark dataset used in Gneg-mPLoc [135], iLoc-Gneg [29] and Gneg-ECC-mPLoc [63] is used to evaluate the prediction performance of our system. The gram-negative bacterial dataset contains 1392 different proteins, called actual proteins, which are distributed in 8 locations. Among these gram-negative proteins, 1328 belong to one subcellular location, 64 to two locations, and none to more locations. Hence, there are 1456 (1328+64 *2) locative proteins in total in this dataset. The concept of locative proteins and actual proteins has been explained in detail in literature [14, 63]. The name of these eight locations and the number of proteins in each location are shown at Table 4.2. The pairwise sequence identity among proteins in this dataset is controlled fewer than 25%. This benchmark is available at: http://www. csbio.sjtu.edu.cn/bioinf/Gneg-multi/.

The proposed predictor, called predMultiLoc-Gneg, has followed the following steps

1. GO Subspace Selection
2. Extract GO vector using GO terms (relevant GO terms) of GO subspace.
3. Finally train support vector machine with resolving data imbalance issue and produce prediction output.

Table 4.2: Distribution of protein subcellular locations in the gram-negative bacterial dataset

No.	Subcellular location	No. of proteins
1	Cell inner membrane	557
2	Cell outer membrane	124
3	Cytoplasm	410
4	Extracellular	133
5	Fimbrium	32
6	Flagellum	12
7	Nucleoid	8
8	Periplasm	180
Total number of locative proteins		1456
Total number of different proteins		1392

To provide an intuitive view, the working procedure of our system is shown in Figure 4.1.

4.3.2.2 Feature Extraction Approaches of Proteins

The appropriate features of protein samples plays very important roles for the prediction of protein subcellular localization, as a result it draws much attention of scientist that how to select the core and essential features of protein samples. Moreover, as most existing machine learning algorithm can handle only vector but not sequence sample, one of the critical problem in bioinformatics is how to extract vector from biological sequence with keeping considerable sequence characteristics [49]. In this experiment, primarily GO based feature has been used and dipeptide composition (DC) feature has been used as backup where GO based feature is not available. Dipeptide composition (DC) feature extraction approach has been discussed in section 3.6.5.2. In this study, we have extracted GO based feature a little bit different ways than that defined in section 3.6.5.7, which is discussed in below:

Gene Ontology

Gene Ontology (GO) is a set of standardized vocabularies that annotate the function of genes and gene products across different species [14, 136]. In the GO database,

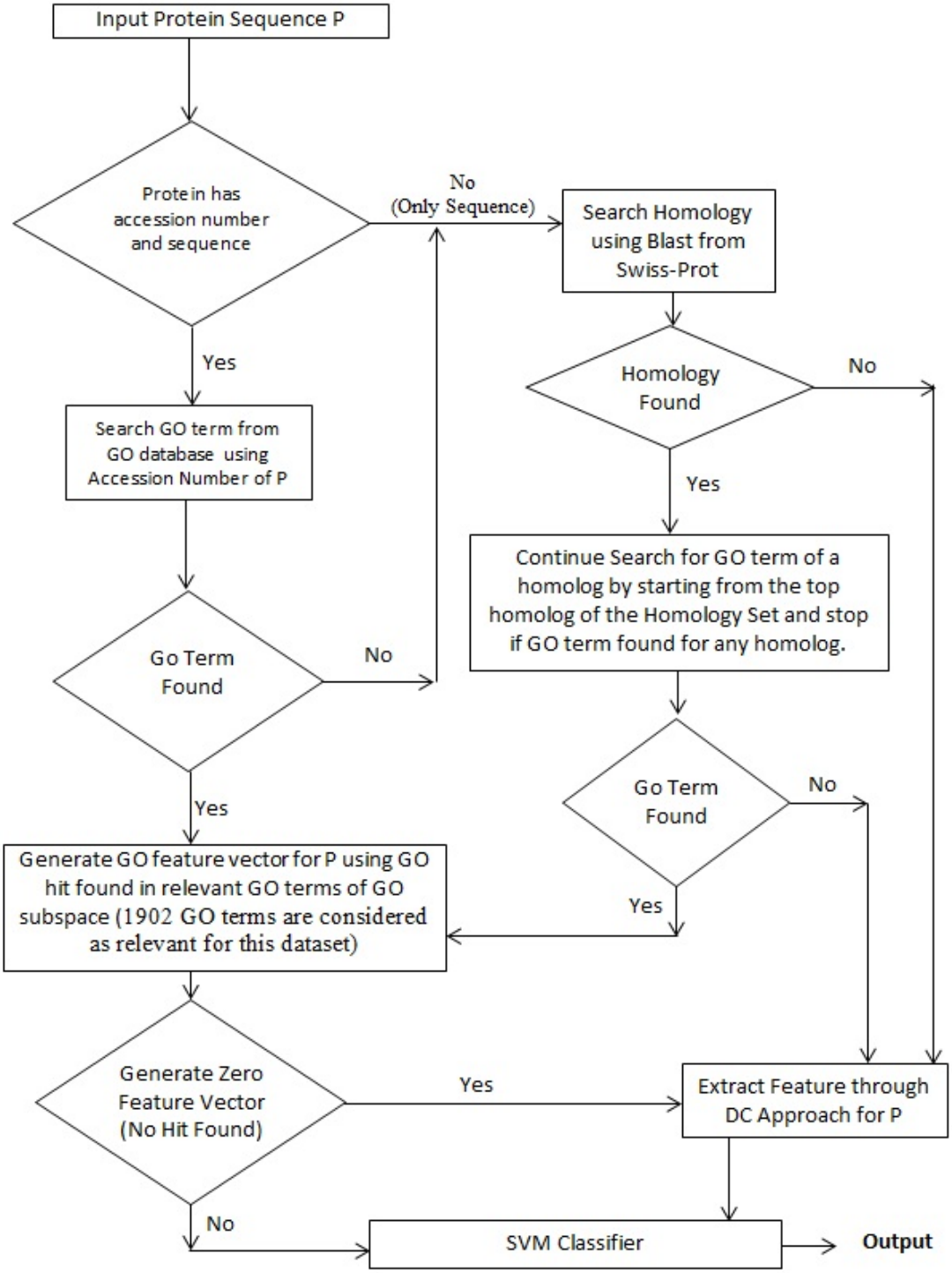

Figure 4.1: A flowchart to show the prediction process of predMultiLoc-Gneg

the annotations of gene products are organized in three related ontologies: cellular components, biological processes, and molecular functions. For having continuous effort of the GO consortium annotation, the Gene Ontology (GO) annotation database has become a large and comprehensive resource for proteomics research [136] such as predicting the enzymatic attribute of proteins, predicting the transcription factor DNA binding preference. As a result, Gene Ontology could be used to improve the predictive performance of protein subcellular localization [33]. In this paper, the GO based feature extraction process has been divided into two parts :- (I) GO subspace selection and (II) GO terms extraction and GO-vector construction.

(I) GO Subspace Selection

To facilitate the sophisticated machine learning approach for the multi-label problem, GO subspace selection is adopted from the work of Wan et. al.[14]. Unlike some prediction methods [135, 29, 63] which use all of the GO terms in the GO annotation database to form the GO-vector space, predMultiLoc-Gneg selects a GO subspace by finding a set of distinct relevant GO terms. With the rapid growth of the GO database, the number of GO terms is also increasing. As of January 2016, the number of GO terms is 27003, which means that without feature selection, the GO vectors will have dimension 27003. This imposes computational burden on the classifier, especially when leave-one-out cross validation is used for evaluation [14]. There is no doubt that many of the GO terms in the full space are redundant, irrelevant or even detrimental to prediction performance [14]. In this work, we have selected 1902 GO terms that appeared for the 1392 gram-negative bacterial proteins in the GO database to form a GO subspace. By selecting a set of distinct GO terms to form a GO subspace, predMultiLoc-Gneg has reduced the irrelevant information and at the same time increased the speed of the prediction system without compromising the performance. It should be mentioned that these selected 1902 GO terms will be called as relevant GO term.

(II) GO Terms Extraction and GO-Vector Construction

The predMultiLoc-Gneg predictor supports two types of input format of protein. One format considers both the accession numbers and amino acid (AA) sequences as input and should follow the FASTA format. The other one consider only amino acid (AA) sequences as input. When we know both accession numbers and amino acid (AA) sequences of query protein then we will use the following steps to find GO terms and construct GO-vector.

1. Search the accession number against the GO database (release of GO database on 20-01-2016) at http://www.ebi.ac.uk/GOA/ to find the corresponding GO term. If we get GO term for that protein using the accession number then go to step 4. However, many proteins have not been annotated in the GO database due to recently addition to Swiss-Prot. As a result, if we use the accession numbers of these proteins to search against the GO database, the corresponding GO term

will be empty. This suggests that we should use the ACs of their homologs as the searching keys. In order to get ACs of a protein's homologs go to step 2.

2. Use BLAST [117] to search the homologous proteins of the query protein P from the Swiss-Prot database (File Name: uniprot_sprot.fasta, Download Date: 14-01-2016), with the expect value $E \leq 0.001$ for the BLAST parameter. Those proteins which have $\geq 60\%$ pairwise sequence identity with the query protein P are collected into a set, S_P^H, called the "homology set" of P. All the elements in S_P^H can be deemed as the "representative proteins" of P. Because they are retrieved from the Swiss-Prot database, these representative proteins must have their own accession numbers.

3. Use the accession number of top homolog of S_P^H to find GO term against the GO database and go to step 4 if GO term has been found. However, we observed that for some novel proteins, even the top homologs do not have any GO terms annotated to them. To overcome this limitation, the following procedure has been adopted. For the proteins whose top homologs do not have any GO terms in the GO database, we have used the second-top homolog to find the GO terms, similarly, for the proteins whose top 2^{nd} homologs do not have any GO terms, the third-top homolog has been used, and so on until got an accession number from S_P^H that have GO terms. If GO terms of an accession number has been found in the search procedure then stop the search and go to step 4. It should be noted that if we do not get any GO term for any homolog of S_P^H then our system will use backup feature to predict the subcellular location of the query protein.

4. Suppose K distinct GO terms have been found in step 1 or step 3. Then the query protein P has been expressed by the following formulation

$$P_{GO} = [\theta_1^G \theta_2^G \cdots \theta_u^G \cdots \theta_{1902}^G]^T \tag{4.5}$$

where T is the transposing operator, and

$$\theta_i^G = \begin{cases} 1 & \text{if a hit is found the i-th GO term} \\ 0 & \text{otherwise} \end{cases}$$

In order to use the benefit of GO approach the accession number of a query protein is indispensable as an input. But, there are many proteins, such as synthetic and hypothetical proteins, or newly-discovered sequences without being deposited into databanks yet, which do not have accession numbers [135, 112]. In that case, we have followed the above mentioned steps 2 to 4 to extract GO terms and construct GO vector.

Note that the GO formulation of Eq. 4.5 may become a naught or meaningless vector under any one of the following situations: (1) the query protein itself and its homolog proteins do not have GO information, (2) the query protein does not have

significant homology to any protein in the Swiss-Prot database, i.e., $S_P^H = \emptyset$ meaning the homology set is an empty one, (3) there is no common GO term between found GO terms in the above mentioned steps 2 and 3 and the relevant GO terms defined in GO subspace selection. Therefore, it is necessary to consider the dipeptide composition (DC) based feature representation for those proteins which fail to be meaningfully expressed the GO feature representation.

4.3.2.3 Support Vector Machine and its Kernel

The multiclass multi-label SVM modeling algorithm has been used in this study. It is explained in section 2.2.1.3. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.3.2.4 Imbalance Data Management

In this experiment, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem for this kind of prediction that is discussed in 4.2.1.1.

4.3.2.5 Experimental Setting

In this paper, in order to save the computational time, two stage approaches has been used to find the best model for our predictor. In the first stage, we have used 5 complete runs of the 5-fold cross validation and each time we have used grid-search technique to select the values of parameters considering the highest actual accuracy. In this way, we have got 5 sets of parameters (such as values of C and sigma) for SVM. In the second stage, we have used jackknife test and searched again using grid search approach to select the values of parameter. But, this time the search space will be limited to the 5 sets of values of the parameter which has been found in the first stage.

4.3.2.6 Measuring Metrics

In this work we have used the (overall) locative and absolute accuracy to measure the performance of multi-label predictors. The formulation of locative proteins and actual proteins has been define in section 4.2.3.1.

4.3.3 Results and Discussion

4.3.3.1 Model Selection for SVM

The selection of an appropriate kernel and its parameters for a certain classification problem influence the performance of the SVM. The literature survey have showed that most of the researchers applied radial basis function (RBF) kernel to build SVM based subcellular localization prediction [14] and have found the value of its parameter by using different techniques such as trial and error, heuristics or grid search procedure. In our experiment, we have also used RBF kernel and at the same time, grid-search

Table 4.3: Selected C and σ of 5 complete runs of the 5-fold cross-validation for RBF kernel based SVM.

No. of Completes Run	C	σ
1^{st}	2^8	2^7
2^{nd}	2^6	2^6
3^{rd}	2^8	2^7
4^{th}	2^8	2^7
5^{th}	2^7	2^7

technique has been used to find the best model for SVM. Grid search method selects the best solution by evaluating several combinations of possible values. To find the parameter value C (penalty term for soft margin) and sigma, we have considered the values from 2^{-8} to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^+ and C^- defined in equation 4.1

According to the experimental setting defined in section 4.3.2.5, in the first stage, we have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter (value of C and sigma) of the classifier depending on the value of actual accuracy using grid search technique. In this way, five sets of C and sigma have been selected for SVM classifier as shown in Table 4.3. However, in the second stage, we have performed jackknife test and searched the best parameter for C and sigma among the selected five sets of C and sigma. Finally, we have found the best model with the value of $C = 2^8$ and $\sigma = 2^7$ for our predictor through jackknife test.

It can be mentioned here that we have used Matlab 2014b version to implement our system where the svmtrain function of Matlab by default uses DEC with the same cost defined in Eq. 4.1 to handle imbalance situation.

4.3.3.2 Prediction Performance Evaluation

In this section, we have compared our proposed predMultiLoc-Gneg predictor with three state-of-the-art gram negative bacterial multi-label predictors, i.e., Gneg-mPLoc [135], iLoc-Gneg [29] and Gneg-ECC-mPLoc [63] predictors. In this article, we have used the jackknife test to evaluate the prediction performance of our proposed predMultiLoc-Gneg predictors. Note that if a query protein itself and its homologous proteins do not have any GO term from the GO database or produce zero GO-vector if GO terms found, i.e. no GO hit found in GO subspace (explained in section 4.3.2.2), dipeptide composition feature based backup method is used for predicting subcellular localization of

Table 4.4: Performance comparison of predMultiLoc-Gneg with the state-of-the-art predictors on the gram-negative bacterial benchmark dataset by the jackknife test

Subcellular location	Success rate by jackknife test			
	predMultiLoc-Gneg	Gneg-ECC-mPLoc	Gneg-mPLoc	iLoc-Gneg
Cell inner membrane	98.38%	95.5%	94.3%	96.8%
Cell outer membrane	95.16%	94.4%	84.7%	83.1%
Cytoplasm	96.59%	92.2%	87.1%	89.5%
Extracellular	96.99%	93.2%	59.4%	86.5%
Fimbrium	100%	93.8%	87.5%	93.8%
Flagellum	100%	100%	0.0%	100%
Nucleoid	87.50%	87.5%	0.0%	50%
Periplasm	96.11%	94.4%	85.6%	89.4%
Overall locative accuracy	97.18%	94.1%	85.7%	91.4%
Overall actual accuracy	92.96%	92.4%	_	89.9%

protein. In the gram-negative bacterial benchmark dataset, we have not got any situation defined above. But, in order to predict future data, we have added DC feature based backup classifier using SVM in our prediction system.

Table 4.4 shows the comparison results of our proposed predMultiLoc-Gneg predictor against three state-of-the-art multi-label predictors for gram-negative bacterial benchmark dataset. As can be seen from Table 4.4, considering the overall locative accuracy, predMultiLoc-Gneg performs better than Gneg-mPLoc, iLoc-Gneg and Gneg-ECC-mPLoc. Specifically, the overall locative accuracy achieved by predMultiLoc-Gneg is 97.18%, which is more than 11% higher than that achieved by Gneg-mPLoc, 5% higher than that achieved by iLoc-Gneg and 3% higher than that achieved of Gneg-ECC-mPLoc . On the other, the overall actual accuracy of predMultiLoc-Gneg is 92.96%, which is little bit higher than Gneg-ECC-mPLoc and 3% higher than iLoc-Gneg.

As for the individual locative accuracies for the gram-negative bacterial dataset, predMultiLoc-Gneg achieves higher locative accuracies than Gneg-ECC-mPLoc and iLoc-Gneg for all of the 8 locations, except for the 'Flagellum' for which predMultiLoc-

Gneg, Gneg-ECC-mPLoc, and iLoc-Gneg achieve the similar locative accuracies. On the other hand, the locative accuracies of predMultiLoc-Gneg for all of the 8 locations are remarkably higher than Gneg-mPLoc, as shown in Table 4.4.

Compared with the existing predictors, predMultiLoc-Gneg have done better due to the following two advantages: (I) predMultiLoc-Gneg reduces the dimension of GO space by selecting a GO subspace which eliminates redundant information and then improves the prediction performance of multi-label proteins [14], (II) it minimizes the effect of imbalance training dataset which can generate better prediction results [49].

4.3.3.3 Protocol Guide

To attract more users especially for the experimental scientists and enhance the value of practical application, a user-friendly web-server for predMultiLoc-Gneg has been established at http://research.ru.ac.bd/predMultiLoc-Gneg/. A brief step-by-step guide on how to use the web server is given below

I Open the web server at http://research.ru.ac.bd/ predMultiLoc-Gneg / and you will find the home page of the predictor on your display as shown in Fig. 4.2. You will have to either type or copy and paste the query protein sequence into the input text box at the center of Fig. 4.2. The predMultiLoc-Gneg predictor supports two types of input protein format. One format has both the accession numbers and amino acid (AA) sequences as input and should follow the FASTA format. The other one consider only amino acid (AA) sequences as input. The example of these input format of protein sequences are available by clicking at example button located right above the input text box.

II In order to get the predicted result, click on the Submit button. For example, if you use any of the query protein sequence given under Example button as input, it will take 30s or more from the time of your submission to get desired output of each sequence.

III In order to get batch prediction, you will have to enter desired batch input file (in FASTA format of course) via Browse button located on the lower panel, as shown in Fig. 4.2.

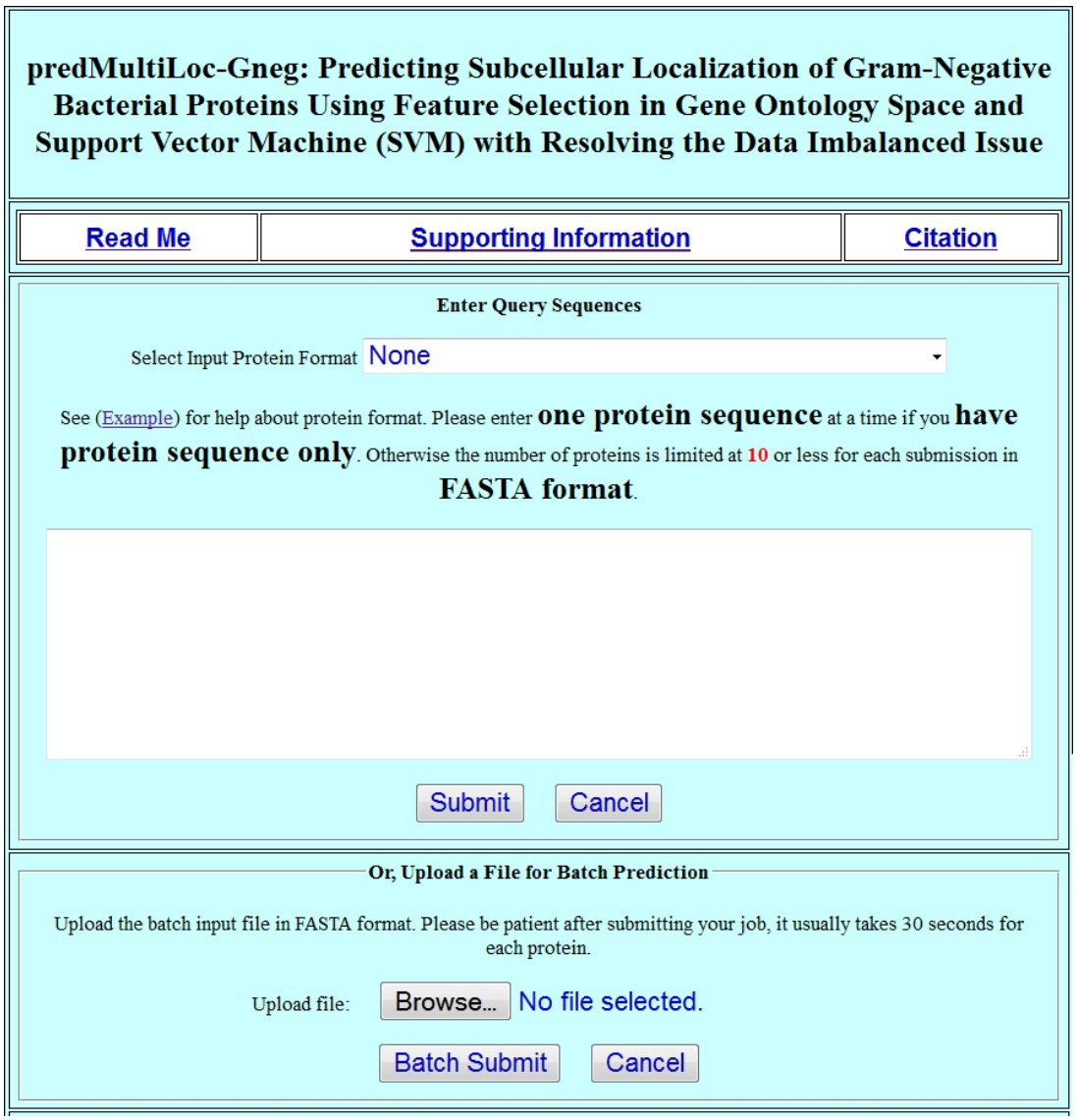

Figure 4.2: A Semi-screenshot for the home page of the webserver predMultiLoc-Gneg at http://research.ru.ac.bd/predMultiLoc-Gneg/

4.4 Experiment No 2:- predCar-Site: Carbonylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

4.4.1 Motivation and Goals

The carbonylation is found as an irreversible post-translational modification (PTM) and considered a biomarker of oxidative stress. In protein carbonylation sites, most carbonyl groups are formed from lysine (K), proline (P), arginine (R), and threonine (T) residues. The carbonylation plays major role not only in orchestrating various biological processes but also associated with some diseases such as Alzheimer's disease,

diabetes, Parkinson's disease, chronic renal failure, chronic lung disease, and sepsis. As a result, it requires an easiest way to detect carbonylation modification in proteins. However, since the experimental technologies are costly and time-consuming, so it's quite hard to detect the carbonylation modification timely at lost to face the explosive growth of protein sequences in postgenomic age. In this context, an accurate computational method for predicting carbonylation sites is an urgent issue which can be useful for drug development. Recently various types of computational classifiers [107, 106, 105] have been developed to identify carbonylation sites through different types of machine learning algorithms such as random forest, support vector machine. Moreover, in the recent works, the performance of PTMPred [107], CarSpred [106], and iCar-PseCp [105] on a large set of proteins has been studied in [105]. However, in order to meet the current demand to produce efficient high-throughput tools, additional effort are required to further improve the prediction quality [106, 105].

In this experiment, a novel computational tool termed predCar-Site has been developed to predict protein carbonylation sites by (1) incorporating the sequence-coupled information in to the general pseudo amino acid composition, (2) balancing the effect of skewed training dataset by Different Error Costs (DEC) method, and (3) constructing a predictor using support vector machine as classifier. Moreover, a user-friendly web server for the predCar-Site has also been established at http://research.ru.ac.bd/predCar-Site/

4.4.2 Materials and Methods

4.4.2.1 Short Description of Dataset

iCar-PseCp's [105] benchmark dataset set has been used in this study. iCar-PseCp's dataset was derived from the 230 carbonylated protein sequences from human [137, 138] and 20 carbonylated protein sequences from Photobacterium and Escherichia coli [138, 139].

In iCar-PseCp [105], according to Chou's scheme, a peptide sample was generally expressed by

$$P_{\xi}(\odot) = R_{-\xi}R_{-(\xi-1)}R_{-(\xi-2)}\cdots R_{-2}R_{-1} \odot R_{+1}R_{+2}\cdots R{+}(\xi-1)R_{+\xi} \tag{4.6}$$

where the symbol $\odot$ represents a sinlge amino acid code K, P, R, or T, the subscript ξ is an integer, $R_{-\xi}$ represents the ξ-th up stream amino acid residue from the center, the R_{+} represents the ξ-th downstream amino acid residue, and so forth.

The $(2\xi+1)$-tuple peptide sample $P_{\xi}(\odot)$ was further classified into the following two categories [105]

$$P_{\xi}(\odot) \in \begin{cases} P_{\xi}^{+}(\odot) & \text{if its center is a phosphorylation site} \\ P_{\xi}^{-}(\odot) & \text{otherwise} \end{cases} \tag{4.7}$$

Table 4.5: Summary of carbonylation site samples in the benchmark dataset

Attribute	Carbonylation Type and Number of Samples			
	$\odot = K$	$\odot = P$	$\odot = R$	$\odot = T$
Positive	300	126	136	121
Negative	1949	792	847	732

where $P^{+}(\odot)$ denotes a true carbonylation segment with K, P, R, or T at its center, $P^{-}(\odot)$ a false carbonylation segment with K, P, R, or T at its center, and the symbol $\in$ means "a member of" in the set theory.

The benchmark dataset $S_{\xi}(\odot)$ in iCar-PseCp's study was formulated as

$$\begin{aligned} S_{\xi}(K) &= S_{\xi}^{+}(K) \cup S_{\xi}^{-}(K), \quad when \odot = K \\ S_{\xi}(P) &= S_{\xi}^{+}(P) \cup S_{\xi}^{-}(P), \quad when \odot = P \\ S_{\xi}(R) &= S_{\xi}^{+}(R) \cup S_{\xi}^{-}(R), \quad when \odot = R \\ S_{\xi}(T) &= S_{\xi}^{+}(T) \cup S_{\xi}^{-}(T), \quad when \odot = T \end{aligned} \tag{4.8}$$

where the positive subset of $S_{\xi}^{+}(\odot)$ only contains the samples of true carbonylation segments $P_{\xi}^{+}(\odot)$, and negative subset $S_{\xi}^{-}(\odot)$ only contains the samples of false carbonylation segments $P_{\xi}^{-}(\odot)$ and the symbol $\cup$ means "union" in the set theory.

In iCar-PseCp's work, $(2\xi + 1)$-tuple peptide window was used to collect peptide segment that have K, P, R, or T at the center. It should be mentioned here that if the upstream or downstream in a protein sequence is less than ξ or greater than $L - \xi$ (L is the length of the protein sequence concerned) then the lacking amino acid has been filled with a dummy residue X in iCar-PseCp [105].

After applying some screening procedure based on some constraints on that collected peptide samples, for example, considering window size, $\geq 30\%$ pairwise sequence identity to any other peptides, iCar-PseCp finally constructed a benchmark dataset [105]. The detail procedure about the construction of iCar-PseCp's benchmark dataset is explained in [105].

Note that, depending on some preliminary test, window size was selected as 15 $(2 * \xi + 1)$ in iCar-PseCp's study, where $\xi = 7$. Thus, the benchmark dataset obtained by iCar-PseCp for $S_{\xi=7}(K)$, $S_{\xi=7}(P)$, $S_{\xi=7}(R)$, and $S_{\xi=7}(T)$ are available at online supplementary materials (http://research.ru.ac.bd/predCar-Site/) as Supporting Information S1, S2, S3, and S4, respectively. It should be mention that our published online supplementary materials are taken from iCar-PseCp's work [105]. A summary of this benchmark dataset is given in Table 4.5.

4.4.2.2 Feature Extraction

In this work, the sequence-coupling model has been adopted to extract feature from peptide segment. This feature extraction aprroach has been discussed in section 3.7.5.1.

4.4.2.3 Support Vector Machine and its Kernel

The SVM modeling algorithm has been used in this study. It is defined in section 2.2.1. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.4.2.4 Imbalance Data Management

In this experiment, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem of carbonylation sites prediction that is discussed in 4.2.1.1.

4.4.2.5 Experimental Setting

In this study, we have used K-fold cross validation (subsampling) method to save the computational time. As the information about the exact 10-way splits of dataset used in previous studies is not published [105], therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 10-fold cross validation for 5 times (i.e. 50 runs in total). It can be mentioned here that in each 10-fold cross validation the given training samples are randomly partitioned into 10 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally, we have reported the average results of all metrics in this study.

4.4.2.6 Measuring Metrics

For measuring the predictive capability and reliability for this kind of classification, a set of four metrics is usually used in the literature: (i) overall accuracy or Acc, (ii) Mathew's correlation coefficient or MCC, (iii) sensitivity or Sn, and (iv) specificity or Sp [106, 132, 133, 134]. The definition of these metrics is given in section 4.2.3.2.

However, in addition to these four metrics, we have used the measure of precision since it is one of the most important measurements to evaluate the degree of credibility of a prediction system. The precision is defined as

$$precision = \frac{TP}{(TP + FP)} \tag{4.9}$$

where the meaning of TP and FP is defined in section 4.2.3.2.

At last, AUC (area under the curve) is also used to evaluate our system which will be calculated from ROC curve (receiver operating characteristic curve).

Table 4.6: Selected C and σ of 5 complete runs of the 10-fold cross-validation for RBF kernel based SVM.

No. of Completes Run	Type of Carbonylation							
	K		P		R		T	
	C	σ	C	σ	C	σ	C	σ
1^{st}	2^5	2^4	2^6	2^3	2^{-1}	2^3	2^{-2}	2^3
2^{nd}	2^6	2^4	2^6	2^3	2^2	2^3	2^1	2^3
3^{rd}	2^5	2^4	2^5	2^3	2^2	2^3	2^1	2^3
4^{th}	2^6	2^4	2^6	2^3	2^2	2^3	2^{-2}	2^3
5^{th}	2^5	2^4	2^4	2^3	2^{-2}	2^2	2^{-2}	2^3

4.4.3 Results and Discussion

4.4.3.1 Model Selection for SVM

In order to generate highly performing SVM classifiers capable of dealing with real data an efficient model selection is required. In our experiment, grid-search technique has been used to find the best model for SVM. This method selects the values of parameters considering highest performance which will be measured using a specific metric (AUC, in this case) and then time if more than one position in search space has the same performance. We have performed 5 complete runs of the 10-fold cross-validation and each time we have selected the best parameter of the classifier on basis of the value of AUC (area under the curve).

It noted here that depending on the four types of residues (K, P, R, or T) which are susceptive to carbonylation, four times model selection has been considered. If the center residue of a query peptide is $\odot = K$ then the corresponding training data must be taken from $S_{\xi=7}(K)$ if the center residue of a query peptide is $\odot = P$, then the training data must be taken from $S_{\xi=7}(P)$ and so forth.

For radial basis function (RBF) kernel based SVM, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8} to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^+ and C^- defined in equation 4.1. We have performed 5 times complete run of 10 fold cross-validations and each time we have selected the best parameter of the classifier depending on the value AUC (area under curve). The selected C and sigma of 5 complete runs of the 10-fold cross-validation on each types (dataset depending on K, P, R, or T) of training dataset is

Table 4.7: Selection of C and σ to train the system for web server

Type of PTM	C	σ
K	2^5	2^4
P	2^6	2^3
R	2^2	2^3
T	2^{-2}	2^3

shown in Table 4.6. Finally, we have averaged our results in order to ensure unbiased model selection.

It should be mentioned that we have used that value of C and sigma which appears most of the times as best model in 5 complete runs of the 10-fold cross-validation to trained the system for the web server. Considering the mentioned criteria, the selected C and sigma for each type of residue (K, R, P, or T) is given in Table 4.7.

4.4.3.2 Prediction Performance Evaluation

The values of the four metrics (cf. Eq. 4.4) obtained by the current predCar-Site predictor for K-, P-, R-, and T-type carbonylation are given in the Table 4.8. These values are the average result of 5 times complete run of 10 fold cross-validation on the benchmark dataset given in Supporting information S1, S2, S3 and S4 respectively. Moreover, standard deviations of each metrics of 5 complete runs of the 10-fold cross-validation are shown in parentheses.

The Table 4.8 also includes the corresponding rates achieved by PTMPred [107], CarSpred [106], and iCar-PseCp [105], the three existing predictors for identifying the carbonylation sites in the aforesaid benchmark dataset. It should be mentioned here that the performance of PTMPred [107], CarSpred [106] , and iCar-PseCp [105] as shown in Table 4.8 are noted from [105].

It is obvious from the Table 4.8, predCar-Site has performed remarkably better over PTMPred [107], CarSpred [106], and iCar-PseCp [105] while considering Acc, MCC, and Sn. It indicates that, the proposed new predictor has produced over all better accuracy, sensitivity, and stability. Although the achieved Sp by iCar-PseCp is higher than that by our predictor in the case of center residues K and R, the gap between its Sn and Sp is very large (54% for K, 53% for R). Which implies that the results achieved by iCar-PseCp contain many false negative events [140] and hence its higher achieved Sp rate is problematic.

The area under the ROC curve is called AUC (area under the curve). The greater the AUC value is, the better the predictor will be [141, 142]. As we can see from

Table 4.8: A Comparison of the proposed predictor with the existing methods on the same 250 carbonylated proteins.

Predictor	Metrics	Type of Carbonylation			
		K	P	R	T
PTMPred	Acc (%)	88.59	82.93	86.64	88.39
CarSpred		87.22	82.93	86.22	86.61
iCar-PseCp		84.43	86.79	84.23	86.17
predCar-Site		96.95 (±0.10)	99.61 (±0.09)	99.10 (±0.26)	99.11 (±0.16)
PTMPred	MCC	0.1892	0.2573	0.1878	0.2186
CarSpred		0.2268	0.2331	0.2245	0.2040
iCar-PseCp		0.5906	0.6006	0.6076	0.6185
predCar-Site		0.8799 (±0.0034)	0.9837 (±0.0039)	0.9642 (±0.0101)	0.9646 (±0.0059)
PTMPred	Sn (%)	23.45	21.43	20.02	22.38
CarSpred		23.17	25.34	25.47	21.39
iCar-PseCp		45.18	48.20	46.67	50.68
predCar-Site		96.67 (±0.33)	99.68 (±0.43)	1	99.34 (±0.69)
PTMPred	Sp (%)	92.99	93.20	90.99	91.36
CarSpred		92.43	93.28	93.39	93.42
iCar-PseCp		99.25	98.54	99.57	98.58
predCar-Site		96.99 (±0.14)	99.60 (±0.14)	98.96 (±0.31)	99.07 (±0.22)
PTMPred	AUC	0.6858	0.6903	0.5981	0.6563
CarSpred		0.6849	0.7163	0.7158	0.7134
iCar-PseCp		0.8728	0.8484	0.8668	0.8603
predCar-Site		0.9959 (±0.00002)	0.9999 (±0.000005)	1	0.9997 (±0.000005)

Table 4.8, the value of AUC clearly indicates that the proposed predictor is better than PTMPred [107], CarSpred [106], and iCar-PseCp [105]. Therefore, it is projected that predCar-Site may become a useful and higher throughput tool in carbonylation sites predictions.

Apart from the above mentioned metrics, we have calculated precision too for our system and got the average ($\pm$ standard deviation) values of 83.19($\pm$0.62)%, 97.52($\pm$0.82)%, 93.95($\pm$1.67)%, and 94.66($\pm$1.19)% in predicting the carbonylation sites for K, P, R, and T, respectively. Since the values of precision for the other systems (PTMPred [107], CarSpred [106], and iCar-PseCp [105]) are not publicly available, as a result, we could not show those findings. The achieved values of precision of our system is very promising and encouraging. Note that precision measures how much believable the system is when it says a peptides sample is carbonylated.

Why can the proposed method enhance the prediction quality so significantly? First, the coupling effects among the amino acids around the target sites have been taken into account via the conditional probability. Second, the predictor used Different Error Costs (DEC) method to balance the effect of skewed training dataset.

4.4.3.3 Protocol Guide

To attract more users especially for the experimental scientists and enhance the value of practical application, a user-friendly web-server for predCar-Site has been established at http://research.ru.ac.bd/predCar-Site/. A step-by-step guide on how to use the web server is given below:

I Open the web server at http://research.ru.ac.bd/predCar-Site/ and you will find the home page of the predictor on your display as shown in Fig. 4.3. You will have to either type or copy and paste the query protein sequence into the input text box at the center of Fig. 4.3. The input sequence should follow the FASTA format. The example of a sequence of FASTA format is available by clicking at example button located right above the input text box.

II In order to get the predicted result, at first you have to check one of the four option (K, P, R, or T) and then click on the Submit button. For example, if you use the Sequence_K query protein sequences given under Example button as input and check on the K button, it will take 20s or more from the time of your submission to get desired output of each sequence in separate tables. All the predicted results of each lysine (K) are presented in each row of the table.

III In order to get batch prediction, you will have to enter desired batch input file (in FASTA format of course) via Browse button located on the lower panel, as shown in Fig. 4.3.

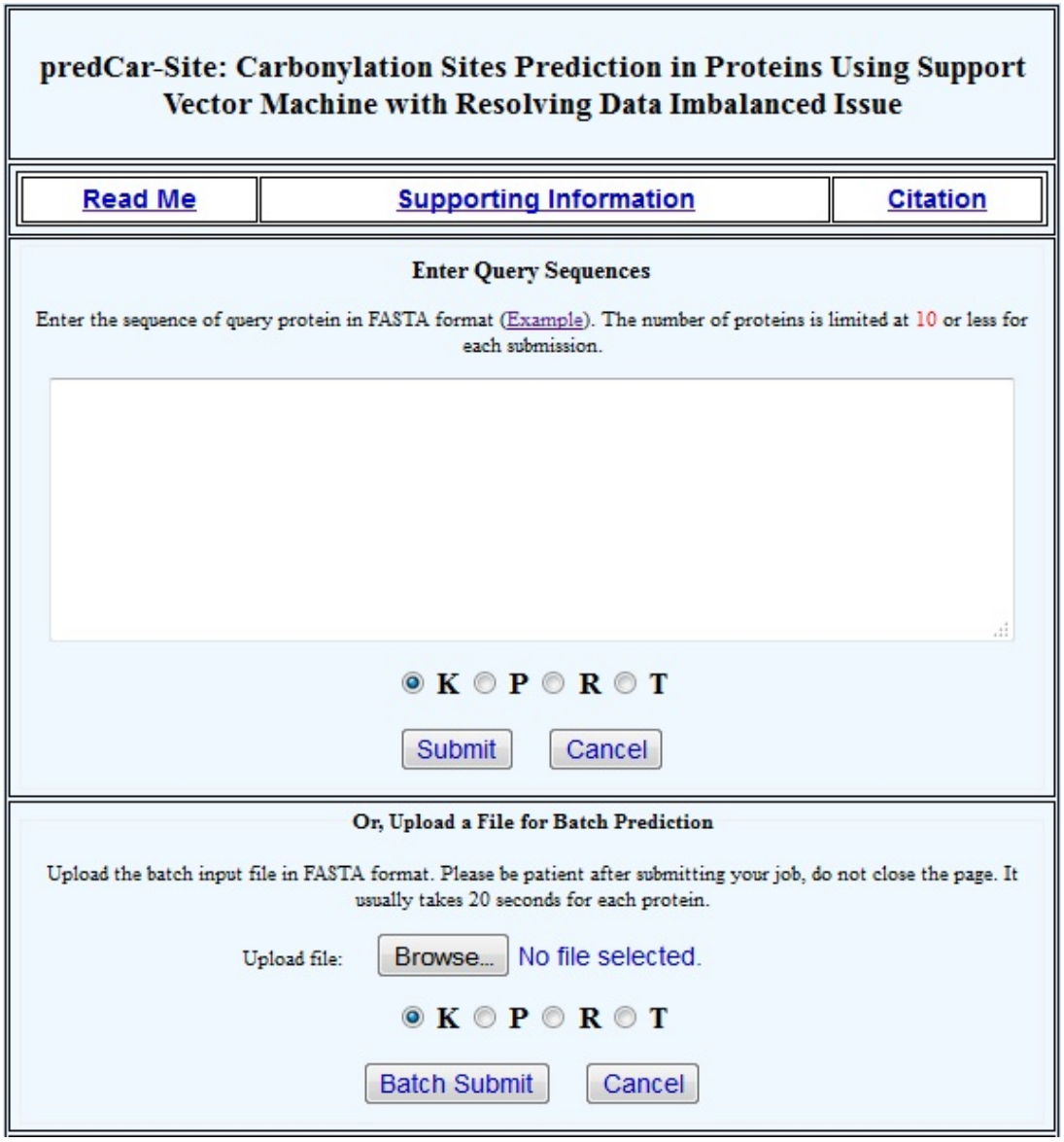

Figure 4.3: A semi-screenshot for the home page of the webserver predCar-Site at http://research.ru.ac.bd/predCar-Site/

4.5 Experiment No 3:- predSucc-Site: Lysine Succinylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

4.5.1 Motivation and Goals

The Lysine succinylation is found as an important post-translational modification where succinyle group is added to a Lys (K) residue of a protein molecule. It plays major role not only in regulating the cellular processes but also associated with some diseases. As a result, it requires an easiest way to detect succinylation modification in proteins. However, since the experimental technologies are costly and time-consuming, so it's

quite hard to detect the succinylation modification timely at low cost to face the explosive growth of protein sequences in postgenomic age. In this context, an accurate computational method for predicting succinylation sites is an urgent issue which can be useful for drug development. Recently various types of computational classifiers have been developed to identify succinylation sites through different types of machine learning algorithms [143, 51, 49]. However, in order to meet the current demand to produce efficient high-throughput tools, additional effort are required to enrich the prediction quality [51, 49].

In this study, a novel computational tool termed predSucc-Site has been developed to predict protein succinylation sites by (1) incorporating the sequence-coupled information into the general pseudo amino acid composition, (2) balancing the effect of skewed training dataset by Different Error Costs (DEC) method, and (3) constructing a predictor using support vector machine as classifier. Moreover, a user-friendly web server for the predSucc-Site has also been established at http://research.ru.ac.bd/predSucc-Site/

4.5.2 Materials and Methods

4.5.2.1 Benchmark Dataset

pSuc-Lys's benchmark dataset set has been used in this study, that dataset was derived from the CPLM, a protein lysine modification database [49, 144]. CPLM contains 2521 lysine succinylation sites and 24128 non-succinylation sites determined from 896 proteins [144]. In pSuc-Lys, all of the corresponding protein sequences were derived from the UniProt [21] database. After applying some screening procedure based on some constraints on that dataset, for example, considering window size, $<= 40\%$ pairwise sequence identity to any other peptides, pSuc-Lys finally extracted a filtered training dataset. Detail description of screening procedure is explained in [49].

The final dataset of pSuc-Lys consisted of 896 proteins with 1167 lysine succinylation sites and 3553 non-succinylation sites. It can be noted here that sliding window method was used to encode every lysine residue K of that dataset since succinylation only occurred in lysine residues K. According to [51, 49], window size has been selected as 31 $(2 * \xi + 1)$ in our study, where $\xi = 15$. The detailed sequences for the 1167 samples in the positive subset $(S^{+}_{\xi=15})$ and those for the 3553 samples in the negative subset $(S^{-}_{\xi=15})$ are available at online supplementary materials (http://research.ru.ac.bd/predSucc-Site/). Thus, the benchmark dataset set S for the current study can be formulated as

$$S_{\xi=15} = S^{+}_{\xi=15} \cup S^{-}_{\xi=15} \tag{4.10}$$

4.5.2.2 Feature Extraction

In this work, the sequence-coupling model has been adopted to extract feature from peptide segment. This feature extraction aprroach has been discussed in section 3.7.5.1.

4.5.2.3 Support Vector Machine and its Kernel

The SVM modeling algorithm has been used in this study. It is defined in section 2.2.1. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.5.2.4 Imbalance Data Management

In this paper, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem for this kind of prediction that is discussed in 4.2.1.1.

4.5.2.5 Experimental Setting

In this study, we have used K-fold cross validation (subsampling) method to save the computational time. As the information about the exact 5-way splits of dataset used in previous studies is not published [105], therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 5-fold cross-validation for 5 times (i.e. 25 runs in total). It can be mentioned here that in each 5-fold cross-validation the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally, we have reported the average results of all metrics in this study.

4.5.2.6 Measuring Metrics

For measuring the predictive capability and reliability for this kind of classification, a set of four metrics is usually used in the literature: (i) overall accuracy or Acc, (ii) Mathew's correlation coefficient or MCC, (iii) sensitivity or Sn, and (iv) specificity or Sp [106, 132, 133, 134]. The definition of these metrics are given in section 4.2.3.2. In addition, the AUC (area under the curve) is also used to evaluate the prediction system which will be calculated from ROC curve (receiver operating characteristic curve).

4.5.3 Results and Discussion

4.5.3.1 Model Selection for SVM

In order to generate highly performing SVM classifiers capable of dealing with real data an efficient model selection is required. In our experiment, grid-search technique has been used to find the best model for SVM. In our experiments, this method selects the values of parameters considering highest performance which will be measured using a specific metric (AUC, in this case) and then time if more than one position in search space has the same performance. We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier on basis of the value of AUC (area under the curve).

For radial basis function (RBF) kernel based SVM, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8}

Table 4.9: Selected C and σ of 5 complete Runs of the 5-fold cross-validation for RBF kernel based SVM

No of Complete Run	C	σ
1^{st}	2^{-2}	2^3
2^{nd}	2^{-3}	2^3
3^{rd}	2^{-3}	2^3
4^{th}	2^{-3}	2^3
5^{th}	2^{-2}	2^3

to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^+ and C^- defined in equation 4.1. We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier depending on the value of AUC (area under curve). The selected C and sigma of 5 complete runs of the 5-fold cross-validation on the benchmark data set is shown in Table 4.9. Finally, we have averaged our results in order to ensure unbiased model selection. It should be mentioned that we have used $C = 2^{-3}$ and $\sigma = 2^3$ to train the system for the web server, because most of the times, the best model is found for the value of $C = 2^{-3}$ and $\sigma = 2^3$.

4.5.3.2 Prediction Performance Evaluation

The values of the four metrics (cf. Eq. 4.4) obtained by the current predSucc-Site predictor are given in the Table 4.10. These values are the average result of 5 complete runs of the 5-fold cross-validation on the benchmark dataset given in Supporting information S1 and Supporting information S2. Moreover, standard deviations of each metrics of 5 times complete run of 5 fold cross validation are shown in parentheses.

The Table 4.10 also includes the corresponding rates achieved by iSuc-PseAAC [124], SucPred [143], pSuc-Lys [49], and iSuc-PseOpt [51], the four existing predictors for identifying the lysine succinylation sites in the aforesaid 896 proteins. It should be mentioned here that the performance of iSuc-PseAAC [124], SucPred[143], pSuc-Lys [49], iSuc-PseOpt [51] as shown in table 4.10 are noted from [51, 49].

It is obvious from the Table 4.10, predSucc-Site has performed remarkably better over iSuc-PseAAC, SucPred, pSuc-Lys and iSuc-PseOpt while considering Acc, MCC, and Sn. It indicates that, the proposed new predictor has produced over all better accuracy, sensitivity, and stability. Although the achieved Sp by SucPred, pSuc-Lys and iSuc-PseOpt is higher than that by our predictor, the gap between its Sn and Sp is very large (48% for SucPred, 19% for pSuc-Lys, 27% for iSuc-PseOpt). Which implies

that the results achieved by SucPred, pSuc-Lys and iSuc-PseOpt contain many false negative events [30] and hence its higher achieved Sp rate is problematic.

Table 4.10: A Comparison of the proposed predictor with the existing methods

Method	Acc (%)	MCC	Sn (%)	Sp (%)	AUC
iSuc-PseAAC	79.98	0.4370	50.63	89.68	0.7823
SucPred	85.32	0.5710	49.13	97.17	0.8933
pSuc-Lys	90.83	0.7695	76.79	95.97	0.9325
iSuc-PseOpt**	87.86	0.7193	69.38	96.86	0.9475
predSucc-Site	92.00 ($\pm$0.07)	0.8029 ($\pm$0.0023)	93.42 ($\pm$0.35)	91.47 ($\pm$0.04)	0.9788 ($\pm$0.0001)

** Taken the best result of iSuc-PseOpt from Table 1 of [51]

The area under the ROC curve is called AUC (area under the curve). The greater the AUC value is, the better the predictor will be [141]. As we can see from Table 4.10, the value of AUC clearly indicates that the proposed predictor is better than iSuc-PseAAC [124], SucPred [143], pSuc-Lys [49], and iSuc-PseOpt [51]. Therefore, it is projected that predSucc-Site may become a useful and higher throughput tool in succinylation sites predictions.

4.5.3.3 Protocol Guide

It has been well accepted that the availability of web-server of a prediction method will attract more users. In that context, a user-friendly web-server for predSucc-Site has been established for the convenience of experimental scientists. A brief guide on how to use the web server is given below:

I Open the web server at http://research.ru.ac.bd/predSucc-Site/ and you will find the home page of the predictor on your display as shown in Fig. 4.4. You will have to either type or copy and paste the query protein sequence into the input text box at the center of Fig. 4.4. The input sequence should follow the FASTA format. The example of a sequence of FASTA format is available by clicking at example button located right above the input text box.

II In order to get the predicted result, click on the Submit button. For example, if you use the two query protein sequences given under Example button as input, it will take 20s or more from the time of your submission to get desired output of each sequence in separate tables. All the predicted results of each lysine are presented in each row of the table.

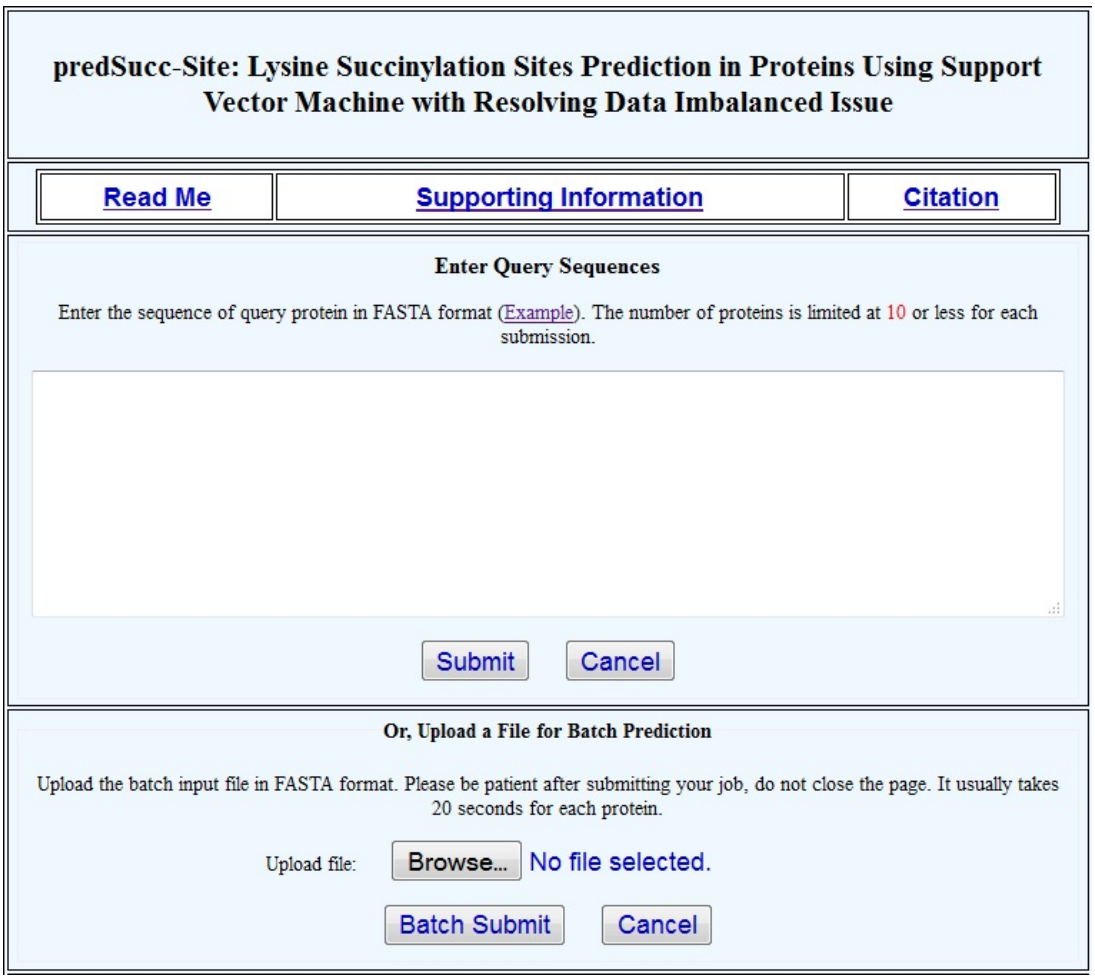

Figure 4.4: A semi-screenshot for the home page of the webserver predSucc-Site at http://research.ru.ac.bd/predSucc-Site/

4.6 Experiment No 4:- mLysPTMpred: Multiple Lysine PTM Site Prediction Using Combination of SVM Classifier with Resolving Data Imbalanced Issue

4.6.1 Motivation and Goals

Protein post-translational modification (PTM) increases the functional diversity of the proteome and plays major role not only in orchestrating various biological processes but also associated with some diseases. PTMs do modification by introducing new functional groups to the side chain of amino acid of a protein. Among all amino acid residues, the side chain of lysine can undergo many types of PTM, called K-PTM, such as 'acetylation', 'crotonylation', 'methylation' and 'succinylation' and also responsible for occurring multiple PTM in the same lysine of a protein which lead to the requirement of multi-label PTM site identification. In this context, in order to avoid the costly and time-consuming experimental technologies, an accurate computational method

for predicting multi-label PTM sites is an urgent issue which can be useful for drug development. Meanwhile, many computational methods have been developed to predict the modification sites in proteins for different K-PTM types which called single-label prediction [126, 105, 51, 130]. According to our best knowledge, so far one computational tool has been developed to predict several different K-PTM types simultaneously for multiplex lysine residues. However, in order to meet the current demand to produce efficient high-throughput tools, additional effort are required to further improve the prediction quality [104].

Therefore, in this experiment, a novel computational tool termed mLysPTMpred has been developed to predict multi-label lysine PTM sites by (1) incorporating the sequence-coupled information into the general pseudo amino acid composition, (2) balancing the effect of skewed training dataset by Different Error Costs (DEC) method, and (3) constructing a multi-label predictor using a combination of support vector machine (SVM) as classifier. Moreover, a user-friendly web server for the mLysPTMpred has also been established at http://research.ru.ac.bd/mLysPTMpred/

4.6.2 Materials and Methods

4.6.2.1 Short Description of Dataset

iPTM-mLys's [104] benchmark dataset set has been used in this study. iPTM-mLys's dataset was derived from the 1769 protein sequences from human. These 1769 protein sequences were collected from the web site at http://www.uniprot.org/ by giving various constrains such as i) experimental assertion for evidence, ii)consider only human protein sequences, and iii) use keywords of 'acetyllysine', 'crotonyllysine', 'methyllysine' or 'succinyllysine' in the advance search option.

In iPTM-mLys [104], according to Chou's scheme, a peptide sample was generally expressed by

$$P_{\xi}(K) = R_{-\xi}R_{-(\xi-1)}R_{-(\xi-2)}\cdots R_{-2}R_{-1}KR_{+1}R_{+2}\cdots R{+}(\xi-1)R_{+\xi} \tag{4.11}$$

where the subscript ξ is an integer, $R_{-\xi}$ represents the ξ-th up stream amino acid residue from the center, the $R_{+\xi}$ represents the ξ-th downstream amino acid residue, and so forth.

The $(2\xi+1)$-tuple peptide sample $P_{\xi}(K)$ was further classified into the following two categories [104]

$$P_{\xi}(K) \in \begin{cases} P_{\xi}^{+}(K) & \text{if its center is a lysine PTM site} \\ P_{\xi}^{-}(K) & \text{otherwise} \end{cases} \tag{4.12}$$

After applying some screening procedure based on some constraints on that collected peptide samples, for example, considering window size, keep only one when two or more samples share same sequence, iPTM-mLys finally constructed a benchmark dataset [104]. It should be mentioned here that iPTM-mLys constructed four

Table 4.11: Summary of the four benchmark dataset

Attribute	PTM Type and Number of Samples			
	Ace	Cro	Met	Suc
	S(1)	S(2)	S(3)	S(4)
Positive	3991	115	127	1169
Negative	2403	6279	6267	5225

Ace, acetylation; Cro, crotonylation; Met, methylation; Suc, succinylation.

bench mark dataset for 'acetylation', 'crotonylation', 'methylation' and 'succinylation', respectively using same the above mentioned procedure. The detail procedure about the construction of iPTM-mLys's benchmark dataset is explained in [104].

The four benchmark dataset $S_\xi(K)$ in iPTM-mLys's study was formulated as

$$\begin{aligned} S_\xi(acetylation) &= S_\xi^+(acetylation) \cup S_\xi^-(acetylation) \\ S_\xi(crotonylation) &= S_\xi^+(crotonylation) \cup S_\xi^-(crotonylation) \\ S_\xi(methylation) &= S_\xi^+(methylation) \cup S_\xi^-(methylation) \\ S_\xi(succinylation) &= S_\xi^+(succinylation) \cup S_\xi^-(succinylation) \end{aligned} \tag{4.13}$$

where the positive subset $S_\xi^+(acetylation)$ contains only the peptide samples with their center residues K (Eq. 4.13) confirmed by experiments being able to be of acetylation, while the negative subset $S_\xi^-(acetylation)$ only contains those samples unable to be of acetylation, and the symbol $\cup$ means union in the set theory. Likewise, the remaining three sub-equations in Eq. 4.13 have exactly the same definition but refer to 'crotonylation', 'methylation' and 'succinylation', respectively.

Using numeric values, in iPTM-mLys's study, the equation 4.13 was formulated as

$$\begin{aligned} S_\xi(1) &= S_\xi^+(1) \cup S_\xi^-(1) \\ S_\xi(2) &= S_\xi^+(2) \cup S_\xi^-(2) \\ S_\xi(3) &= S_\xi^+(3) \cup S_\xi^-(3) \\ S_\xi(4) &= S_\xi^+(4) \cup S_\xi^-(4) \end{aligned} \tag{4.14}$$

where the numerical argument 1, 2, 3 and 4 denotes 'acetylation', 'crotonylation', 'methylation' and 'succinylation', respectively.

Note that, depending on some preliminary test, window size was selected as 27 $(2 * \xi + 1)$ in iPTM-mLys's study, where $\xi = 13$. Thus, the benchmark dataset obtained by iPTM-mLys for $S_{\xi=13}(1)$, $S_{\xi=13}(2)$, $S_{\xi=13}(3)$, and $S_{\xi=13}(4)$ are available at online supplementary materials (http://research.ru.ac.bd/mLysPTMpred/) as Supporting Information. It should be mentioned that our published online supplementary materials

are taken from iPTM-mLys's work [104]. A summary of this benchmark dataset is given in Table 4.11.

4.6.2.2 Feature Extraction

In this work, the sequence-coupling model has been adopted to extract feature from peptide segment. This feature extraction approach has been discussed in section 3.7.5.1.

4.6.2.3 Support Vector Machine and its Kernel

The SVM modeling algorithm has been used in this study. It is defined in section 2.2.1. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.6.2.4 Imbalance Data Management

In this experiments, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem for this kind of prediction that is discussed in section 4.2.1.1.

4.6.2.5 Experimental Setting

In this experiments, we have used K-fold cross validation (subsampling) method to save the computational time. As the information about the exact 5-way splits of dataset used in previous studies is not published [104], therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 5-fold cross validation for 5 times (i.e. 25 runs in total). It can be mentioned here that in each 5-fold cross validation the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally, we have reported the average results of all metrics in this study.

4.6.2.6 Measuring Metrics

According to the description of iPTM-mLys dataset in [104], we have a total of 6394 samples, of which 3991 are labeled with 'acetylation' 115 with 'crotonylation', 127 with 'methylation', 1169 with 'succinylation' and 1,750 with 'non-K-PTM'. However, in the above samples, some have two or more labels. It should be noted that in this study, we have considered {acetylation, crotonylation, methylation, succinylation, $\emptyset$} as class label set for a protein. Here $\emptyset$ is used to denote non-K-PTM. Since we are dealing with a multi-label system [145], so the metrics for a multi-label system will be used in this work instead of the conventional metrics defined for single-label systems [132, 133, 134].

For measuring the predictive capability and reliability for this kind of classification, a set metrics are usually used in the literature which are define below [145]:

$$
\begin{aligned}
Aiming &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Z_k\|} \right) \\
Coverage &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Y_k\|} \right) \\
Accuracy &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Y_k \cup Z_k\|} \right) \\
Absolute - True &= 1/N \sum_{k=1}^{N} \triangle (Y_k, Z_k) \\
Absolute - False &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cup Z_k\| - \|Y_k \cap Z_k\|}{M} \right)
\end{aligned}
\tag{4.15}
$$

where N is the total number of the samples concerned, M the total number of labels in the system, $\cup$ and $\cap$ the symbols are for the 'union' and 'intersection' in the set theory, $\|-\|$ means the operator acting on the set therein to count the number of its elements, Y_k denotes the subset that contains all the labels experiment-observed for the k-th sample, Z_k represents the subset that contains all the labels predicted for the kth sample, and

$$
\triangle (Y_k, Z_k) = \begin{cases} 1 & \text{if all labels in } Z_k \text{ are identical with those in } Y_k \\ 0 & \text{otherwise} \end{cases}
$$

All of these metrics defined in this section have been successfully applied to study several multi-label systems, such as those in which a protein may stay in two or more different subcellular locations [114], or a membrane protein may have two or more different types [146], or an antimicrobial peptide may have two or more different types [147].

4.6.3 Results and Discussion

4.6.3.1 Model Selection and Working Procedure of the Proposed System

In order to generate highly performing SVM classifiers capable of dealing with real data an efficient model selection is required. Grid-search technique has been used to find the best model for SVM in this work. In our experiments, grid-search technique selects the values of parameters considering highest performance which is measured using a metric and then time if more than one position in search space has the same performance.

In this study, four SVM classifiers, one for each dataset, have been used for predicting the acetylation, crotonylation, methylation and succinylation sites. The model selection of each SVM classifiers has been done separately as binary classifier using the corresponding benchmark dataset given in Table 4.11. In this work, we have used RBF kernel for all SVM classifiers.

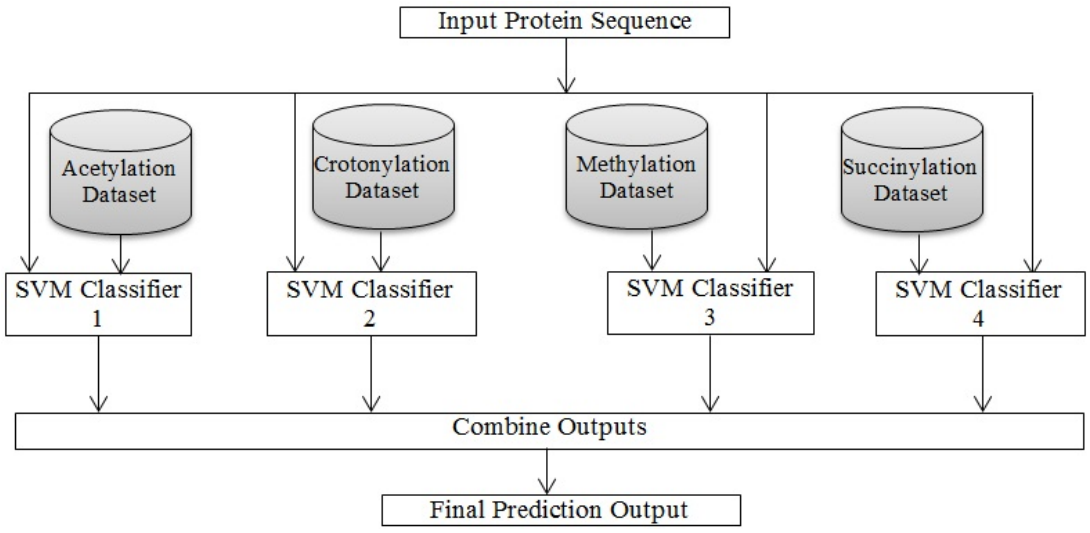

Figure 4.5: A flowchart to show how the mLysPTMpred predictor works

For radial basis function (RBF) kernel, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8} to 2^{8} for C and from 2^{-8} to 2^{8} for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^{+} and C^{-} defined in equation 4.1. Since the information about the exact 5-way splits of dataset used in previous studies is not published [104], We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier depending on the value of AUC (area under curve). It should be noted here that AUC (area under the curve) is an important metric for single-label PTM site prediction [51, 105] which will be calculated from ROC curve (receiver operating characteristic curve). Finally, five sets of C and sigma have been selected from 5 complete runs of the 5-fold cross-validation for each SVM classifier (as shown in Table 4.12) which is dedicated to a specific types of training dataset (acetylation, crotonylation, methylation, or succinylation).

After getting the four trained binary SVM classifier with appropriate values of C and sigma, a multi-label predictor, named mLysPTMpred, has been developed by combing output from these four SVM classifiers, as shown in Figure 4.5. It should be noted here that the negative results of all (four) classifiers will be treated as class label 'non-K-PTM' protein. As we have repeated the 5-fold cross validation for 5 times for our mLysPTMpred, we have got five sets of values for all metrics defined in section 4.6.2.6. Finally, we have averaged our results in order to ensure unbiased model selection.

However, in order to train the system for the web server, we have used that value of C and sigma which appears most of the times as best model in 5 complete runs of the 5-fold cross-validation in each dataset. Note that, a random selection of the value of C and sigma has also been performed from 5 set of C and sigma of each dataset where 'most of the times' criteria fail to select C and sigma. In this way, the selected C and sigma for each type of dataset is given in 4.13.

Table 4.12: Selected C and σ of 5 complete runs of the 5-fold cross-validation for RBF kernel based SVM

No. of Completes Run	Type of Carbonylation							
	acetylation		crotonylation		methylation		succinylation	
	C	σ	C	σ	C	σ	C	σ
1^{st}	2^4	2^4	2^{-1}	2^4	2^0	2^4	2^1	2^3
2^{nd}	2^4	2^4	2^0	2^6	2^4	2^4	2^1	2^3
3^{rd}	2^1	2^1	2^2	2^7	2^0	2^4	2^5	2^4
4^{th}	2^1	2^1	2^{-2}	2^5	2^4	2^4	2^1	2^3
5^{th}	2^4	2^4	2^{-2}	2^7	2^5	2^4	2^5	2^4

Table 4.13: Selection of C and σ to train the system for the web server

Type of PTM	C	σ
acetylation	2^4	2^4
crotonylation	2^{-2}	2^7
methylation	2^0	2^4
succinylation	2^1	2^3

It can be mentioned here that we have used Matlab 2014b version to implement our system where the svmtrain function of Matlab by default uses DEC with the same cost defined in Eq. 4.1 to handle imbalance situation.

4.6.3.2 Prediction Performance Evaluation

The values of the five metrics (cf. Eq. 4.15) obtained by the current mLysPTMpred predictor for multi-label lysine PTM site are given in the Table 4.14. These values are the average result of 5 complete runs of the 5-fold cross-validation on the benchmark dataset given in Supporting Information. Moreover, standard deviations of each metrics of 5 times complete run of 5-fold cross validation are shown in parentheses.

The Table 4.14 also includes the corresponding rates achieved by iPTM-mLys [104], the one existing predictors for identifying the multiple lysine PTM site in the aforesaid

Table 4.14: A Comparison of the proposed predictor with the existing methods on the same dataset

Predictor	Aiming (%)	Coverage (%)	Accuracy (%)	Absolute-True (%)	Absolute-False (%)
iPTM-mLys	69.78	74.54	68.37	60.92	13.40
mLysPTMpred	84.82 ($\pm$0.22)	86.56 ($\pm$0.21)	83.73 ($\pm$0.24)	79.73 ($\pm$0.29)	6.66 ($\pm$0.009)

benchmark dataset. It should be mentioned here that the performance of iPTM-mLys [104] as shown in Table 4.14 are noted from [104].

In Eq. 4.15, the first four metrics are completely opposite to the last one. For the former, the higher the rate is, the better the multi-label predictor's performance will be; for the latter, the lower the rate is, the better its performance will be [104]. The rate of "Absolute-False" or "Hamming- Loss" [145] for our predictor is 6.66% which is about half than iPTM-mLys. So, the average ratio of the completely wrong hits over the total prediction events for mLysPTMpred is significantly lower than iPTM-mLys.

Among the five metrics in Eq. 4.15, the most strict and harsh one is the 'Absolute-True'. According to [104], very few multilabel predictors in biology could reach over 50% for the absolute true rate. However, the absolute-true rate achieved by mLysPTMpred can reach over 80% as shown in Table 4.14.

Also, among the same five metrics, the most important is the 'Accuracy', the average ratio of the correctly predicted labels over the total labels including correctly and incorrectly predicted ones as well as those real labels but are missed out during the prediction. The mLysPTMpred achieves 83.73% accuracy which is considerable amount of higher than iPTM-mLys. In addition, the rate of "Aiming" or "Precision" [145] and the rate of "Coverage" or "Recall" [145] achieved by mLysPTMpred also better than iPTM-mLys.

Therefore, it is obvious from the Table 4.14, mLysPTMpred has performed remarkably better over iPTM-mLys [104] in all types of metrics measurement. To provide an intuitive comparison, a bar chart to represent the Table 4.14 is shown Figure 4.6. Therefore, it is projected that mLysPTMpred may become a useful and higher throughput tool in multiple lysine PTM sites predictions.

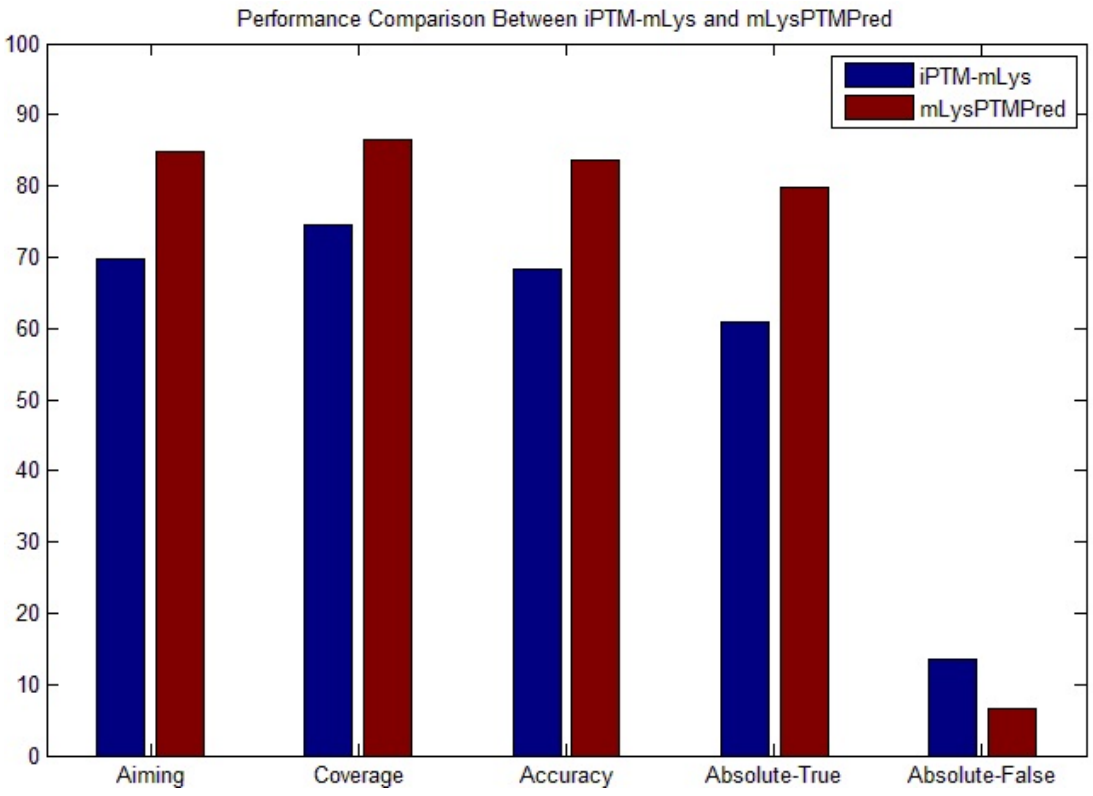

Figure 4.6: Performance comparison between iPTM-mLys and mLysPTMpred

In iPTM-mLys, an example protein sequence (Q16778) has been used to validate their findings. It should be noted here that the example protein sequence (Q16778) is also available in our site under Example button. For making comparison, we have not changed the sequence as example. The prediction result using this sequence (Q16778) from mLysPTMpred and the actual experimental result of this sequence is reported in Table 4.15. By putting this predicted result in equation 4.15, we have got the rate of aiming=88.33%, coverage=87.50%, accuracy=85.83%, absolute-true= 80.00% and absolute-false= 6.00% which is similar to the rates obtained by the cross-validation tests as given in Table 4.14.

Why can the proposed method enhance the prediction quality so significantly? First, the coupling effects among the amino acids around the target sites have been taken into account via the conditional probability as done by many investigators in successfully enhancing the prediction quality in some applications [55, 148, 149]. Second, the predictor used Different Error Costs (DEC) method to balance the effect of skewed training dataset and hence many false prediction events produced by imbalanced and skewed training datasets can be avoided as established in some recent studies [105, 51, 55, 148].

Table 4.15: Comparison between the predicted and experimental results on protein Q16778

Sites	Predicted Result				Experimental Result			
	Ace	Cro	Met	Suc	Ace	Cro	Met	Suc
6	Yes	Yes	Yes	No	Yes	Yes	No	No
12	Yes	Yes	No	No	Yes	Yes	No	No
13	Yes	Yes	No	No	Yes	Yes	No	No
16	Yes	Yes	No	No	Yes	Yes	No	No
17	Yes	Yes	No	No	Yes	Yes	No	No
21	Yes	Yes	No	No	Yes	Yes	No	No
24	Yes	Yes	No	No	Yes	Yes	No	No
25	No	No	No	No	No	No	No	No
28	No	No	No	No	No	No	No	No
29	No	No	No	No	No	No	No	No
31	No	No	No	No	No	No	No	No
35	No	Yes	No	No	No	Yes	No	No
44	No	No	No	No	No	No	No	No
47	No	No	Yes	No	No	No	Yes	No
58	No	No	Yes	No	No	No	Yes	No
86	Yes	No	Yes	No	Yes	No	Yes	No
109	No	No	Yes	No	No	No	Yes	No
117	Yes	No	No	No	No	No	No	No
121	Yes	No	No	No	No	No	No	No
126	Yes	No	No	No	No	No	No	No

4.6.3.3 Web Server

To attract more users especially for the experimental scientists and enhance the value of practical application, a user-friendly web-server for mLysPTMpred has been established at http://research.ru.ac.bd/mLysPTMpred/. In order to get the predicted result, users are required to submit protein sequence through the input text box in our site. The input sequence should follow the FASTA format. An example of a sequence of FASTA format is available under example button in our published site. Moreover, in order to get batch prediction, users are required to enter desired batch input file in the FASTA format. Noted that, the benchmark dataset used to train and test the mLysPTMpred predictor are available under Supporting Information button.

4.7 Experiment No 5:- Protein Subcellular Localization Prediction using Support Vector Machine with the Choice of Proper Kernel

4.7.1 Motivation and Goals

Prediction of subcellular locations of proteins can provide useful hints for revealing their functions as well as for understanding the mechanisms of some diseases and finally for developing novel drugs. As the number of newly discovered proteins has been growing exponentially, which in turns, makes the subcellular localization prediction by purely laboratory tests prohibitively laborious and expensive. In that context, to tackle the challenges, computational methods are developed as an alternative choice to help biologists in selecting target proteins and designing related experiments. However, the success of protein subcellular localization prediction is still a complicated and challenging problem, particularly when query proteins may have the multi-label characteristics, i.e., their simultaneous existence in more than one subcellular location or if they move between two or more different subcellular location as well. At this point, to get rid of this problem, many computational techniques, such as the neural network [34], K-nearest neighbor (KNN) [35, 36, 37], Bayesian [17, 31, 30] approach and a few ensemble classifiers [38, 39] have been introduced for the prediction of protein subcellular localization. In recent times, support vector machine (SVM) [14, 21, 33, 40] has also been extensively applied to provide potential solutions for the prediction of protein subcellular localization. However, the practicability of SVM is affected by the difficulties in selecting its appropriate kernel as well as in selecting the parameters of that selected kernel. The literature survey have showed that most of the researchers applied radial basis function (RBF) kernel to build SVM based subcellular localization prediction. Surprisingly still there are many other kernel functions which are not yet applied in protein subcellular localization prediction. But the nature of classification problem requires applying of different kernels for SVM to ensure optimal result.

From this viewpoint, this experiment presents the work to apply different kernels

for SVM in protein subcellular localization prediction to find out which kernel is the best for SVM. We have evaluated our system on a combined dataset containing 5447 single-localized proteins (originally published as part of the Höglund dataset) and 3056 multi-localized proteins (originally published as part of the DBMLoc set). It can be noted here that these dataset was used by Briesemeister et al. in their extensive comparison of multi-localization prediction systems [30].

4.7.2 Materials and Methods

4.7.2.1 Datasets

In our experiments we use a combined dataset containing 5447 single-localized proteins, originally published as part of the Höglund dataset [30] and 3056 multi-localized proteins that was originally published as part of the DBMLoc set [150]. This combined dataset was first constructed for an extensive comparison of multi-localization prediction systems by Briesemeister et al. [30]. This dataset is already homology-reduced, i.e. the protein sequences from the Höglund dataset share no more than 30% sequence identity with each other, and in the same time, sequences from the DBMLoc dataset share less than 80 % sequence similarity with each other. We report results using different evaluation metric that obtained over the set of multi-localized proteins for comparing our system to other published systems, since the results for these systems are only available for this subset. In the case, where reports obtained over the combined set of single- and multi-localized proteins from other system are available, we also make comparison with our system. The 5447 single-localized proteins covering the following 9 locations (abbreviations and number of proteins per location are given in parentheses): cytoplasm (cyt, 1411 proteins); endoplasmic reticulum (ER, 198), extra cellular space (ex, 843), golgi apparatus (gol, 150), lysosome (lys, 103), mitochondrion (mi, 510), nucleus (nuc, 837), membrane (mem, 1238), and peroxisome (per, 157). The multi-localized proteins come from the following pairs of locations: cyt and nuc (cyt_nuc , 1882 proteins), ex and pm (ex_mem , 334), cyt and mem (cyt_mem, 252), cyt and mi (cyt_mi, 240), nuc and mi (nuc_mi, 120), er and ex (ER_ex, 115), and ex and nuc (ex_nuc , 113). It should be noted that all the multi-location subsets used have over 100 representative proteins and this is currently the largest data set of proteins from multiple locations [62].

4.7.2.2 Biological Input Features of Protein

In this study, we have used the 30-dimensional features vector of protein as same as used by Briesemeister et al. for YLoc+ and R. Ramanuja Simha for MDLoc and BNCs [17, 31, 30, 151]. However, thirteen of those features constructed directly from the protein sequence such as length of the amino acid chain, length of the longest very hydrophobic region, respective number of Methionine, Asparagine, and Tryptophane, occurring in the N-terminus etc [17]. Again, nine of those features are extracted from pseudo-amino acid composition, which is based on certain physical and

chemical properties of amino acid subsequences. The remaining 8 feature were come from two types of annotation based features. Here, the first type annotation-based features contain two features constructed using two distinct groups of PROSITE patterns, and the second type annotation-based features contain six features extracted based on GO-annotations [17, 31].

4.7.2.3 Support Vector Machine and its Kernel

The multiclass multi-label SVM modeling algorithm has been used in this study. It is explained in section 2.2.1.3. In the study, we have used three kernels namely linear, polynomial, RBF, and laplace kernel. The definition of these kernels has been defined in section 2.2.1.1.

4.7.2.4 Evaluation Metrics

Performance measurement in multi-label classification is more complicated than traditional single-label classification, as each example could be associated with multiple labels simultaneously. In this study, we have used various types of adapted measures such as accuracy, F_1 score, Pre_{s_i}, Rec_{s_i}, $Pre\_Std_{s_i}$, $Rec\_Std_{s_i}$, and $F_1 - label$. The definitions of these metrics are discussed in section 4.2.3.3.

4.7.2.5 Experimental Setting

In this study, to save the computational time, we have used K-fold cross validation (subsampling) methods and compared the performance of our system (SVM with the best performed kernel based system) to that of other systems (MDLoc [31], BNCs [17], YLoc+ [30], Euk-mPLoc [28], WoLF PSORT [18], and $KnowPred_{site}$ [25]) and the systems based on other kernels. It should be noted that the performance of YLoc+, Euk-mPLoc, WoLF PSORT, and $KnowPred_{site}$ on a large set of multi-localized proteins have been studied comprehensively in [31]. As the information about the exact 5-way splits of dataset used in previous studies is not published, therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 5-fold cross validation for 5 times (i.e. 25 runs in total). It can be mentioned here that in each 5-fold cross validation the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally we have reported the average results in this study.

4.7.3 Results and Discussion

4.7.3.1 Model Selection of SVM

In order to generate highly performing SVM classifiers capable of dealing with real data an efficient model selection is required. Grid-search technique has been used to find the best model for SVM with different kernel in this work. In our experiments, this method selects the values of parameters considering highest multi-label accuracy

Table 4.16: Selected Parameters of 5 times run of the 5-fold cross-validation on combined set of single- and multi-localized proteins for each kernel (Linear, Polynomial, RBF, Laplace).

No. of Complete Run	Linear	Polynomial		RBF		Laplace	
	C	C	d	C	σ	C	σ
1^{st}	2^{-1}	2^{-2}	3	2^{1}	2^{1}	2^{8}	2^{3}
2^{nd}	2^{-1}	2^{1}	3	2^{2}	2^{1}	2^{5}	2^{2}
3^{rd}	2^{-4}	2^{-4}	3	2^{1}	2^{1}	2^{8}	2^{3}
4^{th}	2^{-4}	2^{1}	3	2^{1}	2^{1}	2^{8}	2^{3}
5^{th}	2^{-2}	2^{1}	3	2^{7}	2^{1}	2^{8}	2^{3}

and then time if more than one position in search space has the same multi-label accuracy. According to the experimental setting, we have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier on basis of the multi-label accuracy.

For linear kernel, to find the parameter value C (penalty term for soft margin), we have considered the values from 2^{-4} to 2^{4} as our searching space. The selected C of 5 complete runs of the 5-fold cross-validation on combined set of single- and multi-localized proteins is shown in Table 4.16. From the Table 4.16, it is seen that most of the times, the best model is found for the value of $C = 2^{-4}$ or $C = 2^{-1}$. Finally, we have used $C = 2^{-4}$ (using random selection between these two values) in all 5 complete runs of the 5-fold cross-validation and averaged our results in order to ensure unbiased model selection.

For polynomial kernel, to find the parameter value C (penalty term for soft margin) and d, we have considered the value from 2^{-4} to 2^{4} and from 1 to 3 for d as our searching space. The selected C and d of 5 complete runs of the 5-fold cross-validation on combined set of single- and multi-localized proteins is shown in Table 4.16. From the Table 4.16, it is seen that most of the times, the best model is found for the value of $C = 2^{1}$ and d=3. Finally, we have used $C = 2^{1}$ and d=3 in all 5 complete runs of the 5-fold cross-validation and averaged our results in order to ensure unbiased model selection.

For radial basis function (RBF) kernel, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8} to 2^{8} for C and from 2^{-8} to 2^{8} for sigma as our searching space. The selected C and sigma of 5 complete runs of the 5-fold cross-validation on combined set of single- and multi-localized proteins is shown in Table 4.16. From the Table 4.16, it is seen that most of

Table 4.17: Comparison of the results of multi-location prediction systems of different kernels, averaged over 5 times run of the 5-fold cross-validation applied on combined set of single-localized and multi-localized proteins.

	Linear	Polynomial	RBF	Laplace
F_1	0.613(±0.016)	0.677(±0.033)	0.810(± 0.017)	0.829(±0.002)
Acc	0.498(±0.016)	0.602(±0.031)	0.764(± 0.017)	0.786(±0.002)

the times, the best model is found for the value of $C = 2^1$ and $\sigma = 2^1$. Finally, we have used $C = 2^1$ and $\sigma = 2^1$ in all 5 complete runs of the 5-fold cross-validation and averaged our results in order to ensure unbiased model selection.

Again, for Laplace kernel, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8} to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. The selected C and sigma of 5 complete runs of the 5-fold cross-validation on combined set of single- and multi-localized proteins is shown in Table 4.16. From the Table 4.16, it is seen that most of the times, the best model is found for the value of $C = 2^8$ and $\sigma = 2^3$. Finally, we have used $C = 2^8$ and $\sigma = 2^3$ in all 5 complete runs of 5-fold cross-validation and averaged our results in order to ensure unbiased model selection.

4.7.3.2 Prediction Performance Evaluation

In this section, we have compared the performance of each kernel for SVM and also compared the performance of the best performed kernel (SVM with laplace kernel based system, called LKLoc) with that of existing location prediction systems. We have trained our system using combined dataset and measured two set of results, one set is for combined set of single and multi-localized proteins and another one is for multi-localized proteins only. It is noted that all the values of all metrics of our system are the average result of 5 complete runs of the 5-fold cross-validation. Moreover, standard deviations of each metrics of 5 complete runs of the 5-fold cross validation are shown in parentheses.

Table 4.17 shows comparisons of the F1 score and the accuracy obtained by each kernel used in SVM for combined dataset. The table shows that SVM with Laplace kernel based system, named LKLoc, performs better than other kernels. In addition, Table 4.18 shows the comparative studies of F_1 score and the accuracy obtained by LKLoc with those obtained by other multi-location predictors applied on combined dataset (BNCs as reported in Table 2 of Ramanuja Simha et al [17]). It is clear from this table that LKLoc provides better accuracy than the existing systems.

Table 4.19 shows comparative studies of the results of per-location predictions applied on combined dataset of both single- and multi-localized proteins obtained by

Table 4.18: Comparison of the results of multi-location prediction systems, averaged over 5 times run of the 5-fold cross-validation applied on combined set of single-localized and multi-localized proteins.

	LKLoc	BNCs
F_1	0.829($\pm$0.002)	0.81($\pm$0.01)
Acc	0.786($\pm$0.002)	0.76 ($\pm$0.01)

LKLoc and those obtained by MDLoc and BNCs [17, 31]. In the Table 4.19, the results are shown for the five locations with the largest number of associated proteins. It is obvious from the table, most of the cases the precision values provided by LKLoc are somewhat higher than those of MDLocs, and on the other hand, recall provided by LKLoc has a little bit variation (up and down) than those of MDLocs.

Table 4.20 shows comparisons of the F_1-label score and the accuracy obtained by the best performed kernel (Laplace kernel in this case) with those obtained by other multi-location predictors for multi-localized proteins only (MDLoc [31], BNCs [17], YLoc+ [30], Euk-mPLoc [28], WoLF PSORT [18], and $KnowPred_{site}$ [25] as reported in Table 1 of Ramanuja Simha et al. [31]). It can be noted here that all the predictors mentioned above used the same set of multi-localized proteins. The table shows that the prediction based on SVM with Laplace kernel or LKLoc performs better than the existing top-systems, including MDLoc, YLoc+, and BNCs which have the best performance reported so far.

Table 4.21 shows the per-location prediction results for multilocalized proteins obtained by LKLoc compared with those systems reported by MDLoc [31]. Since the per-location predictions for the other systems (BNCs, Euk-mPLoc, WoLF PSORT and $KnowPred_{site}$) are not publicly available, as a result, we could not show those findings. In the Table 4.21, the results are shown for the five locations with the largest number of associated proteins. However, for each location s_i, we show Multilabel-Precision (Pre_{s_i})) and Multilabel-Recall (Rec_{s_i}) as well as standard precision ($Pre - Std_{s_i}$) and recall ($Rec - Std_{s_i}$). The results shows that, almost all of the cases, these four measures obtained from LKLoc are significantly higher than those obtained using MDLoc and YLoc+ for all protein locations.

Table 4.19: Per-locations based results, averaged over 5 times run of the 5-fold cross-validation applied on combined dataset.

Metrics	Predictor	cyt (3785)	nuc (2952)	ex (1405)	mem (1824)	mi (870)
Rec_{s_i}	LKLoc	0.829 (±0.008)	0.818 (±0.015)	0.818 (±0.005)	0.800 (±0.002)	0.753 (±0.004)
	MDLoc	0.825 (±0.009)	0.830 (±0.010)	0.780 (±0.020)	0.822 (±0.012)	0.773 (±0.013)
	BNCs	0.795 (±0.011)	0.784 (±0.017)	0.737 (±0.022)	0.780 (±0.014)	0.730 (±0.025)
Pre_{s_i}	LKLoc	0.833 (±0.004)	0.843 (±0.005)	0.886 (±0.009)	0.864 (±0.001)	0.868 (±0.005)
	MDLoc	0.819 (±0.013)	0.822 (±0.014)	0.864 (±0.020)	0.872 (±0.014)	0.861 (±0.024)
	BNCs	0.809 (±0.018)	0.832 (±0.013)	0.912 (±0.019)	0.900 (±0.012)	0.885 (±0.023)
$Rec - Std_{s_i}$	LKLoc	0.890 (±0.003)	0.764 (±0.027)	0.837 (±0.009)	0.740 (±0.004)	0.713 (±0.009)
	MDLoc	0.867 (±0.015)	0.808 (±0.021)	0.715 (±0.030)	0.842 (±0.017)	0.719 (±0.028)
	BNCs	0.861 (±0.014)	0.736 (±0.031)	0.652 (±0.024)	0.805 (±0.017)	0.664 (±0.034)
$Pre - Std_{s_i}$	LKLoc	0.855 (±0.004)	0.814 (±0.008)	0.907 (±0.009)	0.831 (±0.003)	0.868 (±0.003)
	MDLoc	0.854 (±0.014)	0.783 (±0.020)	0.839 (±0.028)	0.882 (±0.014)	0.843 (±0.026)
	BNCs	0.840 (±0.011)	0.786 (±0.026)	0.906 (±0.022)	0.900 (±0.015)	0.873 (±0.034)

Table 4.20: Multi-location prediction results, averaged over 5 times run of the 5-fold cross-validation, for multi-localized proteins only.

	LKLoc	MDLoc	BNCs	YLoc+	Euk-mPLoc	WoLF PSORT	$KnowPred_{site}$
$F_1 - label$	0.741 ($\pm$0.004)	0.71 ($\pm$0.02)	0.66 ($\pm$0.02)	0.68	0.44	0.53	0.66
Acc	0.700 ($\pm$0.010)	0.68 ($\pm$0.01)	0.63 ($\pm$0.01)	0.64	0.41	0.43	0.63

4.8 Experiment No 6:- Protein Subcellular Localization Prediction Using Multiple Kernel Learning Based Support Vector Machine

4.8.1 Motivation and Goals

Prediction of subcellular locations of proteins can provide useful hints for revealing their functions as well as for understanding the mechanisms of some diseases and finally for developing novel drugs. As the number of newly discovered proteins has been growing exponentially, which in turns, makes the subcellular localization prediction by purely laboratory tests prohibitively laborious and expensive. In that context, to tackle the challenges, computational methods are developed as an alternative choice to help biologists in selecting target proteins and designing related experiments. However, the success of protein subcellular localization prediction is still a complicated and challenging problem, particularly when query proteins may have the multi-label characteristics, i.e., their simultaneous existence in more than one subcellular location or if they move between two or more different subcellular location as well. At this point, to address this problem, many computational techniques, such as the neural network [34], K-nearest neighbor (KNN) [35, 36, 37], Bayesian [17, 31, 30] approach and a few ensemble classifiers [38, 39] have been introduced for the prediction of protein subcellular localization. Some methods use variations of k-NN to predict multiple locations for proteins such as WoLF PSORT [18] uses k-NN with a distance measure that combines Euclidean and Manhattan distances, Euk-mPLoc [28] uses an ensemble of k-NN. Again, $KnowPred_{site}$ [25] uses sequence-based similarity to create a collection of location-annotated peptide fragments and predict multiple locations for proteins. However, in recent days, the support vector machine (SVM) [14, 21, 33, 40] has also been extensively applied to provide potential solutions for the subcellular localization prediction. But, the selection of an appropriate kernel and its parameters for a certain classification problem influence the performance of the SVM. The selection of the appropriate kernel and kernel parameters are both considered as the choice of kernel

Table 4.21: Per-location based results, averaged over 5 times run of the 5-fold cross-validation, for multi-localized proteins only.

Metrics	Predictor	cyt (2374)	nuc (2115)	mem (586)	ex (562)	mi (360)
Rec_{s_i}	LKLoc	0.750 (±0.014)	0.776 (±0.017)	0.557 (±0.009)	0.590 (±0.008)	0.527 (±0.006)
	MDLoc	0.750 (±0.012)	0.776 (±0.014)	0.527 (±0.022)	0.547 (±0.035)	0.519 (±0.026)
	YLoc+	0.712 (±0.009)	0.728 (±0.011)	0.543 (±0.018)	0.573 (±0.026)	0.536 (±0.031)
Pre_{s_i}	LKLoc	0.934 (±0.003)	0.944 (±0.0006)	0.870 (±0.013)	0.917 (±0.008)	0.868 (±0.014)
	MDLoc	0.911 (±0.008)	0.929 (±0.008)	0.807 (±0.036)	0.833 (±0.044)	0.832 (±0.042)
	YLoc+	0.893 (±0.010)	0.924 (±0.008)	0.764 (±0.029)	0.740 (±0.053)	0.765 (±0.033)
$Rec - Std_{s_i}$	LKLoc	0.849 (±0.004)	0.700 (±0.034)	0.615 (±0.020)	0.440 (±0.009)	0.431 (±0.017)
	MDLoc	0.817 (±0.021)	0.746 (±0.028)	0.588 (±0.042)	0.385 (±0.058)	0.388 (±0.062)
	YLoc+	0.786 (±0.020)	0.684 (±0.015)	0.614 (±0.042)	0.401 (±0.037)	0.429 (±0.060)
$Pre - Std_{s_i}$	LKLoc	0.950 (±0.002)	0.929 (±0.001)	0.867 (±0.013)	0.921 (±0.006)	0.829 (±0.013)
	MDLoc	0.942 (±0.009)	0.904 (±0.014)	0.794 (±0.039)	0.830 (±0.046)	0.784 (±0.057)
	YLoc+	0.935 (±0.009)	0.914 (±0.014)	0.730 (±0.047)	0.771 (±0.055)	0.670 (±0.055)

problem [3, 4]. The literature survey have showed that most of the researchers applied radial basis function (RBF) kernel to build SVM based subcellular localization prediction [14, 16, 39] and have found the value of its parameter by using different techniques such as trial and error, heuristics or grid search procedure, unfortunately, these approaches are time consuming[41]. To reduce the complexity in finding kernel parameters such as sigma for RBF (choice of kernel from the set of RBF kernel), multiple kernel learning approach can be adopted [41]. In this case, the set of radial basis function (RBF) kernels (different values of sigma create different kernels) has been considered as the search space of the choice of kernel problem [3]. However, Several researchers have proposed various multiple kernel learning approach in which multiple kernels are used to construct a combined kernel [41, 3, 75, 7, 8, 44, 85].

This experiment is aimed to develop an efficient multi-label protein subcellular localization prediction system, named MKLoc, by introducing multiple kernel learning (MKL) based SVM. We have evaluated MKLoc on a combined dataset containing 5447 single-localized proteins (originally published as part of the Höglund dataset) and 3056 multi-localized proteins (originally published as part of the DBMLoc set). It can be noted here that this dataset was used by Briesemeister et al. in their extensive comparison of multi-localization prediction systems [30].

4.8.2 Materials and Methods

4.8.2.1 Datasets

In our experiments we use a combined dataset containing 5447 single-localized proteins, originally published as part of the Höglund dataset [30] and 3056 multi-localized proteins that was originally published as part of the DBMLoc set [150]. This combined dataset was first constructed for an extensive comparison of multi-localization prediction systems by Briesemeister et al. [30]. This dataset is already homology-reduced, i.e. the protein sequences from the Höglund dataset share no more than 30% sequence identity with each other, and in the same time, sequences from the DBMLoc dataset share less than 80 % sequence similarity with each other. We report results using different evaluation metric that obtained over the set of multi-localized proteins for comparing our system to other published systems, since the results for these systems are only available for this subset. In the case, where reports obtained over the combined set of single- and multi-localized proteins from other system are available, we also make comparison with our system. The 5447 single-localized proteins covering the following 9 locations (abbreviations and number of proteins per location are given in parentheses): cytoplasm (cyt, 1411 proteins); endoplasmic reticulum (ER, 198), extra cellular space (ex, 843), golgi apparatus (gol, 150), lysosome (lys, 103), mitochondrion (mi, 510), nucleus (nuc, 837), membrane (mem, 1238), and peroxisome (per, 157). The multi-localized proteins come from the following pairs of locations: cyt and nuc (cyt_nuc , 1882 proteins), ex and pm (ex_mem , 334), cyt and mem (cyt_mem, 252), cyt and mi (cyt_mi, 240), nuc and mi (nuc_mi, 120), er and ex (ER_ex, 115),

and ex and nuc (ex_nuc , 113). It should be noted that all the multi-location subsets used have over 100 representative proteins and this is currently the largest data set of proteins from multiple locations [62].

4.8.2.2 Biological Input Features of Protein

In this study, we have used the 30-dimensional features vector of protein as same as used by Briesemeister et al. for YLoc+ and R. Ramanuja Simha for MDLoc and BNCs [17, 31, 30, 151]. However, thirteen of those features constructed directly from the protein sequence such as length of the amino acid chain, length of the longest very hydrophobic region, respective number of Methionine, Asparagine, and Tryptophane, occurring in the N-terminus etc [17]. Again, nine of those features are extracted from pseudo-amino acid composition [19], which is based on certain physical and chemical properties of amino acid subsequences. The remaining 8 feature were come from two types of annotation based features. Here, the first type annotation-based features contain two features constructed using two distinct groups of PROSITE patterns, and the second type annotation-based features contain six features extracted based on GO-annotations [17, 31].

4.8.2.3 Support Vector Machine and its Kernel

The multiclass multi-label SVM modeling algorithm has been used in this study. It is explained in section 2.2.1.3. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.8.2.4 Multiple Kernel Learning

A successful application of SVMs depends heavily on the determination of the right type and suitable parameter settings of kernel functions [41]. The selection of the appropriate kernel and kernel parameters are both considered as the choice of kernel problem [3, 4]. However, Hsu et al.[64] claimed the RBF kernel to be a reasonable first choice for the SVM. Again, various other researchers have used trial and error, heuristic or grid search procedures to determine the settings of the hyperparameters of a kernel [65, 152]. This obviously takes a lot of efforts.

In order to avoid, the time consuming parameter tuning problem (choice of kernel among the set of RBF kernels), one approach of MKL, which is adopted by many practitioners, is to discretize the parameter space (space of sigma for RBF kernel) into r values and then find an appropriate combination of the resulting set of base kernels, $S = k_{\sigma_1}, k_{\sigma_2}, \cdots, k_{\sigma_r}$ [3]. The advantage of this approach is that once the set S is fixed, any of the standard MKL methods available in the literature can be used to find the coefficients for combining the base kernels in S [41, 10, 43, 8, 85, 153, 9, 79, 3, 46, 45]. In this work, we have used RBF kernel and applied MKL for solving the problem of choosing sigma of RBF kernel.

The form of linear combined kernel of multiple kernel learning (MKL) is defined below [10]:

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i^m, x_j^m) \tag{4.16}$$

where η_m denotes the kernel weights, $k_m(x_i^m, x_j^m)$ take P feature representations (not necessarily different) of data instances, $x_i^m \in R^{D_m}$, and D_m is the dimensionality of the corresponding feature representation.

In our research work, in order to get the combination function parameters or kernel weights, a heuristic approach proposed by Qiu and Lane [85] has been used that uses kernel alignment based similarity measure to find the kernel weights. Note that kernel alignment based similarity measure is discussed in section 2.3.2.2 and 2.3.2.3.

Qiu and Lane [85] propose the following simple heuristic for classification problems to select the kernel weights using kernel alignment:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P} K_h, yy^T} \qquad \forall m \tag{4.17}$$

where we obtain the combined kernel as a convex combination of the input kernels.

4.8.2.5 Implementation Procedures of the Proposed System

To implement the proposed system, we have followed the following steps:

I Discretize the parameter space of sigma of RBF kernel into 17 values in order to get the set of base kernels, $S = k_{\sigma_1}, k_{\sigma_2}, \cdots, k_{17}$. In our work, 17 values of sigma for base kernels is $2^{-8}, 2^{-7}, \cdots, 2^7, 2^8$

II Find kernel weights for each of these 17 kernels using equation 4.17.

III Combine these 17 kernels using equation 4.16 in order to get the combined kernel.

IV N (N=9 in our case) independent binary SVMs are trained, one for each location, using the combined kernel.

V Make prediction of a query protein through learned SVM using equations 2.8 and 2.9 defined in section 2.2.1.3.

4.8.2.6 Experimental Setting

In this study, to save the computational time, we have used K-fold cross validation (subsampling) methods and compared the performance of MKLoc to that of other systems (MDLoc [31], BNCs [17], YLoc+ [30], Euk-mPLoc [28], WoLF PSORT [18], and $KnowPred_{site}$ [25]). It should be noted that the performance of YLoc+, Euk-mPLoc, WoLF PSORT, and $KnowPred_{site}$ on a large set of multi-localized proteins have been studied comprehensively in [31]. As the information about the exact 5-way

splits of dataset used in previous studies is not published, therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 5-fold cross validation for 5 times (i.e. 25 runs in total). It can be mentioned here that in each 5-fold cross validation the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally we have reported the average results in this study.

4.8.2.7 Evaluation Metrics

Performance measurement in multi-label classification is more complicated than traditional single-label classification, as each example could be associated with multiple labels simultaneously. In this study, we have used various types of adapted measures such as accuracy, F_1 score, Pre_{s_i}, Rec_{s_i}, $Pre\_Std_{s_i}$, $Rec\_Std_{s_i}$, and $F_1 - label$. The definitions of these metrics are discussed in section 4.2.3.3.

4.8.3 Results and Discussion

4.8.3.1 Model Selection of SVM

In order to generate highly performing classifiers capable of dealing with real data an efficient model selection is required. In our experiment, grid-search technique has been used to find the best model. This method selects the best solution by evaluating several combinations of possible values of parameters of the classifier. We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier on basis of the multi-label accuracy.

In SVM with multiple kernel, we have used combined kernel that will be formed using equation 4.16 from a set of RBF kernel with different values of sigma. Here, in our work, we have discretized the parameter space of sigma of RBF kernel into 17 values in order to get the set of base kernels, $S = k_{\sigma_1}, k_{\sigma_2}, \cdots, k_{\sigma_{17}}$. We have also considered the resulting set of sigma for base kernels is $2^{-8}, 2^{-7}, \cdots, 2^7, 2^8$.

Next, to find the parameter value C (penalty term for soft margin), we have considered the value from 2^{-8} to 2^8 for C as our searching space. We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier depending on the value of accuracy. Here, in our work, this model selection is applied on combined set of single- and multi-localized proteins. In this way, the selected C for 5 complete runs of the 5-fold cross-validation shown in Table 4.22.

From Table 4.22, we see that most of the times, best model is found for the value of $C = 2^5$. Finally, we have used $C = 2^5$ in all 5 complete runs of the 5-fold cross-validation and averaged our results on combined dataset in order to ensure unbiased model selection.

We have also implemented and tuned parameters (C and sigma for RBF Kernel) for single kernel based SVM using the same procedure as like MKL based SVM and finally,

Table 4.22: Selected C for 5 times run of the 5-fold cross-validation on combined set of single- and multi-localized proteins.

No. of Completes Run	C
1^{st}	2^6
2^{nd}	2^5
3^{rd}	2^5
4^{th}	2^6
5^{th}	2^5

used $C = 2^0$ and $\sigma = 2^1$ in all 5 times 5 fold cross validation run and averaged our results on combined dataset.

4.8.3.2 Prediction Performance Evaluation

In this section, we have compared the performance of MKLoc on 'multi-localized proteins' as well as on 'combined set of single and multi-localized proteins' with that of existing location prediction systems. We have trained our system using combined dataset and measured two set of results, one set is for multi-localized proteins only and another one is for combined set of single and multi-localized proteins.

Table 4.23 shows comparisons of the F1-label score and the accuracy obtained by MKLoc with those obtained by other multi-location predictors for multi-localized proteins only (MDLoc [31], BNCs [17], YLoc+ [30], Euk-mPLoc [28], WoLF PSORT [18], and $KnowPred_{site}$ [25] as reported in Table 1 of Ramanuja Simha et al. [31]). In addition to the tabular presentation shown in Table 4.23, comparisons of the F_1-label and Acc results are also graphically shown in Fig. 4.7 and 4.8, respectively. It can be noted here that all the predictors mentioned above used the same set of multi-localized proteins. The table as well as the figures show that MKLoc performs better than the existing top-systems, including MDLoc, YLoc+, and BNCs which have the best performance reported so far.

Table 4.23: Multi-location prediction results, averaged over 5 times run of the 5-fold cross-validation, for multi-localized proteins only.

	MKLoc	MDLoc	BNCs	YLoc+	Euk-mPLoc	WoLF PSORT	$KnowPred_{site}$
$F_1 - label$	0.727 ($\pm$0.005)	0.71 ($\pm$0.02)	0.66 ($\pm$0.02)	0.68	0.44	0.53	0.66
Acc	0.694 ($\pm$0.003)	0.68 ($\pm$0.01)	0.63 ($\pm$0.01)	0.64	0.41	0.43	0.63

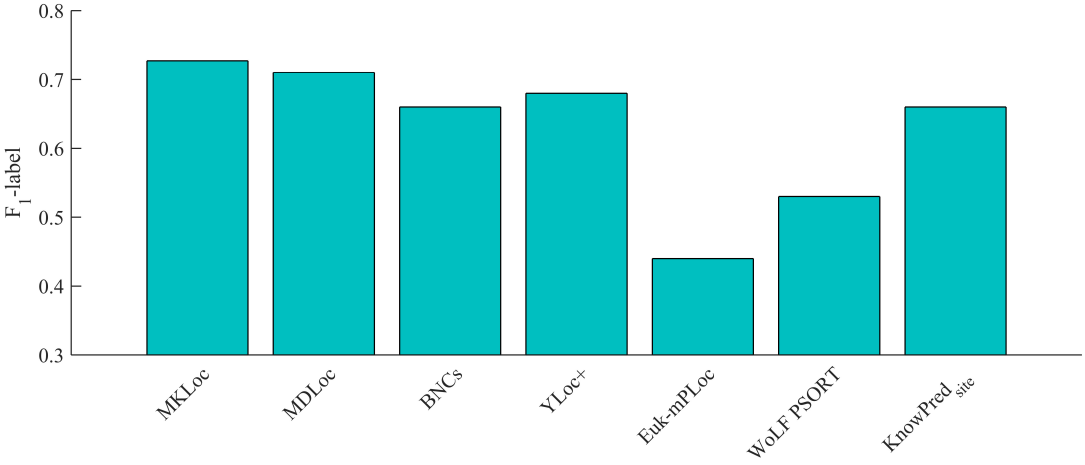

Figure 4.7: Multi-location prediction results for multi-localized proteins only when considering the evaluation metric F_1-label.

Table 4.24 shows the per-location prediction results for multilocalized proteins obtained by MKLoc compared with those systems reported by MDLoc [31]. Since the per-location predictions for the other systems (BNCs , Euk-mPLoc, WoLF PSORT and $KnowPred_{site}$) are not publicly available, as a result, we could not show those findings. In the Table 4.24, the results are shown for the five locations with the largest number of associated proteins. However, for each location s_i, we show Multilabel-Precision (Pre_{s_i}) and Multilabel-Recall (Rec_{s_i}) as well as standard precision ($Pre\_Std_{s_i}$) and ($Rec\_Std_{s_i}$) . The results shows that, almost all of the cases, these four measures obtained from MKLoc are significantly higher than those obtained using MDLoc and YLoc+ for all protein locations.

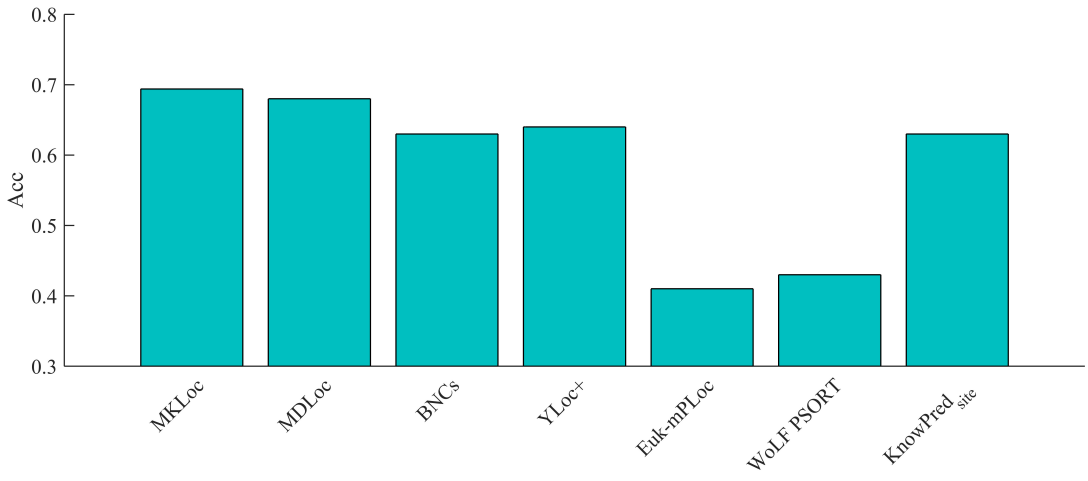

Figure 4.8: Multi-location prediction results for multi-localized proteins only when considering the evaluation metric Acc.

Table 4.25 shows the comparative studies of F_1 score and the accuracy obtained by MKLoc with those obtained by other multi-location predictors for combined dataset (single kernel based SVM and BNCs as reported in Table 2 of Ramanuja Simha et al [17]). It is clear from this table that MKLoc provides better accuracy than the existing systems. Moreover, for the experiments described in this work, we have run 5 times run of the 5-fold cross validation, where the total run time including tuning parameter is nearly 30 hours (wall clock) which is about half of time required for BNCs implementation. In addition to it, another finding from this table is that, the F_1 and accuracy found from our implementation with single kernel based SVM are better than those from single kernel SVM [17] and very close to MKLoc. The main problem of single kernel based SVM is that the required execution time is about double of MKLoc, as shown in Table 4.25. It should be mentioned that the F_1, accuracy and execution time for the other system [17, 31] except BNCs are not compared as they are not publicly available.

Table 4.26 shows comparative studies of the results of per-location predictions for combined dataset of both single- and multi-localized proteins obtained by MKLoc and those obtained by MDLoc and BNCs [17, 31]. It is obvious from the table, MKLoc's recall values are somewhat lower than those of MDLocs, on the other hand, MKLoc's precision is typically higher.

The following point may be the reason why the proposed predictor can improve the success rates over single kernel based SVM. MKLoc uses combined kernel which has been derived from all the kernels with different rates of contribution from the set of base kernels defined in model selection section 4.8.3.1 to build SVM based classifier. Here it should be noted that the influences or weights of all kernel has been derived using MKL. On the other hand, the main reasons for taking less time than single kernel based SVM is that it is avoiding the grid search operation for the value of sigma which is mandatory for single kernel based SVM.

Table 4.24: Per-location based results, averaged over 5 times run of the 5-fold cross-validation, for multi-localized proteins only.

Metrics	Predictor	cyt (2374)	nuc (2115)	mem (586)	ex (562)	mi (360)
Rec_{s_i}	MKLoc	0.749 (±0.003)	0.778 (±0.002)	0.534 (±0.004)	0.572 (±0.002)	0.513 (±0.002)
	MDLoc	0.750 (±0.012)	0.776 (±0.014)	0.527 (±0.022)	0.547 (±0.035)	0.519 (±0.026)
	YLoc+	0.712 (±0.009)	0.728 (±0.011)	0.543 (±0.018)	0.573 (±0.026)	0.536 (±0.031)
Pre_{s_i}	MKLoc	0.936 (±0.001)	0.944 (±0.001)	0.886 (±0.004)	0.925 (±0.009)	0.840 (±0.014)
	MDLoc	0.911 (±0.008)	0.929 (±0.008)	0.807 (±0.036)	0.833 (±0.044)	0.832 (±0.042)
	YLoc+	0.893 (±0.010)	0.924 (±0.008)	0.764 (±0.029)	0.740 (±0.053)	0.765 (±0.033)
$Rec - Std_{s_i}$	MKLoc	0.834 (±0.003)	0.713 (±0.003)	0.606 (±0.006)	0.416 (±0.002)	0.429 (±0.014)
	MDLoc	0.817 (±0.021)	0.746 (±0.028)	0.588 (±0.042)	0.385 (±0.058)	0.388 (±0.062)
	YLoc+	0.786 (±0.020)	0.684 (±0.015)	0.614 (±0.042)	0.401 (±0.037)	0.429 (±0.060)
$Pre - Std_{s_i}$	MKLoc	0.955 (±0.001)	0.928 (±0.003)	0.879 (±0.008)	0.928 (±0.009)	0.802 (±0.006)
	MDLoc	0.942 (±0.009)	0.904 (±0.014)	0.794 (±0.039)	0.830 (±0.046)	0.784 (±0.057)
	YLoc+	0.935 (±0.009)	0.914 (±0.014)	0.730 (±0.047)	0.771 (±0.055)	0.670 (±0.055)

Table 4.25: Comparison of the results of Multi-location prediction systems, averaged over 5-times run of the 5-fold cross-validation applied on combined set of single-localized and multi-localized proteins.

	MKLoc	Our Implementation of SVM with Single Kernel	Single Kernel SVM [17]	BNCs
F_1	0.814($\pm$0.001)	0.810($\pm$0.017)	0.77($\pm$0.01)	0.81($\pm$0.01)
Acc	0.771($\pm$0.002)	0.764($\pm$0.017)	0.72 ($\pm$0.01)	0.76($\pm$0.01)
Running Time	30 Hours (including run time of model selection)	75 Hours (including run time of model selection)	Data is not available	75 Hours

4.9 Experiment No 7:- Protein Subcellular Localization Prediction Using Kernel Based Feature Fusion

4.9.1 Motivation and Goals

The success of protein subcellular localization prediction is a complicated and challenging problem, particularly when query proteins may have multiple feature representation and multiplex character, i.e., simultaneously exist at, or move between, two or more different subcellular location sites. To get rid of this problem, several types of subcellular localization prediction methods have been proposed depending on various feature extractions methods which produce different levels of accuracy. Conventional methods for subcellular localization prediction can be roughly divided into sequence-based methods and annotation-based methods [14, 17]. Sequence-based predictors make use of (I) sequence-coded sorting signals such as WoLF PSORT [18] (II) amino acid composition information [19, 20], (III) both information sources [18, 21]. Annotation-based predictors use information about functional domains and motifs [23], protein-protein interaction [24], homologous proteins such as $KnowPred_{site}$ [25], annotated Gene Ontology (GO) terms [26] such as Euk-OET-PLoc [27], Euk-mPLoc [28], iLoc-Gneg [29].

Among the feature extractions methods, feature extractions based on GO terms provides better accuracy [14, 16, 154, 155, 135, 156, 157]. However, there are several cases, especially for newly discovered proteins, where the GO term feature representation are not available, in some cases GO terms of top homology are not found [16, 156] and finally in some worse cases, the homology of proteins are not available too [156], all these proteins from all the cases are called as 'non-GO termed' proteins. In such cases, researcher depends on some backup methods using other features extraction ap-

Table 4.26: Per-locations based results, averaged over 5 times run of the 5-fold cross-validation applied on combined dataset.

Metrics	Predictor	cyt (3785)	nuc (2952)	ex (1405)	mem (1824)	mi (870)
Rec_{s_i}	MKLoc	0.825 (±0.001)	0.82 (±0.001)	0.805 (±0.003)	0.776 (±0.003)	0.741 (±0.005)
	MDLoc	0.825 (±0.009)	0.830 (±0.010)	0.780 (±0.020)	0.822 (±0.012)	0.773 (±0.013)
	BNCs	0.795 (±0.011)	0.784 (±0.017)	0.737 (±0.022)	0.780 (±0.014)	0.730 (±0.025)
Pre_{s_i}	MKLoc	0.835 (±0.002)	0.842 (±0.001)	0.905 (±0.002)	0.883 (±0.003)	0.79 (±0.004)
	MDLoc	0.819 (±0.013)	0.822 (±0.014)	0.864 (±0.020)	0.872 (±0.014)	0.861 (±0.024)
	BNCs	0.809 (±0.018)	0.832 (±0.013)	0.912 (±0.019)	0.900 (±0.012)	0.885 (±0.023)
$Rec - Std_{s_i}$	MKLoc	0.878 (±0.002)	0.773 (±0.002)	0.828 (±0.004)	0.713 (±0.002)	0.706 (±0.008)
	MDLoc	0.867 (±0.015)	0.808 (±0.021)	0.715 (±0.030)	0.842 (±0.017)	0.719 (±0.028)
	BNCs	0.861 (±0.014)	0.736 (±0.031)	0.652 (±0.024)	0.805 (±0.017)	0.664 (±0.034)
$Pre - Std_{s_i}$	MKLoc	0.863 (±0.003)	0.806 (±0.001)	0.924 (±0.003)	0.846 (±0.004)	0.789 (±0.005)
	MDLoc	0.854 (±0.014)	0.783 (±0.020)	0.839 (±0.028)	0.882 (±0.014)	0.843 (±0.026)
	BNCs	0.840 (±0.011)	0.786 (±0.026)	0.906 (±0.022)	0.900 (±0.015)	0.873 (±0.034)

proaches but unfortunately the reason of selecting their feature extraction approach is not explained. It can be noted here that in most of the cases, prediction performance of only the backup method is not provided separately, that is, combined prediction performance is given based on GO term based method along with the backup method. This makes it harder to get any idea about the prediction performance of the non-GO termed proteins.

Therefore, in this experiment, our main goal is to develop an efficient prediction system using feature representation other than GO term feature representation for predicting the subcellular location of proteins especially for non-GO termed proteins. Keeping this goal in mind, we have considered seven types of feature extraction approaches for the gram-negative dataset. Finally, we have developed a subcellular location prediction system , named KFFLoc-Gneg, using kernel based feature fusion through multiple kernel learning (MKL) based support vector machine (SVM).

4.9.2 Materials and Methods

4.9.2.1 Short Description of Dataset and Working Procedure of the Proposed System

In this paper, the gram-negative bacterial benchmark dataset used in Gneg-mPLoc [135], iLoc-Gneg [29] and Gneg-ECC-mPLoc [63] is used to evaluate the prediction performance of our system. The gram-negative bacterial dataset contains 1392 different proteins, called actual proteins, which are distributed in 8 locations. Among these gram-negative proteins, 1328 belong to one subcellular location, 64 to two locations, and none to more locations. Hence, there are 1456 (1328+64 *2) locative proteins in total in this dataset. The concept of locative proteins and actual proteins has been explained in detail in literature [13, 14, 16, 63]. The name of these eight locations and the number of proteins in each location are shown at Table 4.27. The pairwise sequence identity among proteins in this dataset is controlled fewer than 25%. This benchmark is available at: http://www. csbio.sjtu.edu.cn/bioinf/Gneg-multi/.

The proposed predictor, called predMultiLoc-Gneg, has followed the following steps:

1. Extract seven types of features using various types of feature extraction approaches.

2. Integrate data sources and train a system using multiple kernel learning based support vector machine. Here, multiple kernel learning has been used to do kernel based feature fusion.

To provide an intuitive view, the working procedure of our system is shown in Figure 4.9.

Table 4.27: Distribution of protein subcellular locations in the gram-negative bacterial dataset

No.	Subcellular location	No. of proteins
1	Cell inner membrane	557
2	Cell outer membrane	124
3	Cytoplasm	410
4	Extracellular	133
5	Fimbrium	32
6	Flagellum	12
7	Nucleoid	8
8	Periplasm	180
Total number of locative proteins		1456
Total number of different proteins		1392

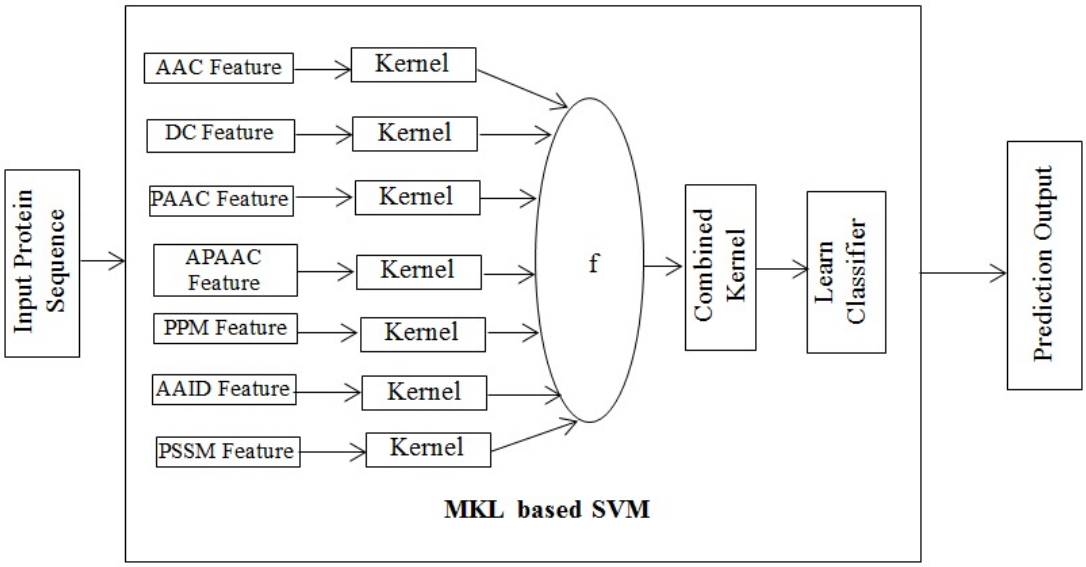

Figure 4.9: A flowchart to show the prediction process of KFFLoc-Gneg

4.9.2.2 Feature Extractions

In this study, we have considered the following seven approaches to extract feature from protein sequences.

1. Amino Acid Composition (AAC) [See section 3.6.5.1]
2. Dipeptide Composition (DC) [See section 3.6.5.2]
3. Pseudo-Amino Acid Composition (PAAC) [See section 3.6.5.3]
4. Amphiphilic Pseudo-Amino Acid Composition (APAAC) [See section 3.6.5.4]
5. Physicochemical Properties Model (PPM) [See section 3.6.5.5]
6. Amino Acid Index Distribution (AAID) [See section 3.6.5.6]
7. Sequential Evolution Information (PSSM) [See section 3.6.5.8]

4.9.2.3 Support Vector Machine and its Kernel

The multiclass multi-label SVM modeling algorithm has been used in this study. It is explained in section 2.2.1.3. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.9.2.4 Multiple Kernel Learning

Multiple kernel learning (MKL) based SVM is as an extension of single kernel based SVM to incorporate multiple kernels in classification [41, 10, 43]. There are two well-known problem formulations of MKL namely one-stage MKL and two-stage MKL [10, 45]. In this paper, the two-stage MKL problem formulation has been used. In two-stage MKL based system, at first, an individual kernel on each set of features are defined, after that these individual kernels are combined into a single kernel using specific rules of MKL, and finally, trained a single SVM on that combined kernel.

The form of linear combined kernel of multiple kernel learning (MKL) is defined below [10]

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i^m, x_j^m) \tag{4.18}$$

where η_m denotes the kernel weights, $k_m(x_i^m, x_j^m)$ take P feature representations (not necessarily different) of data instances, $x_i^m \in R^{D_m}$, and D_m is the dimensionality of the corresponding feature representation.

In our research work, in order to get the combination function parameters or kernel weights, a heuristic approach proposed by Qiu and Lane [85] has been used that uses kernel alignment based similarity measure to find the kernel weights. Note that kernel alignment based similarity measure is discussed in section 2.3.2.2 and 2.3.2.3. Qiu and Lane [85] propose the following simple heuristic for classification problems to select the kernel weights using kernel alignment:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P} K_h, yy^T} \qquad \forall m$$

where we obtain the combined kernel as a convex combination of the input kernels.

4.9.2.5 Imbalance Data Management

In this work, we have used a hybrid model using Random oversampling and Different Error Costs (DEC), defined in section 4.2.1.2, to handle imbalance dataset problem of gram-negative dataset. In our work, equations 4.1 has been used to assign the cost for the positive and negative classes in the DEC method.

4.9.2.6 Experimental Setting

In this study, to save the computational time, we have used subsampling (K-fold cross validation) methods and compared the performance among our developed classifiers. The problem with this kind of subsampling test is that the number of possible selections in dividing a benchmark dataset is an astronomical figure even for a very simple dataset [158]. Therefore, in any actual subsampling cross-validation tests, only an extremely small fraction of the possible selections are taken into account. Since different selections will always lead to different results even for a same benchmark dataset and a same predictor, the subsampling test cannot provide a unique outcome. To reduce the problem of unique outcome, we have repeated the 5-fold cross validation for 5 times (i.e. 25 runs in total), where the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally we have reported average result of 5 complete runs of the 5-fold cross-validation. The use of multiple runs with different splits helps validate the stability and the statistical significance of the results.

4.9.2.7 Measuring Metrics

In this work we have used the (overall) locative and absolute accuracy to measure the performance of multi-label predictors. The formulation of locative proteins and actual proteins has been define in section 4.2.3.1.

4.9.3 Results and Discussion

4.9.3.1 Model Selection of SVM (Both Single and Multiple Kernel Based SVM)

In order to generate highly performing classifiers capable of dealing with real data an efficient model selection is required. In our experiment, grid-search technique has been used to find the best model for support vector machine. This method selects the best solution by evaluating several combinations of possible values. We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter of the classifier depending on the actual accuracy.

For each feature representation, we have used SVM with single kernel predictor. In SVM with single kernel, we have used radial basis function as kernel and tune it kernel parameter, sigma, as well as soft margin parameter C of SVM. To find the parameter value C (penalty term for soft margin) and sigma, we have considered the values from 2^{-8} to 2^{8} for C and from 2^{-8} to 2^{8} for sigma as our searching space. Herein, the value

Table 4.28: Selected C and σ of 5 complete runs of the 5-fold cross-validation for single-kernel based SVM using RBF kernel

Feature Extraction Approaches	Selected C from 5 Times Run					Selected σ from 5 Times Run				
	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
AAC	2^0	2^2	2^2	2^{-1}	2^1	2^1	2^2	2^1	2^1	2^2
DC	2^1	2^1	2^1	2^1	2^1	2^4	2^4	2^4	2^4	2^4
PAAC	2^3	2^1	2^4	2^1	2^4	2^3	2^2	2^4	2^2	2^4
APAAC	2^2	2^2	2^1	2^4	2^1	2^3	2^3	2^3	2^4	2^3
PPM	2^5	2^5	2^4	2^4	2^4	2^2	2^2	2^2	2^2	2^2
AAID	2^2	2^1	2^1	2^1	2^2	2^2	2^2	2^2	2^2	2^2
PSSM	2^6	2^6	2^4	2^3	2^3	2^3	2^2	2^3	2^2	2^2

of C will be used to find the misclassification cost of C^+ and C^- defined in equation 4.1. The selected C and sigma of 5 complete runs of the 5-fold cross-validation for the gram-negative bacteria dataset for each feature extraction approach is shown in Table 4.28.

In feature fusion case, we have used SVM with MKL predictor, named KFFLoc-Gneg. In SVM with multiple kernel, we have also used radial basis function as base kernel and tune it kernel parameter, sigma, as well as soft margin parameter C of SVM. To find the parameter value C (penalty term for soft margin) and sigma, we have considered the values from 2^{-8} to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. The selected C and sigma of 5 complete runs of the 5-fold cross-validation for the gram-negative bacteria dataset for feature fusion based approach is shown in Table 4.29.

It can be mentioned here that We have used Matlab 2014b version to implement our system where the svmtrain function of Matlab by default uses DEC with the same cost defined in Eq. 4.1 to handle imbalance situation.

4.9.3.2 Prediction Performance Evaluation

The major objective of this work is to fuse different feature extracted using different extraction approach that may be capable to produce a higher performance of protein subcellular localization prediction especially for non-GO termed protein. Keeping in mind this objective, we have compared the prediction performance of seven different

Table 4.29: Selected C and σ of 5 complete runs of the 5-fold cross-validation for MKL based SVM using RBF kernel

No. of Completes Run	C	σ
1^{st}	2^{-2}	2^{3}
2^{nd}	2^{-3}	2^{3}
3^{rd}	2^{-3}	2^{3}
4^{th}	2^{-3}	2^{3}
5^{th}	2^{-2}	2^{3}

feature representations extracted from seven different approaches and feature fusion approach, named KFFLoc-Gneg.

Table 4.30: Prediction results, averaged over 5 times run of the 5-fold cross-validation, for each prediction algorithm using different feature extraction approaches and feature fusion approaches for the gram-negative bacterial dataset.

Feature Extraction Approaches	Classification Method	Actual ACC %	Locative ACC %
AAC	SVM (Single Kernel)	66.27 ($\pm$0.448)	70.92 ($\pm$2.216)
DC	SVM (Single Kernel)	65.40 ($\pm$0.749)	69.71 ($\pm$0.702)
PAAC	SVM (Single Kernel)	64.59 ($\pm$0.368)	68.89 ($\pm$2.224)
APAAC	SVM (Single Kernel)	64.75 ($\pm$0.460)	68.35 ($\pm$0.962)
PPM	SVM (Single Kernel)	61.97 ($\pm$0.347)	66.19 ($\pm$0.538)
AAID	SVM (Single Kernel)	64.96 ($\pm$0.416)	69.11 ($\pm$0.968)
PSSM	SVM (Single Kernel)	61.14 ($\pm$0.389)	66.08 ($\pm$1.486)
Feature Fusion	SVM with MKL (KFFLoc-Gneg)	73.14 ($\pm$0.299)	75.47 ($\pm$0.384)

Table 4.30 provide the actual accuracy as well as locative accuracy of the predictor based on each of the seven feature extraction approach and feature fusion approach for the gram-negative bacteria datasets. For each of feature extraction approach and

feature fusion approach, 5 complete runs of the 5-fold cross-validation have been performed and the averaged performance have been presented in Table 4.30. In addition to tabular presentation as in Table 4.30, the comparison of the results of actual accuracy and locative accuracy have also been shown graphically in Figure 4.10 and 4.11 respectively. From the Table 4.30 as well as from the figures, it is clear that feature fusion based predictor produces the best performance while considering actual accuracy and locative accuracy than all other single feature based predictor. According actual accuracy, feature fusion based predictor i.e. SVM with MKL (KFFLoc-Gneg) produces at least 8% better result than any of the single feature based predictor. On the other hand, fusion based predictor provides at least 5% better locative accuracy than any of the single feature based predictor. Moreover, feature fusion based predictor achieved smaller standard deviation than all single feature based predictor.

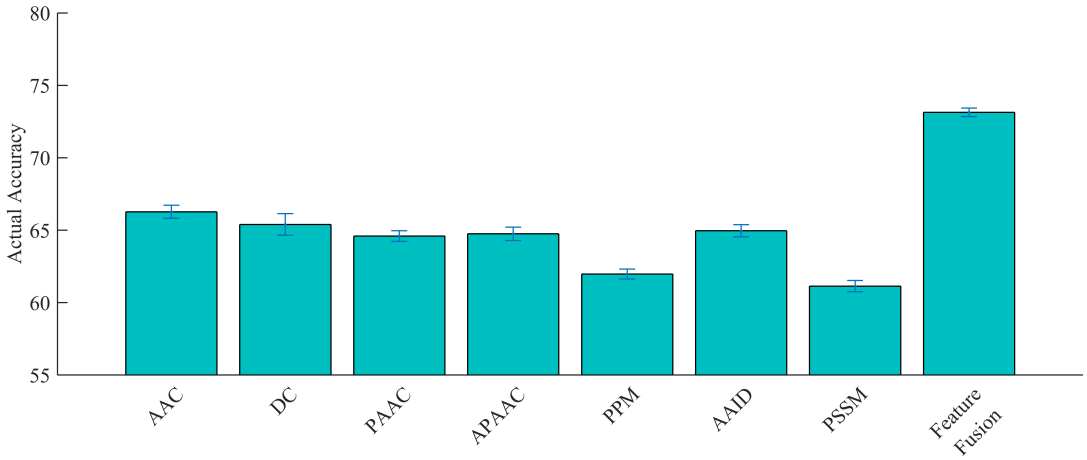

Figure 4.10: Performance comparison of actual accuracy among single feature based predictor VS feature fusion based predictor

The following point may be the reason why the proposed feature fusion based predictor can improve the success rates over single feature based predictor. KFFLoc-Gneg uses combined kernel which has been derived from all the kernels with different rates of contribution for each of the feature representation in order to build SVM based classifier. It should be noted that different feature provides complementary views about a protein. We can expect that having more information about a subject, we have better chance of analyzing it with higher accuracy. Here it should be noted that the influences or weights of all kernel has been derived and combined using MKL.

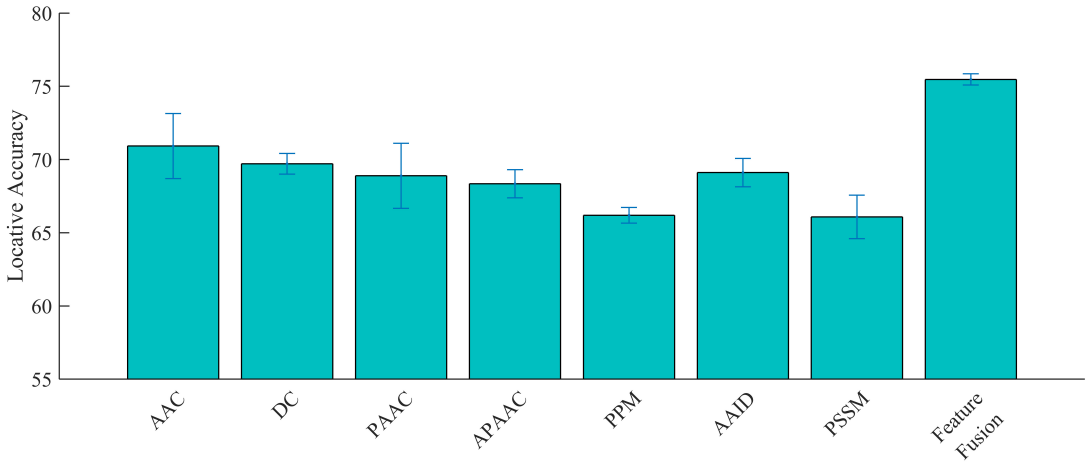

Figure 4.11: Performance comparison of locative accuracy among single feature based predictor VS feature fusion based predictor

4.10 Experiment No 8:- predHumPhos: Predicting Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

4.10.1 Motivation and Goals

Human protein phosphorylation is one of the most important post-translational modifications (PTMs) which show a crucial role in the transmission of signals that controls a diverse array of cellular functions, such as cell growth, survival differentiation, and metabolism. In this context, in order to avoid the costly and time-consuming experimental technologies [159, 123], an accurate computational method for predicting phosphorylated protein is an urgent issue which can be useful for drug development. Various types of computational classifiers have been developed for identifying the phosphorylation sites for a known phosphorylated protein [50, 52, 53], to our best knowledge, so far very few computational methods has been developed to predict whether an uncharacterized protein is being able to be phosphorylated or not [54]. However, in order to meet the current demand to produce efficient high-throughput tools for prediction, additional effort are required to enrich the prediction quality [54, 160].

Therefore, in this experiment, a novel computational tool termed predHumPhos has been developed to predict phosphorylated proteins by (1) extracting three different set of features from protein sequences, (2) defining an individual kernel on each set of features and combining them into a single kernel using multiple kernel learning (MKL), (3) constructing a predictor using support vector machine (SVM) which has been trained with the combined kernel. In addition, we have balanced the effect of skewed training dataset by Different Error Costs (DEC) method in the development of

our system. Moreover, a user-friendly web server for the predHumPhos has also been established at http://research.ru.ac.bd/predHumPhos/

4.10.2 Materials and Methods

4.10.2.1 Short Description of Dataset

iPhos-PseEvo's [54] benchmark dataset set has been used in this study. iPhos-PseEvo's dataset consist of 1770 protein sequences from human. At first, in iPhos-PseEvo's study, some protein sequences were collected from the web site at http://www.uniprot.org/ by giving various constrains such as i) experimental assertion for evidence, ii)consider only human protein sequences, iii) use keywords of 'phosphoserine', 'phosphothreonine', or 'phosphotyrosine' in the advance search option. After that some more additional constraint has been applied on that collected sequences such as a) exclude sequences with less than 50 and more than 5000 amino acid residues for the convenience of constructing Position Specific Scoring Matrix (PSSM) and b) reduce the redundancy and homology bias by removing $> 50\%$ pairwise sequence identity to any other in the same subset. Finally, iPhos-PseEvo obtained 1770 human protein sequences [54].

Among 1770 protein sequences of iPhos-PseEvo's dataset, denoted as dataset S, 1132 are phosphorylated proteins, noted as S^+, and 638 non-phosphorylated proteins, noted as S^-. Also, among the 1132 phosphorylated proteins, 845 are phosphorylated at serine (S), 386 at threonine (T), and 249 at tyrosine (Y). The proteins benchmark dataset S can be formulated

$$S = S^+ \cup S^-$$

where S^+ is used to denote the set of phosphorylated proteins, and S^- is used to denote the set of non-phosphorylated proteins.

In iPhos-PseEvo, the protein samples was generally expressed as

$$P = R_1, R_2, R_3, \cdots, R_{(L-1)}, R_L \ \triangleright \{S; T; or Y\}$$

where R_1 represents the 1^{st} residue of the protein P, R_2 the 2^{nd} residue, R_3 the 3^{rd} residue, and so forth; the symbol $\triangleright$ means that, of the L amino acid residue, at least one must be S (serine), or T (threonine), or Y (tyrosine). In other words, there is at least one $i \in 1, 2, 3, \cdots, L$ subject to $R_i \in S, T, Y$.

Thus, the benchmark dataset obtained by iPhos-PseEvo for S^+, S^- are available at online supplementary materials (http://research.ru.ac.bd/predHumPhos/) as Supporting Information S1 and Supporting Information S2 respectively . It should be mention that our published online supplementary materials are taken from iPhos-PseEvo's work [54]. A summary of this benchmark dataset is given in Table 4.31.

Table 4.31: Summary of the dataset

Subset	Number of Samples
S^+	1132
S^-	638
Total different proteins	1770

4.10.2.2 Feature Extraction

In this study, we have considered the following three approaches to extract feature from protein sequences.

1. Amino Acid Composition (AAC) [See section 3.6.5.1]
2. Dipeptide Composition (DC) [See section 3.6.5.2]
3. Sequential Evolution Information (PSSM) [See section 3.6.5.8]

4.10.2.3 Support Vector Machine and its Kernel

The SVM modeling algorithm has been used in this study. It is defined in section 2.2.1. In the study, we have used radial basis function (RBF) kernel. The definition of RBF kernel has been defined in section 2.2.1.1.

4.10.2.4 Multiple Kernel Learning

Multiple kernel learning (MKL) based SVM is as an extension of single kernel based SVM to incorporate multiple kernels in classification [41, 10, 43]. There are two well-known problem formulations of MKL namely one-stage MKL and two-stage MKL [10, 45]. In this paper, the two-stage MKL problem formulation has been used. In two-stage MKL based system, at first, an individual kernel on each set of features are defined, after that these individual kernels are combined into a single kernel using specific rules of MKL, and finally, trained a single SVM on that combined kernel. The architecture of the proposed system is shown in Figure 4.12.

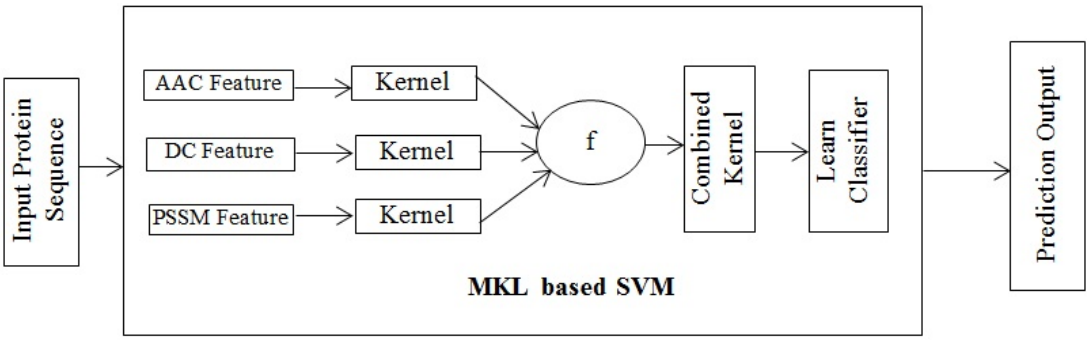

Figure 4.12: A flowchart to show how the predHumPhos predictor works

The form of linear combined kernel of multiple kernel learning (MKL) is defined below [10]

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i^m, x_j^m) \tag{4.19}$$

where η_m denotes the kernel weights, $k_m(x_i^m, x_j^m)$ take P feature representations (not necessarily different) of data instances, $x_i^m \in R^{D_m}$, and D_m is the dimensionality of the corresponding feature representation.

In our research work, in order to get the combination function parameters or kernel weights, a heuristic approach proposed by Qiu and Lane [85] has been used that uses kernel alignment based similarity measure to find the kernel weights. Note that kernel alignment based similarity measure is discussed in section 2.3.2.2 and 2.3.2.3. Qiu and Lane [85] propose the following simple heuristic for classification problems to select the kernel weights using kernel alignment:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P} K_h, yy^T} \qquad \forall m$$

where we obtain the combined kernel as a convex combination of the input kernels.

4.10.2.5 Imbalance Data Management

In this experiments, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem for this kind of prediction that is discussed in section 4.2.1.1.

4.10.2.6 Experimental Setting

In this paper, in order to save the computational time, two stage approaches has been used to find the best model for our predictor. In the first stage, we have used 5 complete runs of the 5-fold cross validation and each time we have used grid-search technique to select the values of parameters considering the highest accuracy. In this way, we have got 5 sets of parameters (values of C and sigma) for the MKL based SVM. In the second stage, we have used jackknife test and searched again using grid search

approach to select the values of parameter. But, this time the search space will be limited to the 5 sets of values of the parameter which has been found in the first stage.

4.10.2.7 Measuring Metrics

For measuring the predictive capability and reliability for this kind of classification, a set of four metrics is usually used in the literature: (i) overall accuracy or Acc, (ii) Mathew's correlation coefficient or MCC, (iii) sensitivity or Sn, and (iv) specificity or Sp [54, 107, 133, 133]. The definitions of these metrics are discussed in section 4.2.3.2.

4.10.3 Results and Discussion

4.10.3.1 Model Selection for SVM

In order to generate highly performing classifiers capable of dealing with real data an efficient model selection is required. It is noted that, we have used RBF kernel as base kernel for each of the kernel of MKL. For radial basis function (RBF) kernel, to find the parameter value C (penalty term for soft margin) and σ (sigma) for predHumPhos, we have considered the value from 2^{-8} to 2^{8} for C and from 2^{-8} to 2^{8} for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^{+} and C^{-} defined in equation 4.1

According to the experimental setting defined in section 4.10.2.6, in the first stage, we have performed 5 complete runs of the 5-fold cross validation and each time we have selected the best parameter (value of C and sigma) of the classifier depending on the value of accuracy using grid search technique. In this way, five sets of C and sigma have been selected for the MKL based SVM classifier as shown in Table 4.32. However, in the second stage, we have performed jackknife test and searched the best parameter for C and sigma among the selected five sets of C and sigma. Finally, we have found the best model with the value of $C = 2^{2}$ and $\sigma = 2^{3}$ for our predictor through jackknife test.

It can be mentioned here that we have used Matlab 2014b version to implement our system where the svmtrain function of Matlab by default uses DEC with the same cost defined in Eq. 4.1 to handle imbalance situation.

4.10.3.2 Prediction Performance Evaluation

The values of the four metrics (cf. Eq. 4.4) obtained by the current predHumPhos predictor for the prediction of phosphorylated proteins using jackknife test with the selected $C = 2^{2}$ and $\sigma = 2^{3}$. From Table 4.33, it is observed that predHumPhos exhibits accuracy rate of 73.50% for predicting phosphorylation proteins which shows that our model can predict whether a given site is positive or negative with high confidence. Likewise, predHumPhos produces Matthew's correlation coefficients (MCC) of 0.4487 for predicting phosphorylated proteins. Finally, predHumPhos also has performed well

Table 4.32: Selected C and σ of 5 complete runs of the 5-fold cross-validations for MKL based SVM using RBF kernel

No. of Completes Run	C	σ
1^{st}	2^2	2^3
2^{nd}	2^3	2^2
3^{rd}	2^0	2^2
4^{th}	2^2	2^3
5^{th}	2^0	2^2

Table 4.33: A Comparison of the proposed predictor with the existing methods on the same dataset

Predictor	ACC(%)	MCC	Sn (%)	Sp (%)
iPhos-PseEvo	71.79	0.4362	71.16	70.06
predHumPhos	73.50	0.4487	74.65	71.47

in regard to both specificity, which measures the predictor's ability to correctly predict non-phosphorylated proteins (71.47%) and sensitivity, which measures the percentage of positive sites that are predicted correctly out of all known phosphorylated proteins (74.65%).

The Table 4.33 also includes the corresponding rates achieved by iPhos-PseEvo [54], the one existing predictors for identifying the phosphorylated protein in the aforesaid benchmark dataset. It should be mentioned here that the performance of iPhos-PseEvo [54] as shown in Table 4.33 are noted from [54].

It is obvious from the Table 4.33, predHumPhos has performed remarkably better over iPhos-PseEvo while considering Acc, MCC, Sn, and Sp. It indicates that, the proposed new predictor has produced over all better accuracy, sensitivity, specificity and stability.

Why can the proposed method enhance the prediction quality so significantly? First, three features are used instead of one feature that provide a more "complete" representation of proteins as done by previous investigators in successfully improving the prediction performance for various protein attributes [50, 20, 161, 77]. Second, multiple kernel learning (MKL) has been used to make a combined kernel by fusing these three features and finally, SVM has been trained using the combined kernel which

enhances the prediction quality and increases the stability as proved by many previous studies [161, 77, 41, 10, 85, 153]. Third, the predictor used Different Error Costs (DEC) method to balance the effect of skewed training dataset and hence reduced many false prediction events caused by imbalanced and skewed training datasets as established in some recent studies [54, 105, 51].

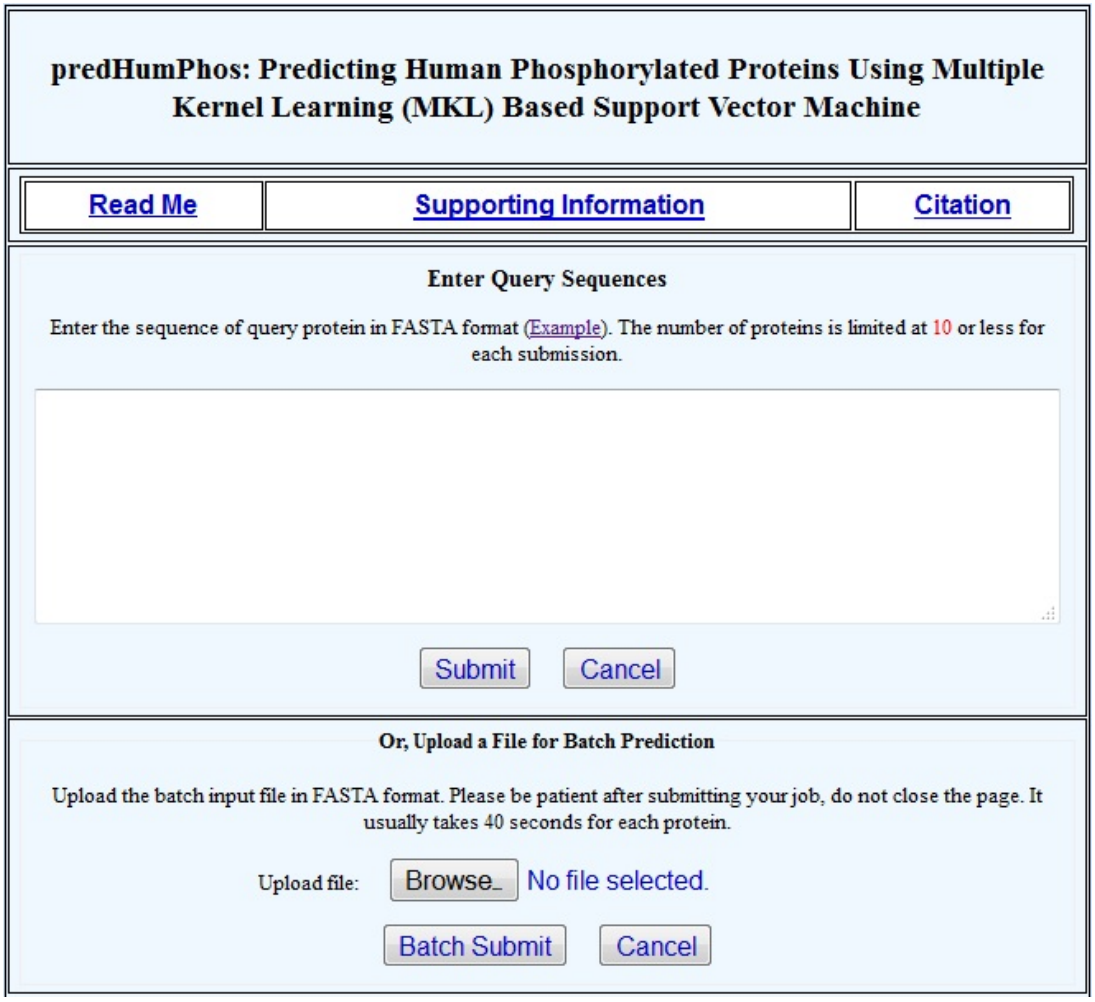

Figure 4.13: A Semi-screenshot for the home page of the webserver predHumPhos at http://research.ru.ac.bd/predHumPhos/

4.10.3.3 Protocol Guide

To attract more users especially for the experimental scientists and enhance the value of practical application, a user-friendly web-server for predHumPhos has been established at http://research.ru.ac.bd/predHumPhos/. A step-by-step guide on how to use the web server is given below

I Open the web server at http://research.ru.ac.bd/predHumPhos/ and you will find the home page of the predictor on your display as shown in Fig. 4.13. You will have to either type or copy and paste the query protein sequence into the

input text box at the center of Fig. 4.13. The input sequence should follow the FASTA format. The example of a sequence of FASTA format is available by clicking at example button located right above the input text box.

II In order to get the predicted result, click on the Submit button. For example, if you use any of the query protein sequence given under Example button as input, it will take 40s or more from the time of your submission to get desired output of each sequence.

III In order to get batch prediction, you will have to enter desired batch input file (in FASTA format of course) via Browse button located on the lower panel, as shown in Fig. 4.13.

4.11 Experiment No 9:- iMulti-HumPhos: A Multi-Label Classifier for Identifying Human Phosphorylated Proteins Using Multiple Kernel Learning Based Support Vector Machine

4.11.1 Motivation and Goals

Human protein phosphorylation is one of the most important post-translational modifications (PTMs) which show a crucial role in the transmission of signals that controls a diverse array of cellular functions, such as cell growth, survival differentiation, and metabolism. In this context, given a human protein sequence, a question may be raised whatever it can be phosphorylated or not? Clearly, this is a very important problem for cellular physiology and pathology, which in turns, helps in providing some valuable evidence for the researcher in the area of biomedical as well as for the designing and development of drugs [55]. Recently various types of computational methods have been developed for predicting the phosphorylation sites for a recognized phosphorylated protein [50, 52, 53], to our best knowledge, so far a very few computational methods [55] has been developed to identify whether an uncharacterized protein can be phosphorylated or not. Moreover, among all residues of protein molecules, three types of amino acid residues, namely serine (S), threonine (T), and tyrosine (Y), have been found susceptible to phosphorylation which leads to the requirement of multi-label phosphorylated protein identification. However, in order to meet the current demand to produce efficient high-throughput tools for both single- and multi-label prediction, additional effort are required to enrich the prediction quality [55].

Therefore, in this experiment, a novel computational tool termed iMulti-HumPhos has been developed to predict multi-label phosphorylated proteins by (1) extracting three different set of features from protein sequences, (2) defining an individual kernel on each set of features and combining them into a single kernel using multiple kernel learning (MKL), (3) constructing a multi-label predictor using a combination

Table 4.34: Summary of the dataset (1)

Subset	Number of Samples
S_1	845
S_2	386
S_3	249
S_4	375
Total locative proteins	1855

of support vector machine (SVM) where each SVM has been trained with the combined kernel. In addition, we have balanced the effect of skewed training dataset by Different Error Costs (DEC) method in the development of our system. Moreover, a user-friendly web server for the iMulti-HumPhos has also been established at http://research.ru.ac.bd/iMulti-HumPhos/

4.11.2 Materials and Methods

4.11.2.1 Short Description of Dataset

Multi-iPPseEvo's [55] benchmark dataset set has been used in this study. Multi-iPPseEvo's dataset consist of 1507 protein sequences from human. At first, in Multi-iPPseEvo's study, 2076 protein sequences were collected from the web site at http:// www.uniprot.org/ by giving various constrains such as i) experimental assertion for evidence, ii)consider only human protein sequences, iii) use keywords of 'phosphoserine', 'phosphothreonine', or 'phosphotyrosine' in the advance search option. After that some more additional constraint has been applied on that collected sequences (2076 sequences) such as a) exclude sequences with less than 50 and more than 5000 amino acid residues for the convenience of constructing Position Specific Scoring Matrix (PSSM) and b) reduce the redundancy and homology bias by removing $> 50\%$ pairwise sequence identity to any other in the same subset. Finally, Multi-iPPseEvo obtained 1507 human protein sequences [55].

Among 1507 protein sequences of Multi-iPPseEvo's dataset, denoted as dataset S, 845 occur in phosphoserine phosphorylation subset, noted as S1, 386 in phosphothreonine phosphorylation subset, noted as S2, 249 in phosphotyrosine phosphorylation subset, noted as S3, and 375 in non-phosphorylation subset, noted as S4. The proteins benchmark dataset S covers 4 different subsets, and can be formulated

Table 4.35: Summary of the dataset (2)

Number of phosphorylation	Number of Samples
One	820
Two	276
Three	36
Zero	375
Total actual proteins	1507

$$S = S_1 \cup S_2 \cup S_3 \cup S_4$$

Here, 1507 different proteins of the dataset S are called actual proteins and the subsets S1, S2 and S3 of S are not mutually exclusive. However, to have a look from the multi-label view of the dataset S, we have found that among 1507 proteins of S, 820 proteins have any one type of phosphorylation among the three types of possible phosphorylations (phosphoserine, phosphothreonine and phosphotyrosine), 276 proteins have any two types of phosphorylation, 36 proteins have all of the three types of phosphorylation and 375 have no phosphorylation at all. Hence, there are 1855 (820+276*2+36*3+375) locative proteins in total in this dataset S. The concept of actual proteins and locative proteins has been described in detail in literature [14, 63]. A summary of this benchmark dataset is given in Table 4.34 and Table 4.35.

In Multi-iPPseEvo, the protein samples was generally expressed as

$$P = R_1, R_2, R_3, \cdots, R_{(L-1)}, R_L \triangleright \{S; T; or Y\}$$

where R_1 represents the 1^{st} residue of the protein P, R_2 the 2^{nd} residue, R_3 the 3^{rd} residue, and so forth; the symbol $\triangleright$ means that, of the L amino acid residue, at least one must be S (serine), or T (threonine), or Y (tyrosine). In other words, there is at least one $i \in 1, 2, 3, \cdots, L$ subject to $R_i \in S, T, Y$.

Thus, the benchmark dataset obtained by Multi-iPPseEvo for S_1, S_2, S_3, and S_4 are available at online supplementary materials (http://research.ru.ac.bd/iMulti-HumPhos/) as Supporting Information. It should be mention that our published online supplementary materials are taken from Multi-iPPseEvo's work [55].

4.11.2.2 Feature Extractions

In this study, we have considered the following three approaches to extract feature from protein sequences.

1. Amino Acid Composition (AAC) [See section 3.6.5.1]
2. Dipeptide Composition (DC) [See section 3.6.5.2]
3. Sequential Evolution Information (PSSM) [See section 3.6.5.8]

4.11.2.3 Support Vector Machine and its Kernel

In this study, a little bit variation in the decision function of multi-label SVM modeling algorithm, defined in section 2.2.1.3, has been used. For better understanding, some part of section 2.2.1.3 are again stated here to accommodate the variation clearly.

In this study, N independent binary MKL based SVMs predictors are trained, one predictor for each class of phosphorylation. In the final step, the phosphorylation(s) of the i-th query protein are predicted as

$$M^*(x_i) = \cup_{j=1}^{N} \{j : f_j(x_i) > 0\} \tag{4.20}$$

Here, $M^*(x_i)$ is a predicted set that may have one, or more elements, even it can be empty too, which enables us to make multi-label prediction. However, if Eq. 4.20 provides an empty class label, i.e., $M^*(x_i) = \emptyset$, in that case, the number of phosphorylation will be zero and will be predicted as non-phosphorylated protein.

4.11.2.4 Multiple Kernel Learning

Multiple kernel learning (MKL) based SVM is as an extension of single kernel based SVM to incorporate multiple kernels in classification [41, 10, 43]. There are two well-known problem formulations of MKL namely one-stage MKL and two-stage MKL [10, 45]. In this paper, the two-stage MKL problem formulation has been used. In two-stage MKL based system, at first, an individual kernel on each set of features are defined, after that these individual kernels are combined into a single kernel using specific rules of MKL, and finally, trained a single SVM on that combined kernel.

The form of linear combined kernel of multiple kernel learning (MKL) is defined below [10]

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i^m, x_j^m) \tag{4.21}$$

where η_m denotes the kernel weights, $k_m(x_i^m, x_j^m)$ take P feature representations (not necessarily different) of data instances, $x_i^m \in R^{D_m}$, and D_m is the dimensionality of the corresponding feature representation.

In our research work, in order to get the combination function parameters or kernel weights, a heuristic approach proposed by Qiu and Lane [85] has been used that uses kernel alignment based similarity measure to find the kernel weights. Note that kernel alignment based similarity measure is discussed in section 2.3.2.2 and 2.3.2.3. Qiu and Lane [85] propose the following simple heuristic for classification problems to select the kernel weights using kernel alignment:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P} K_h, yy^T} \qquad \forall m$$

where we obtain the combined kernel as a convex combination of the input kernels.

4.11.2.5 Imbalance Data Management

In this experiments, we have used a Different Error Costs (DEC) method to handle imbalance dataset problem for this kind of prediction that is discussed in section 4.2.1.1.

4.11.2.6 Experimental Setting

In this work, we have used K-fold cross validation (subsampling) method to save the computational time. As the information about the exact 5-way splits of dataset used in previous studies is not published [55], therefore, in order to validate the stability and the statistical significance of our results, we have repeated the 5-fold cross validation for 5 times (i.e. 25 runs in total). It can be mentioned here that in each 5-fold cross validation the given training samples are randomly partitioned into 5 mutually exclusive sets of approximately equal size and approximately equal class distribution. Finally, we have reported the average results of all metrics in this study.

4.11.2.7 Measuring Metrics

In this study, we have considered {phosphoserine, phosphothreonine, phosphotyrosine, $\emptyset$} as class label set for a protein. Here $\emptyset$ is used to denote non-phosphorylated protein. Since we are dealing with a multi-label system [145], so the metrics for a multi-label system will be used in this work instead of the conventional metrics defined for single-label systems [132, 133, 134].

For measuring the predictive capability and reliability for this kind of classification, a set metrics are usually used in the literature which are define below [145, 162, 163, 104]:

$$
\begin{aligned}
Precision &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Z_k\|} \right) \\
Recall &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Y_k\|} \right) \\
Accuracy &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cap Z_k\|}{\|Y_k \cup Z_k\|} \right) \\
Subset - Accuracy &= 1/N \sum_{k=1}^{N} \triangle (Y_k, Z_k) \\
Humming - loss &= 1/N \sum_{k=1}^{N} \left(\frac{\|Y_k \cup Z_k\| - \|Y_k \cap Z_k\|}{M} \right)
\end{aligned}
\tag{4.22}
$$

where N is the total number of the samples concerned, M the total number of labels in the system, $\cup$ and $\cap$ the symbols are for the 'union' and 'intersection' in the set theory, $\|-\|$ means the operator acting on the set therein to count the number of its elements, Y_k denotes the subset that contains all the labels experiment-observed for the k-th sample, Z_k represents the subset that contains all the labels predicted for the k-th sample, and

$$
\triangle (Y_k, Z_k) = \begin{cases} 1 & \text{if all labels in } Z_k \text{ are identical with those in } Y_k \\ 0 & \text{otherwise} \end{cases}
$$

In Chou's work [145], these metrics are defined with different names in order to make more intuitive and easier to be understood for most biologists. For example, Aiming, Coverage, Accuracy, Absolute-True and Absolute-False in [145] are corresponding to Precision, Recall, Accuracy, Subset-Accuracy and Hamming-loss, respectively [55].

All of these metrics defined in this section have been successfully applied to study several multi-label systems, such as those in which a protein may stay in two or more different subcellular locations [114], or a membrane protein may have two or more different types [146], or an antimicrobial peptide may have two or more different types [147].

4.11.3 Results and Discussion

4.11.3.1 Model Selection for SVM

In order to generate highly performing classifiers capable of dealing with real data an efficient model selection is required. In this paper, grid-search technique has been used to find the best model for MKL based SVM classifier. In our experiments, this method selects the values of parameters considering the highest accuracy and then time if more than one position in search space has the same performance.

In this study, three MKL based SVM classifiers have been used, one for each class label phosphoserine, phosphothreonine, or phosphotyrosine. It should be noted here

Table 4.36: Number of positive and negative samples for the three sub-predictors

Attribute	Phosphorylation Type and Number of Samples		
	phosphoserine	phosphothreonine	phosphotyrosine
Positive	845	386	249
Negative	(1507-845)=662	(1507-386)=1121	(1507-249)=1258

Table 4.37: Selected C and σ of 5 complete runs of the 5-fold cross-validations for MKL based SVM using RBF kernel

No. of Completes Run	Type of Phosphorylation					
	Phosphoserine		Phosphothreonine		Phosphotyrosine	
	C	σ	C	σ	C	σ
1^{st}	2^1	2^3	2^4	2^3	2^4	2^3
2^{nd}	2^1	2^3	2^2	2^3	2^3	2^2
3^{rd}	2^0	2^3	2^3	2^3	2^3	2^3
4^{th}	2^1	2^3	2^4	2^3	2^3	2^3
5^{th}	2^1	2^3	2^3	2^3	2^2	2^3

that the negative results of all (three) classifiers will be treated as class label $\emptyset$ (non-phosphorylated protein). The model selection of each MKL based SVM classifier has been done separately as binary classifier using the corresponding benchmark dataset given in Table 4.36. Noted that Table 4.36 has been derived from Table 4.34 and Table 4.35, where the number of positive proteins comes from the corresponding class (phosphoserine, phosphothreonine, or phosphotyrosine) and the number of negative proteins is calculated by subtracting the number of positive proteins from the number of total actual proteins.

In this work, we have used RBF kernel as base kernel for each of the kernel of MKL. For radial basis function (RBF) kernel based SVM, to find the parameter value C (penalty term for soft margin) and σ (sigma), we have considered the value from 2^{-8} to 2^8 for C and from 2^{-8} to 2^8 for sigma as our searching space. Herein, the value of C will be used to find the misclassification cost of C^+ and C^- defined in equation 4.1. Since the information about the exact 5-way splits of dataset used in previous

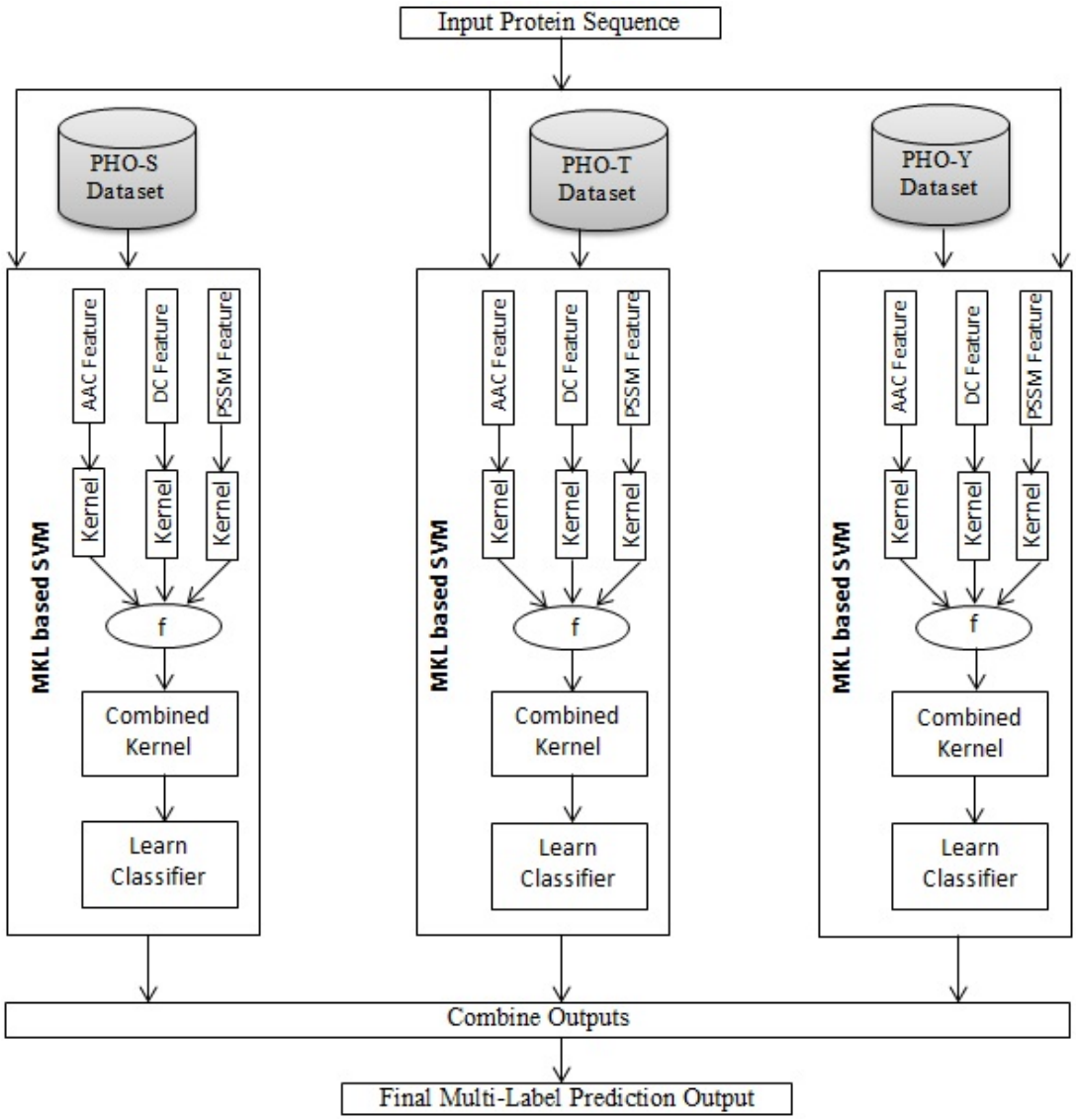

Figure 4.14: A flowchart to show how the iMulti-HumPhos predictor works

studies is not published [55], We have performed 5 complete runs of the 5-fold cross-validation and each time we have selected the best parameter (value of C and sigma) of the classifier depending on the value of accuracy (traditional accuracy). Finally, five sets of C and sigma have been selected from 5 complete runs of the 5-fold cross-validation for each MKL based SVM classifier which is dedicated to a specific types of training dataset (phosphoserine, phosphothreonine, or phosphotyrosine) as shown in Table 4.37.

After getting the three trained binary MKL based SVM classifier with appropriate values of C and sigma (shown in Table 4.37), a multi-label predictor, named iMulti-HumPhos, has been developed by combing output from these three MKL based SVM classifiers, as shown in Figure 4.14. As we have repeated the 5-fold cross validation for 5 times for our iMulti-HumPhos, we have got five sets of values for all metrics defined in section 4.11.2.7. Finally, we have averaged our results in order to ensure unbiased

Table 4.38: Selected C and σ to train the system for the web server

Type of Phosphorylation	C	σ
phosphoserine	2^1	2^3
phosphothreonine	2^3	2^3
phosphotyrosine	2^3	2^3

model selection and cross validations.

However, in order to train the system for the web server, we have used that value of C and sigma which appears most of the times as best model in 5 complete runs of the 5-fold cross validation in each dataset given in Table 4.36. Note that, a random selection of the value of C and sigma has also been performed from 5 sets of C and sigma of each dataset where 'most of the times' criteria fail to select C and sigma. In this way, the selected C and sigma for each type of dataset is given in Table 4.38.

It can be mentioned here that we have used Matlab 2014b version to implement our system where the svmtrain function of Matlab by default uses DEC with the same cost defined in Eq. 4.1 to handle imbalance situation.

4.11.3.2 Comparison with the Existing Methods

The values of the five metrics (cf. Eq. 4.22) obtained by the current iMulti-HumPhos predictor for the prediction of multi-label phosphorylated proteins are given in the Table 4.39. These values are the average result of 5 complete runs of the 5-fold cross-validation on the benchmark dataset given in Supporting Information. Moreover, standard deviations of each metrics of 5 complete runs of the 5-fold cross validation are shown in parentheses.

The Table 4.39 also includes the corresponding rates achieved by Multi-iPPseEvo [55], the one existing predictors for identifying the mult-label phosphorylated protein in the aforesaid benchmark dataset. It should be mentioned here that the performance of Multi-iPPseEvo [55] as shown in Table 4.39 are noted from [55].

In Eq. 4.22, the first four metrics are completely opposite to the last one. For the former, the higher the rate is, the better the multi-label predictor's performance will be; for the latter, the lower the rate is, the better its performance will be [104]. Since, the rate of 'Hamming- Loss' [145] for our predictor is 0.2225, so, the average ratio of the completely wrong hits over the total prediction events for iMulti-HumPhos is lower than Multi-iPPseEvo.

Among the five metrics in Eq. 4.22, the most important and strict one is the 'Subset-Accuracy'. Intuitively, subset- accuracy can be regarded as a multi-label counterpart of the traditional accuracy metric. The iMulti-HumPhos achieves 0.4835 for Subset-

Table 4.39: A Comparison of the proposed predictor with the existing method on the same dataset

Predictor	Precision	Recall	Accuracy	Subset-Accuracy	Hamming Loss
Multi-iPPseEvo	0.5998	0.7335	0.5761	0.4056	0.2520
iMulti-HumPhos	0.6464 ($\pm$0.0045)	0.6286 ($\pm$0.0067)	0.5855 ($\pm$0.0044)	0.4835 ($\pm$0.0068)	0.2225 ($\pm$0.0025)

Accuracy which is higher than Multi-iPPseEvo.

Multi-label accuracy metric measures the average ratio of the correctly predicted labels over the total labels including correctly and incorrectly predicted ones whose higher value is also an indication of good predictor. The iMulti-HumPhos achieves 0.5855 for multi-label accuracy which is also higher than Multi-iPPseEvo.

Although the achieved recall by Multi-iPPseEvo is higher than that by our predictor, the gap between its precision and recall is very large. This implies that the results achieved by Multi-iPPseEvo contain many false positive events due to over prediction and hence its higher achieved recall is problematic.

Therefore, it is obvious from the Table 4.39, iMulti-HumPhos has performed remarkably better over Multi-iPPseEvo [55] in almost all types of metrics measurement. Therefore, it is projected that iMulti-HumPhos may become a useful and higher throughput tool in multi-label phosphorylation protein predictions.

Why can the proposed method enhance the prediction quality so significantly? First, three features are used instead of one feature that provide a more 'complete' representation of proteins as done by previous investigators in successfully improving the prediction performance for various protein attributes [50, 20, 161, 77]. Second, multiple kernel learning (MKL) has been used to make a combined kernel by fusing these three features and finally, SVM has been trained using the combined kernel which enhances the prediction quality and increases the stability as proved by many previous studies [161, 77, 41, 10, 85, 153]. Third, the predictor used Different Error Costs (DEC) method to balance the effect of skewed training dataset and hence reduced many false prediction events caused by imbalanced and skewed training datasets as established in some recent studies [54, 105, 51].

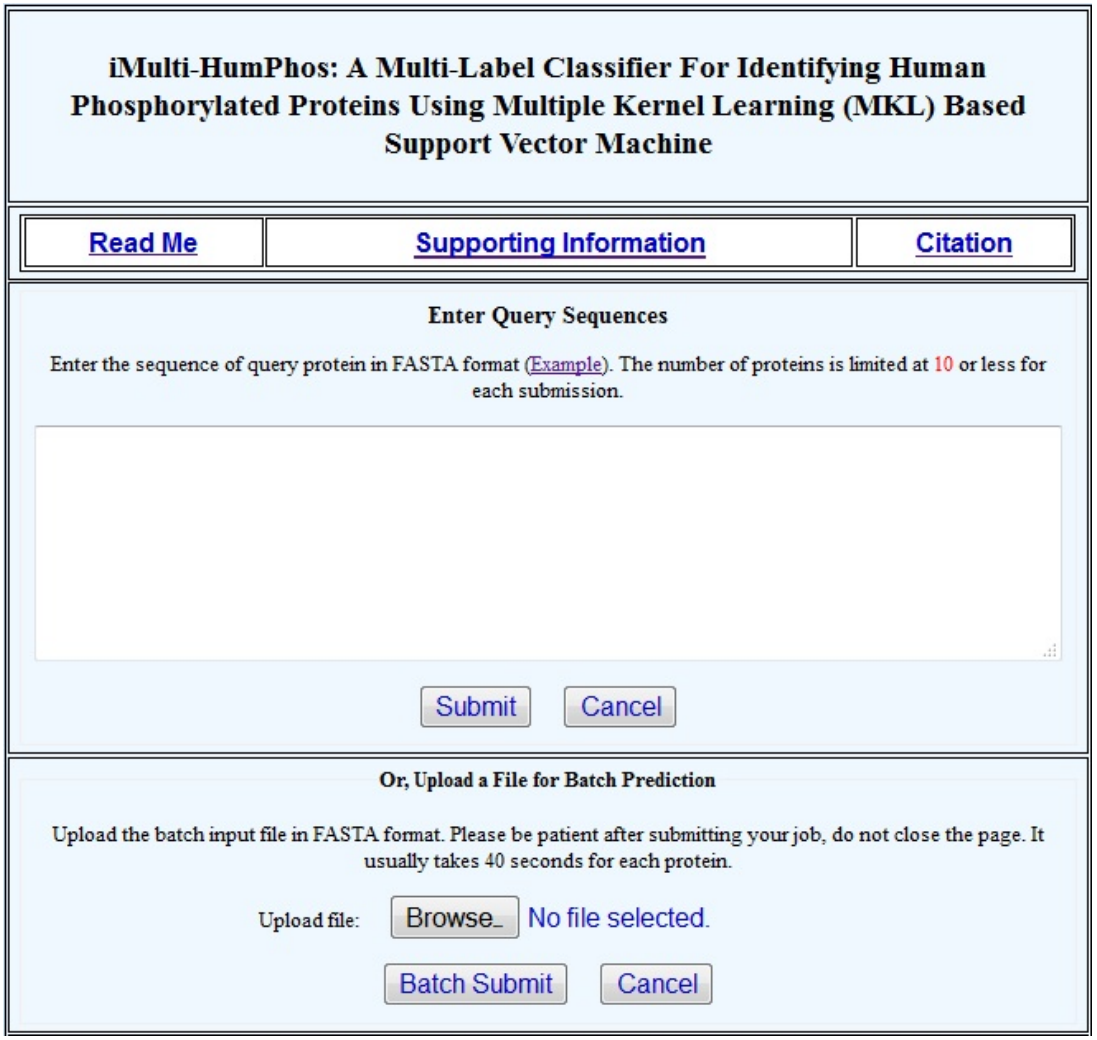

Figure 4.15: A semi-screenshot for the home page of the webserver iMulti-HumPhos at http://research.ru.ac.bd/iMulti-HumPhos/

4.11.3.3 Protocol Guide

To attract more users especially for the experimental scientists and enhance the value of practical application, a user-friendly web-server for iMulti-HumPhos has been established at http://research.ru.ac.bd/iMulti-HumPhos/. A brief step-by-step guide on how to use the web server is given below

I Open the web server at http://research.ru.ac.bd/iMulti-HumPhos/ and you will find the home page of the predictor on your display as shown in Fig. 24.15. You will have to either type or copy and paste the query protein sequence into the input text box at the center of Fig. 4.15. The input sequence should follow the FASTA format. The example of a sequence of FASTA format is available by clicking at example button located right above the input text box.

II In order to get the predicted result, click on the Submit button. For example, if you use any of the query protein sequence given under Example button as input,

it will take 40s or more from the time of your submission to get desired output of each sequence.

III In order to get batch prediction, you will have to enter desired batch input file (in FASTA format of course) via Browse button located on the lower panel, as shown in Fig. 4.15.

Chapter 5

Conclusion

5.1 Introduction

Computational Intelligence (CI) has increasingly gained attention in bioinformatics research and computational biology. With the availability of different types of CI algorithms, it has become common for researchers to apply the off-shelf systems to classify and mine their databases. At present, with various intelligent methods available in the literature, therefore scientists are facing difficulties in choosing the best method that could be applied to a specific data set. Support vector machines (SVMs) has become a popular learning paradigm in bioinformatics in the last decades. However, their performance strongly depends on the choice of kernel and kernel parameters. Therefore, kernel learning becomes a crucial problem for all kernel-based methods, in which the central question is how to choose an appropriate kernel and kernel parameters. Multiple kernel learning provides a potential solution in kernel learning problem.

On the other hand, the recent availability of multiple types of genome-wide data (multiple sources of data) provides biologists with complementary views of a single subject and highlights the need for algorithms capable of unifying these views. Each of these distinct data types provides one view of the molecular machinery of the cell. Moreover, not all of the feature descriptors will have the same discriminative power for all classes. Different data sources are likely to contain different and thus partly independent information about the task at hand. Combining those complementary pieces of information can be expected to enhance the total information about the problem at hand which in turns, enhances the performance of a system. In the near future, research in bioinformatics will focus more and more heavily on methods of data fusion.

This dissertation demonstrates the uses of multiple kernel learning(MKL) for addressing choice of kernel problem for SVM and for handling multiple data sources. This thesis highlights two problems of binoinformatics and uses MKL in order to improve their accuracy than other top existing systems. One problem is protein subcellular localization prediction and another one is post-translational modifications prediction. We have done nine experiments to solve different problems in these two fields. These nine experiments show the contribution of this thesis from four different aspects or

categories.

5.1.1 Category 1: Show the Power or Capability of Single Kernel Based SVM

In this category, four experiments have been done to show the power or capability of SVM with single kernel. In these experiments, we have got better result than other top exiting systems but the time consuming approach has been used to select the kernel (parameter tuning of RBF kernel) here. The concluding remarks of these experiments are shown below:

Experiment No 1:- predMultiLoc-Gneg: Predicting Subcellular Localization of Gram-Negative Bacterial Proteins Using Feature Selection in Gene Ontology Space and Support Vector Machine with Resolving the Data Imbalanced Issue

Concluding Remarks

- Solved imbalance dataset issue using DEC.
- Solved multi-label issue through binary relevance (BR) method.
- Provided better performance than existing top systems.
- Established a web server for public use at http://research.ru.ac.bd/predCar-Site/

Experiment No 2:- predCar-Site: Carbonylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

Concluding Remarks

- Solved imbalance dataset issue using DEC.
- Provided better performance then existing top systems.
- Established a web server for public use at http://research.ru.ac.bd/predCar-Site/

Experiment No 3:- predSucc-Site: Lysine Succinylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

Concluding Remarks

- Solved Imbalance dataset issue using DEC.
- Provided better performance than existing top systems.
- Established a web server for public use at http://research.ru.ac.bd/predSucc-Site/

Experiment No 4:- mLysPTMpred: Multiple Lysine PTM Site Prediction Using Combination of SVM Classifier with Resolving Data Imbalanced Issue

Concluding Remarks

- Solved imbalance dataset issue using DEC.
- Solved multi-label issue using combination of SVM.
- Provided better performance than existing top systems.
- Established a web server for public use at http://research.ru.ac.bd/mLysPTMpred/

5.1.2 Category 2: Show Choice of Kernel Effects for Single Kernel Based SVM

In this aspect, We have done one experiment to show how the choice of kernel problem affects the performance of a system. We have done this experiment in the field of protein subcellular localization prediction. The concluding findings of this experiment are given below:

Experiment No 5:- Protein Subcellular Localization Prediction using Support Vector Machine with the Choice of Proper Kernel

Concluding Remarks

- Evaluated the performance of different kernels for SVM in protein subcellular localization prediction.
- Results indicate that the performance of the SVM classification depends mainly on the types of kernels and their parameters.
- Choice of laplace kernel performed better than other kernels as well as other existing top systems.

5.1.3 Category 3: Use MKL as a Solution for Choice of Kernel Problem

One experiment has been done to show MKL as a tool to solve the choice of kernel problem. In this experiment, RBF kernel has been used in SVM, but the time consuming parameter tuning part (choice of kernel among the set of RBF kernels) has been avoided by using MKL. It should be noted here that different values of sigma of radial basis function (RBF) create different kernels in the domain of RBF kernel. In this work, the range of the parameter space of sigma for RBF kernel has been discretized into r values, made a combined kernel through MKL and finally learned SVM using that combined kernel. The concluding remarks of this experiment are given below:

Experiment No 6:- Protein Subcellular Localization Prediction Using Multiple Kernel Learning Based Support Vector Machine

Concluding Remarks

- Multiple kernel learning (MKL) has been used to overcome the problem of finding actual sigma value for the RBF kernel (choice of kernel among the set of RBF kernels).

- Provided better performance than other top existing systems.
- Takes less execution time than single-kernel based SVM as well as other top system.

5.1.4 Category 4: Use MKL to Fuse Multiple Data Sources

Both problems namely protein subcellular localization prediction prediction and post translational modifications (PTMs) prediction can be solved using different sources of information. In this thesis, we have shown that combination of these features from various sources can produce better result than those systems using single information source. MKL has been used to combine different features. Three experiments have been done in this aspects. The concluding message of these experiments are given below:

Experiment No 7:- Protein Subcellular Localization Prediction Using Kernel Based Feature Fusion

Contributions

- MKL has been used to integrate multiple data sources in predicting protein subcellular localization.
- Provided better performance than existing top systems.
- Developed the system considering multi-label issue.

Experiment No 8:- predHumPhos: Predicting Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

Concluding Remarks

- MKL has been used to integrate multiple data source in predicting human phosphorylated proteins. Here, single label issue (binary classification) has been considered, i.e., the prediction will be whether a protein can be phosphorylated or not.
- Provided better results than existing system.
- Established a web server for public use at http://research.ru.ac.bd/predHumPhos/

Experiment No 9:- iMulti-HumPhos: A Multi-Label Classifier For Identifying Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

Concluding Remarks

- MKL has been used to integrate multiple data source in predicting human phosphorylated proteins.

- Developed the system considering multi-label issue.
- Performed better than existing system.
- Established a web server for public use at http://research.ru.ac.bd/iMulti-HumPhos/

5.2 Future Work

We believe that all of the fingings of this thesis will be give some guidelines to the computational intelligence and bioinformatics research communities to make the applications fo multiple kernel learning (MKL)to various problems of bioinformatics research. The work described in this thesis serves as a starting point on which much more work can be extended upon. Possible extension of the work includes:

1. In this thesis, we have applied MKL in the filed of protein subcellular localization prediction and protein poststranslational modifications prediction. In future, MKL can be used to solve other tasks of bioinformatics such as protein function or structure prediction, protein-protein interaction prediction, drug-target interaction prediction, etc. in order to get better result.
2. In these thesis, we have used MKL in SVM to do supervised classification or prediction. In future, we can also use MKL in upsupervised case such as clustering problem.
3. There are some other ways to combine multiple data sources, in future we will make comparison among these integration methods.
4. In this work, we have used similarity based approach to find the kernel weights and also used two stage approach to build the predictor. In future, we can use structural risk based approach to develop predictor using MKL based SVM where one stage technique will be used.
5. In future, we can make comparison among different techniques of similarity based approaches of MKL which are usually used to find out kernel weights.

Publications

Publications in Core Contribution Area

- Published Article

    1. M. A. M. Hasan, S. Ahmad, M. K. I. Molla, "Protein subcellular localization prediction using multiple kernel learning based support vector machine," Molecular BioSystems, vol. 13, no. 4, pp. 785-795, 2017.

       Impact Factor: 2.829

    2. M. A. M. Hasan, J. Li, S. Ahmad, M. K. I. Molla, "predCar-site: Carbonylation sites prediction in proteins using support vector machine with resolving data imbalanced issue," Analytical Biochemistry, vol. 525, pp. 107-113, 2017.

       Impact Factor: 2.243

    3. Md Al Mehedi Hasan, Shamim Ahmad, and Md Khademul Islam Molla. "Protein Subcellular Localization Prediction using Support Vector Machine with the Choice of Proper Kernel", Issue no 2/2017, BioTechnologia, Journal of Biotechnology, Computational Biology and Bionanotechnology. (Accepted)

- Submitted Article (Under Review)

    1. **Paper Tile:** predSucc-Site: Lysine Succinylation Sites Prediction in Proteins Using Support Vector Machine with Resolving Data Imbalanced Issue

       Authors name: Md Al Mehedi Hasan, Shamim Ahmad, and Md Khademul Islam Molla.

       Journal Name: Journal of Bioinformatics and Computational Biology

    2. **Paper Tile:** mLysPTMpred: Multiple Lysine PTM Site Prediction Using Combination of SVM Classifier with Resolving Data Imbalanced Issue

       Authors name: Md Al Mehedi Hasan, Shamim Ahmad, and Md Khademul Islam Molla.

       Journal Name: IEEE/ACM Transactions on Computational Biology and Bioinformatics

3. **Paper Tile:** predHumPhos: Predicting Human Phosphorylated Proteins Using Multiple Kernel Learning (MKL) Based Support Vector Machine

   Authors name: Md Al Mehedi Hasan, Shamim Ahmad, and Md Khademul Islam Molla.

   Journal Name: Journal of Integrative Bioinformatics

4. **Paper Tile:** iMulti-HumPhos: A Multi-Label Classifier for Identifying Human Phosphorylated Proteins Using Multiple Kernel Learning Based Support Vector Machine

   Authors name: Md Al Mehedi Hasan, Shamim Ahmad, and Md Khademul Islam Molla.

   Journal Name: Molecular BioSystems

Publications in Related Contribution Area

- Published Article

1. M. A. M. Hasan, M. Nasser, B. Pal, S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (IDS)," Journal of Intelligent Learning Systems and Applications, vol. 6, no. 1, pp. 45-52, 2014.
2. M. A. M. Hasan, S. Xu, M. M. J. Kabir, S. Ahmad, "Performance Evaluation of Different Kernels for Support Vector Machine Used in Intrusion Detection System," International Journal of Computer Networks & Communications (IJCNC) vol. 8, no. 6, 2016.

Molecular BioSystems

PAPER

View Article Online
View Journal

Cite this: DOI: 10.1039/c6mb00860g

Protein subcellular localization prediction using multiple kernel learning based support vector machine†

Md. Al Mehedi Hasan,* Shamim Ahmad and Md. Khademul Islam Molla

Predicting the subcellular locations of proteins can provide useful hints that reveal their functions, increase our understanding of the mechanisms of some diseases, and finally aid in the development of novel drugs. As the number of newly discovered proteins has been growing exponentially, which in turns, makes the subcellular localization prediction by purely laboratory tests prohibitively laborious and expensive. In this context, to tackle the challenges, computational methods are being developed as an alternative choice to aid biologists in selecting target proteins and designing related experiments. However, the success of protein subcellular localization prediction is still a complicated and challenging issue, particularly, when query proteins have multi-label characteristics, *i.e.*, if they exist simultaneously in more than one subcellular location or if they move between two or more different subcellular locations. To date, to address this problem, several types of subcellular localization prediction methods with different levels of accuracy have been proposed. The support vector machine (SVM) has been employed to provide potential solutions to the protein subcellular localization prediction problem. However, the practicability of an SVM is affected by the challenges of selecting an appropriate kernel and selecting the parameters of the selected kernel. To address this difficulty, in this study, we aimed to develop an efficient multi-label protein subcellular localization prediction system, named as MKLoc, by introducing multiple kernel learning (MKL) based SVM. We evaluated MKLoc using a combined dataset containing 5447 single-localized proteins (originally published as part of the Höglund dataset) and 3056 multi-localized proteins (originally published as part of the DBMLoc set). Note that this dataset was used by Briesemeister *et al.* in their extensive comparison of multi-localization prediction systems. Finally, our experimental results indicate that MKLoc not only achieves higher accuracy than a single kernel based SVM system but also shows significantly better results than those obtained from other top systems (MDLoc, BNCs, YLoc+). Moreover, MKLoc requires less computation time to tune and train the system than that required for BNCs and single kernel based SVM.

Received 20th December 2016,
Accepted 3rd February 2017

DOI: 10.1039/c6mb00860g

rsc.li/molecular-biosystems

1. Introduction

A biological cell is made up of many different compartments or organelles and these compartments are close to each other and they have different functions. The proteins in the cells are responsible for most of the functions required for a cell's survival. A typical cell contains approximately one billion protein molecules that reside in many different compartments or organelles, usually termed as subcellular locations.[1] It is very interesting that proteins can perform their appropriate functions when they are located in the correct subcellular locations. Knowledge of the subcellular localization of proteins is important because it (a) provides useful insights into their functions, (b) indicates how and in which type of cellular environment they interact with each other and with other molecules, (c) helps in understanding the intricate pathways that regulate biological processes at the cellular level, and (d) helps to identify and prioritize drug targets during the process of drug development.[2,3]

Although various experimental approaches have been developed to determine the subcellular locations of proteins, most of these approaches are costly and time-consuming.[4] Moreover, since the number of newly discovered proteins has been exponentially increasing, the subcellular localization prediction purely by laboratory tests is becoming prohibitively expensive.[5] In this context, computational methods are being developed to help biologists in selecting target proteins and designing related experiments. Moreover, these computational methods are fast and can

Department of Computer Science & Engineering, University of Rajshahi, Rajshahi, Bangladesh. E-mail: mehedi_ru@yahoo.com, shamim_cst@ru.ac.bd, khademul.cse@ru.ac.bd

† Electronic supplementary information (ESI) available. See DOI: 10.1039/c6mb00860g

Analytical Biochemistry 525 (2017) 107–113

Contents lists available at ScienceDirect

Analytical Biochemistry

journal homepage: www.elsevier.com/locate/yabio

predCar-site: Carbonylation sites prediction in proteins using support vector machine with resolving data imbalanced issue

Md. Al Mehedi Hasan [a, *], Jinyan Li [b], Shamim Ahmad [a], Md. Khademul Islam Molla [a]

[a] Department of Computer Science & Engineering, University of Rajshahi, Bangladesh
[b] Advanced Analytics Institute and Centre for Health Technologies, University of Technology Sydney, Australia

ARTICLE INFO

Article history:
Received 9 December 2016
Received in revised form
26 February 2017
Accepted 7 March 2017
Available online 9 March 2017

Keywords:
Carbonylation sites prediction
Sequence-coupling model
General PseAAC
Data imbalance issue
Different error costs
Support vector machine
predCar-site web-server

ABSTRACT

The carbonylation is found as an irreversible post-translational modification and considered a biomarker of oxidative stress. It plays major role not only in orchestrating various biological processes but also associated with some diseases such as Alzheimer's disease, diabetes, and Parkinson's disease. However, since the experimental technologies are costly and time-consuming to detect the carbonylation sites in proteins, an accurate computational method for predicting carbonylation sites is an urgent issue which can be useful for drug development. In this study, a novel computational tool termed predCar-Site has been developed to predict protein carbonylation sites by (1) incorporating the sequence-coupled information into the general pseudo amino acid composition, (2) balancing the effect of skewed training dataset by Different Error Costs method, and (3) constructing a predictor using support vector machine as classifier. This predCar-Site predictor achieves an average AUC (area under curve) score of 0.9959, 0.9999, 1, and 0.9997 in predicting the carbonylation sites of K, P, R, and T, respectively. All of the experimental results along with AUC are found from the average of 5 complete runs of the 10-fold cross-validation and those results indicate significantly better performance than existing predictors. A user-friendly web server of predCar-Site is available at http://research.ru.ac.bd/predCar-Site/

Introduction

The structural and functional diversities of proteins as well as plasticity and dynamics of living cells are significantly dominated by the post-translational modifications (PTMs) [1]. Not only that, PTMs are also responsible for expanding the genetic code and for regulating cellular physiology as well [2,3]. A variety of PTMs such as hydroxylation, nitration, sulfhydrylation, carbonylation and glutathionylation have been induced from Oxidative stress [4] which is the direct result of imbalance in the production and degradation of reactive oxygen species (ROS) and reactive nitrogen species (RNS) [5]. Oxidative stress may occur when an excess production of reactive oxygen species (ROS) has surpassed the detoxification ability of cells and weakened the damage-repairing ability [5–8].

Among a variety of oxidative stress-induced PTMs, the protein carbonylation has been considered as a biomarker for oxidative stress due to its some unique characteristics such as relatively early formation, stability, and irreversibility [9,10]. However, the density of protein carbonylation increases with increase of external oxidative stress, aging and obesity which provides an indication of early stage of diseases [11,12]. Various types of major human diseases including Alzheimer's disease, diabetes, Parkinson's disease, chronic renal failure, chronic lung disease, sepsis are associated with protein carbolnylation [9,13].

As a result, the identification of carbonylation sites in proteins has become a vital question in cellular physiology and pathology, which in turns, helps in providing some valuable evidence for both biomedical research and drug development [5,6].

Mass spectrometry and liquid chromatography are the most common techniques to analyze protein susceptibility of the oxidative PTMs and determine its exact carbonylation sites recently [8,14,15]. It should be mentioned that among all the residues of protein molecules, four types of amino acid residues, namely lysine (K), proline (P), arginine (R), and threonine (T), have been found susceptible to carbonylation [16–18]. However, the purely experimental technique to determine the exact modified sites of carbonylated substrates is expensive as well as time-consuming,

* Corresponding author.
E-mail addresses: mehedi_ru@yahoo.com (Md.A.M. Hasan), Jinyan.Li@uts.edu.au (J. Li), shamim_cst@yahoo.com (S. Ahmad), khademul.cse@ru.ac.bd (Md.K.I. Molla).

http://dx.doi.org/10.1016/j.ab.2017.03.008
0003-2697/

Journal of Intelligent Learning Systems and Applications, **2014, 6, 45-52**
Published Online February 2014 (http://www.scirp.org/journal/jilsa)
http://dx.doi.org/10.4236/jilsa.2014.61005

Scientific Research

Support Vector Machine and Random Forest Modeling for Intrusion Detection System (IDS)

Md. Al Mehedi Hasan[1], Mohammed Nasser[2], Biprodip Pal[1], Shamim Ahmad[3]

[1]Department of Computer Science and Engineering, Rajshahi University of Engineering and Technology (RUET), Rajshahi, Bangladesh; [2]Department of Statistics, University of Rajshahi, Rajshahi, Bangladesh; [3]Department of Computer Science and Engineering, University of Rajshahi, Rajshahi, Bangladesh.
Email: mehedi_ru@yahoo.com, mnasser.ru@gmail.com, biprodip.cse@gmail.com, shamim_cst@yahoo.com

Received June 29th, 2013; revised July 29th, 2013; accepted August 7th, 2013

ABSTRACT

The success of any Intrusion Detection System (IDS) is a complicated problem due to its nonlinearity and the quantitative or qualitative network traffic data stream with many features. To get rid of this problem, several types of intrusion detection methods have been proposed and shown different levels of accuracy. This is why the choice of the effective and robust method for IDS is very important topic in information security. In this work, we have built two models for the classification purpose. One is based on Support Vector Machines (SVM) and the other is Random Forests (RF). Experimental results show that either classifier is effective. SVM is slightly more accurate, but more expensive in terms of time. RF produces similar accuracy in a much faster manner if given modeling parameters. These classifiers can contribute to an IDS system as one source of analysis and increase its accuracy. In this paper, KDD'99 Dataset is used to find out which one is the best intrusion detector for this dataset. Statistical analysis on KDD'99 dataset found important issues which highly affect the performance of evaluated systems and results in a very poor evaluation of anomaly detection approaches. The most important deficiency in the KDD'99 dataset is the huge number of redundant records. To solve these issues, we have developed a new dataset, KDD99Train+ and KDD99Test+, which does not include any redundant records in the train set as well as in the test set, so the classifiers will not be biased towards more frequent records. The numbers of records in the train and test sets are now reasonable, which make it affordable to run the experiments on the complete set without the need to randomly select a small portion. The findings of this paper will be very useful to use SVM and RF in a more meaningful way in order to maximize the performance rate and minimize the false negative rate.

KEYWORDS

Intrusion Detection; KDD'99; SVM; Kernel; Random Forest

1. Introduction

Along with the benefits, the Internet also created numerous ways to compromise the stability and security of the systems connecting to it. Although static defense mechanisms such as firewalls and software updates can provide a reasonable level of security, more dynamic mechanisms such as intrusion detection systems (IDSs) should also be utilized [1]. Intrusion detection is the process of monitoring events occurring in a computer system or network and analyzing them for signs of intrusions. IDSs are simply classified as host-based or network-based. The former operates on information collected from an individual computer system and the latter collects raw network packets and analyzes for signs of intrusions. There are two different detection techniques employed in IDS to search for attack patterns: Misuse and Anomaly. Misuse detection systems find known attack signatures in the monitored resources. Anomaly detection systems find attacks by detecting changes in the pattern of utilization or behavior of the system [2].

PERFORMANCE EVALUATION OF DIFFERENT KERNELS FOR SUPPORT VECTOR MACHINE USED IN INTRUSION DETECTION SYSTEM

Md. Al Mehedi Hasan[1], Shuxiang Xu[2], Mir Md. Jahangir Kabir[2] and Shamim Ahmad[1]

[1]Department of Computer Science and Engineering, University of Rajshahi, Bangladesh.
[2]School of Engineering and ICT, University of Tasmania, Australia.

ABSTRACT

The success of any Intrusion Detection System (IDS) is a complicated problem due to its nonlinearity and the quantitative or qualitative network traffic data stream with numerous features. As a result, in order to get rid of this problem, several types of intrusion detection methods with different levels of accuracy have been proposed which leads the choice of an effective and robust method for IDS as a very important topic in information security. In this regard, the support vector machine (SVM) has been playing an important role to provide potential solutions for the IDS problem. However, the practicability of introducing SVM is affected by the difficulties in selecting appropriate kernel and its parameters. From this viewpoint, this paper presents the work to apply different kernels for SVM in ID Son the KDD'99 Dataset and NSL-KDD dataset as well as to find out which kernel is the best for SVM. The important deficiency in the KDD'99 data set is the huge number of redundant records as observed earlier. Therefore, we have derived a data set RRE-KDD by eliminating redundant record from KDD'99train and test dataset prior to apply different kernel for SVM. This RRE-KDD consists of both KDD99Train+ and KDD99 Test+ dataset for training and testing purposes, respectively. The way to derive RRE-KDD data set is different from that of NSL-KDD data set. The experimental results indicate that Laplace kernel can achieve higher detection rate and lower false positive rate with higher precision than other kernel son both RRE-KDD and NSL-KDD datasets. It is also found that the performances of other kernels are dependent on datasets.

KEYWORDS

Intrusion Detection, KDD'99, NSL-KDD, Support Vector Machine, Kernel, Kernel Selection

1. INTRODUCTION

In spite of having great advantages of Internet, still then it has compromised the stability and security of the systems connected to it. Although static defense mechanisms such as firewalls and software updates can provide a reasonable level of security, more dynamic mechanisms such as intrusion detection systems (IDSs) should also be utilized [1]. Intrusion detection is the process of monitoring events occurring in a computer system or network and analyzing them for signs of

DOI: 10.5121/ijcnc.2016.8604

Bibliography

[1] J.-P. Vert, K. Tsuda, and B. Schölkopf, "A primer on kernel methods," *Kernel Methods in Computational Biology*, pp. 35–70, 2004.

[2] Z. R. Yang, "Biological applications of support vector machines," *Briefings in Bioinformatics*, vol. 5, no. 4, pp. 328–338, 2004.

[3] A. Afkanpour, C. Szepesvári, and M. Bowling, "Alignment based kernel learning with a continuous set of base kernels," *Machine Learning*, vol. 91, no. 3, pp. 305–324, 2013.

[4] A. Afkanpour, *Multiple Kernel Learning with Many Kernels*. PhD thesis, University of Alberta, 2013.

[5] M. Kloft, *Lp-norm multiple kernel learning: making learning with multiple kernels effective*. PhD thesis, Universitätsbibliothek der Technischen Universität Berlin, 2011.

[6] F. R. Bach, G. R. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proceedings of the Twenty-first International Conference on Machine Learning*, p. 6, ACM, 2004.

[7] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research*, vol. 7, no. Jul, pp. 1531–1565, 2006.

[8] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2491–2521, 2008.

[9] C. Cortes, M. Mohri, and A. Rostamizadeh, "Learning non-linear combinations of kernels," in *Advances in Neural Information Processing Systems*, pp. 396–404, 2009.

[10] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2211–2268, 2011.

[11] J. Thomas and L. Sael, "Overview of integrative analysis methods for heterogeneous data," in *Big Data and Smart Computing (BigComp), 2015 International Conference on*, pp. 266–270, IEEE, 2015.

[12] G. R. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.

[13] K.-C. Chou and H.-B. Shen, "Recent progress in protein subcellular location prediction," *Analytical Biochemistry*, vol. 370, no. 1, pp. 1–16, 2007.

[14] S. Wan, M.-W. Mak, and S.-Y. Kung, "mGOASVM: Multi-label protein subcellular localization based on gene ontology and support vector machines," *BMC Bioinformatics*, vol. 13, no. 1, p. 290, 2012.

[15] P. Du and C. Xu, "Predicting multisite protein subcellular locations: progress and challenges," *Expert Review of Proteomics*, vol. 10, no. 3, pp. 227–237, 2013.

[16] S. Wan, M.-W. Mak, and S.-Y. Kung, "GOASVM: A subcellular location predictor by incorporating term-frequency gene ontology into the general form of Chou's pseudo-amino acid composition," *Journal of Theoretical Biology*, vol. 323, pp. 40–48, 2013.

[17] R. Simha and H. Shatkay, "Protein (multi-) location prediction: using location inter-dependencies in a probabilistic framework," *Algorithms for Molecular Biology*, vol. 9, no. 1, p. 8, 2014.

[18] P. Horton, K.-J. Park, T. Obayashi, N. Fujita, H. Harada, C. Adams-Collier, and K. Nakai, "WoLF PSORT: protein localization predictor," *Nucleic Acids Research*, vol. 35, no. suppl 2, pp. W585–W587, 2007.

[19] K.-C. Chou and Y.-D. Cai, "Prediction and classification of protein subcellular locationâĂŤsequence-order effect and pseudo amino acid composition," *Journal of cellular biochemistry*, vol. 90, no. 6, pp. 1250–1260, 2003.

[20] X. Guo, F. Liu, Y. Ju, Z. Wang, and C. Wang, "Human protein subcellular localization with integrated source and multi-label ensemble classifier," *Scientific Reports*, vol. 6, 2016.

[21] A. Höglund, P. Dönnes, T. Blum, H.-W. Adolph, and O. Kohlbacher, "MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs and amino acid composition," *Bioinformatics*, vol. 22, no. 10, pp. 1158–1165, 2006.

[22] S. Wan, M.-W. Mak, and S.-Y. Kung, "GOASVM: A subcellular location predictor by incorporating term-frequency gene ontology into the general form of Chou's pseudo-amino acid composition," *Journal of Theoretical Biology*, vol. 323, pp. 40–48, 2013.

[23] M. S. Scott, D. Y. Thomas, and M. T. Hallett, "Predicting subcellular localization via protein motif co-occurrence," *Genome Research*, vol. 14, no. 10a, pp. 1957–1966, 2004.

[24] C. J. Shin, S. Wong, M. J. Davis, and M. A. Ragan, "Protein-protein interaction as a predictor of subcellular location," *BMC Systems Biology*, vol. 3, no. 1, p. 28, 2009.

[25] H.-N. Lin, C.-T. Chen, T.-Y. Sung, S.-Y. Ho, and W.-L. Hsu, "Protein subcellular localization prediction of eukaryotes using a knowledge-based approach," *BMC Bioinformatics*, vol. 10, no. 15, p. S8, 2009.

[26] S. Wan, M.-W. Mak, and S.-Y. Kung, "Sparse regressions for predicting and interpreting subcellular localization of multi-label proteins," *BMC Bioinformatics*, vol. 17, no. 1, p. 97, 2016.

[27] X. Wang, H. Li, Q. Zhang, and R. Wang, "Predicting Subcellular Localization of Apoptosis Proteins Combining GO Features of Homologous Proteins and Distance Weighted KNN Classifier," *BioMed Research International*, vol. 2016, 2016.

[28] K.-C. Chou and H.-B. Shen, "Euk-mPLoc: a fusion classifier for large-scale eukaryotic protein subcellular location prediction by incorporating multiple sites," *Journal of Proteome Research*, vol. 6, no. 5, pp. 1728–1734, 2007.

[29] X. Xiao, Z.-C. Wu, and K.-C. Chou, "A multi-label classifier for predicting the subcellular localization of gram-negative bacterial proteins with both single and multiple sites," *PloS One*, vol. 6, no. 6, p. e20592, 2011.

[30] S. Briesemeister, J. Rahnenführer, and O. Kohlbacher, "Going from where to why- interpretable prediction of protein subcellular localization," *Bioinformatics*, vol. 26, no. 9, pp. 1232–1238, 2010.

[31] R. Simha, S. Briesemeister, O. Kohlbacher, and H. Shatkay, "Protein (multi-) location prediction: utilizing interdependencies via a generative model," *Bioinformatics*, vol. 31, no. 12, pp. i365–i374, 2015.

[32] T. Blum, S. Briesemeister, and O. Kohlbacher, "MultiLoc2: integrating phylogeny and Gene Ontology terms improves subcellular protein localization prediction," *BMC Bioinformatics*, vol. 10, no. 1, p. 274, 2009.

[33] L. Li, Y. Zhang, L. Zou, C. Li, B. Yu, X. Zheng, and Y. Zhou, "An ensemble classifier for eukaryotic protein subcellular location prediction using gene ontology categories and amino acid hydrophobicity," *PLoS One*, vol. 7, no. 1, p. e31057, 2012.

[34] L. Zou, Z. Wang, and J. Huang, "Prediction of subcellular localization of eukaryotic proteins using position-specific profiles and neural network with weighted inputs," *Journal of Genetics and Genomics*, vol. 34, no. 12, pp. 1080–1087, 2007.

[35] J. He, H. Gu, and W. Liu, "Imbalanced multi-modal multi-label learning for subcellular localization prediction of human proteins with both single and multiple sites," *PloS One*, vol. 7, no. 6, p. e37155, 2012.

[36] X. Xiao, Z.-C. Wu, and K.-C. Chou, "iLoc-Virus: A multi-label learning classifier for identifying the subcellular localization of virus proteins with both single and multiple sites," *Journal of Theoretical Biology*, vol. 284, no. 1, pp. 42–51, 2011.

[37] X. Wang, R. Yan, and J. Song, "DephosSite: a machine learning approach for discovering phosphotase-specific dephosphorylation sites," *Scientific Reports*, vol. 6, 2016.

[38] S. Wan, M.-W. Mak, and S.-Y. Kung, "Ensemble linear neighborhood propagation for predicting subchloroplast localization of multi-location proteins," *Journal of Proteome Research*, vol. 15, no. 12, pp. 4755–4762, 2016.

[39] L. Li, H. Kuang, Y. Zhang, Y. Zhou, K. Wang, and Y. Wan, "Prediction of eukaryotic protein subcellular multi-localisation with a combined KNN-SVM ensemble classifier," *Journal of Computational Biology and Bioinformatics Research*, vol. 3, no. 2, pp. 15–24, 2011.

[40] A. Thakur, A. Rajput, and M. Kumar, "MSLVP: prediction of multiple subcellular localization of viral proteins using a support vector machine," *Molecular BioSystems*, vol. 12, no. 8, pp. 2572–2586, 2016.

[41] C.-Y. Yeh, W.-P. Su, and S.-J. Lee, "An efficient multiple-kernel learning for pattern classification," *Expert Systems with Applications*, vol. 40, no. 9, pp. 3491–3499, 2013.

[42] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," 2002.

[43] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine learning research*, vol. 5, no. Jan, pp. 27–72, 2004.

[44] C. Cortes, M. Mohri, and A. Rostamizadeh, "Two-stage learning kernel algorithms," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 239–246, 2010.

[45] N. ello Cristianini, A. Elisseeff, J. Shawe-Taylor, and J. Kandola, "On kernel-target alignment," *Advances in Neural Information processing systems*, 2001.

[46] X. Liu, L. Wang, J. Zhang, and J. Yin, "Sample-adaptive multiple kernel learning," in *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, pp. 1975–1981, 2014.

[47] M. Gönen and E. Alpaydin, "Localized multiple kernel learning," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 352–359, ACM, 2008.

[48] I. Dalle-Donne, D. Giustarini, R. Colombo, R. Rossi, and A. Milzani, "Protein carbonylation in human diseases," *Trends in Molecular Medicine*, vol. 9, no. 4, pp. 169–176, 2003.

[49] J. Jia, Z. Liu, X. Xiao, B. Liu, and K.-C. Chou, "pSuc-Lys: predict lysine succinylation sites in proteins with PseAAC and ensemble random forest approach," *Journal of Theoretical Biology*, vol. 394, pp. 223–230, 2016.

[50] W.-R. Qiu, X. Xiao, Z.-C. Xu, and K.-C. Chou, "iPhos-PseEn: identifying phosphorylation sites in proteins by fusing different pseudo components into an ensemble classifier," *Oncotarget*, vol. 7, no. 32, pp. 51270–51283, 2016.

[51] J. Jia, Z. Liu, X. Xiao, B. Liu, and K.-C. Chou, "iSuc-PseOpt: identifying lysine succinylation sites in proteins by incorporating sequence-coupling effects into pseudo components and optimizing imbalanced training dataset," *Analytical Biochemistry*, vol. 497, pp. 48–56, 2016.

[52] H. D. Ismail, A. Jones, J. H. Kim, R. H. Newman, and D. B. Kc, "RF-Phos: A Novel General Phosphorylation Site Prediction Tool Based on Random Forest," *BioMed Research International*, vol. 2016, 2016.

[53] S.-Y. Huang, S.-P. Shi, J.-D. Qiu, and M.-C. Liu, "Using support vector machines to identify protein phosphorylation sites in viruses," *Journal of Molecular Graphics and Modelling*, vol. 56, pp. 84–90, 2015.

[54] W.-R. Qiu, B.-Q. Sun, X. Xiao, D. Xu, and K.-C. Chou, "iPhos-PseEvo: Identifying Human Phosphorylated Proteins by Incorporating Evolutionary Information into General PseAAC via Grey System Theory," *Molecular Informatics*, 2016.

[55] W.-R. Qiu, Q.-S. Zheng, B.-Q. Sun, and X. Xiao, "Multi-iPPseEvo: A Multi-label Classifier for Identifying Human Phosphorylated Proteins by Incorporating Evolutionary Information into Chou's General PseAAC via Grey System Theory," *Molecular Informatics*, 2016.

[56] F. Dinuzzo and G. De Nicolao, *Learning functions with kernel methods*. PhD thesis, University of Pavia, 2011.

[57] B. Schölkopf and A. J. Smola, "Learning with kernels," 2002.

[58] T. T. H. Do, *A unified framework for Support Vector Machines, Multiple Kernel Learning and metric learning*. PhD thesis, University of Geneva, 2012.

[59] V. N. Vapnik, *The nature of statistical learning theory*. Springer, New York, 2 ed., 1999.

[60] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[61] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (IDS)," *Journal of Intelligent Learning Systems and Applications*, vol. 6, no. 1, p. 45, 2014.

[62] G. Tsoumakas, I. Katakis, and I. P. Vlahavas, "Mining Multi-label Data," in *Data Mining and Knowledge Discovery Handbook, 2nd ed.*, pp. 667–685, 2010.

[63] X. Wang, J. Zhang, and G.-Z. Li, "Multi-location gram-positive and gram-negative bacterial protein subcellular localization using gene ontology and multi-label classifier ensemble," *BMC Bioinformatics*, vol. 16, no. 12, p. S1, 2015.

[64] C.-W. Hsu, C.-C. Chang, C.-J. Lin, *et al.*, "A practical guide to support vector classification," *Technical Report, National Taiwan University*, 2003.

[65] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 131–159, 2002.

[66] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," *Data Mining Techniques for the Life Sciences*, pp. 223–239, 2010.

[67] C. S. Ong, A. J. Smola, and R. C. Williamson, "Learning the kernel with hyperkernels," *Journal of Machine Learning Research*, vol. 6, no. Jul, pp. 1043–1071, 2005.

[68] A. Argyriou, C. A. Micchelli, and M. Pontil, "Learning convex combinations of continuously parameterized basic kernels," in *International Conference on Computational Learning Theory*, pp. 338–352, Springer, 2005.

[69] S. C. Hoi, R. Jin, and M. R. Lyu, "Learning nonparametric kernel matrices from pairwise constraints," in *Proceedings of the 24th International Conference on Machine learning*, pp. 361–368, ACM, 2007.

[70] B. Kulis, M. Sustik, and I. Dhillon, "Learning low-rank kernel matrices," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 505–512, ACM, 2006.

[71] J. R. Searle, "Minds, brains, and programs," *Behavioral and brain sciences*, vol. 3, no. 03, pp. 417–424, 1980.

[72] P. Gehler and S. Nowozin, "Infinite kernel learning," in *Proceedings of the NIPS 2008 Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, 2008.

[73] Z. Xu, R. Jin, H. Yang, I. King, and M. R. Lyu, "Simple and efficient multiple kernel learning by group lasso," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 1175–1182, 2010.

[74] M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg, "Non-sparse multiple kernel learning," in *NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, vol. 4, 2008.

[75] M. Kloft, U. Brefeld, P. Laskov, K.-R. Müller, A. Zien, and S. Sonnenburg, "Efficient and accurate lp-norm multiple kernel learning," in *Advances in Neural Information Processing Systems*, pp. 997–1005, 2009.

[76] A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil, "A DC-programming algorithm for kernel selection," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 41–48, ACM, 2006.

[77] A. Zien and C. S. Ong, "Multiclass multiple kernel learning," in *Proceedings of the 24th International Conference on Machine learning*, pp. 1191–1198, ACM, 2007.

[78] J. Ye, J. Chen, and S. Ji, "Discriminant kernel and regularization parameter learning via semidefinite programming," in *Proceedings of the 24th International Conference on Machine Learning*, pp. 1095–1102, ACM, 2007.

[79] X. Liu, L. Zhou, L. Wang, J. Zhang, J. Yin, and D. Shen, "An efficient radius-incorporated MKL algorithm for Alzheimer's disease prediction," *Pattern Recognition*, vol. 48, no. 7, pp. 2141–2150, 2015.

[80] H. Do, A. Kalousis, A. Woznica, and M. Hilario, "Margin and radius based multiple kernel learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 330–343, Springer, 2009.

[81] N. Christianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge Univ Press, 2000.

[82] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy, "Gene functional classification from heterogeneous data," in *Proceedings of the Fifth Annual International Conference on Computational Biology*, pp. 249–255, ACM, 2001.

[83] I. M. de Diego, A. Muñoz, and J. M. Moguerza, "Methods for the combination of kernel matrices within a support vector framework," *Machine Learning*, vol. 78, no. 1-2, p. 137, 2010.

[84] H. Tanabe, T. B. Ho, C. H. Nguyen, and S. Kawasaki, "Simple but effective methods for combining kernels in computational biology," in *Research, Innovation and Vision for the Future, 2008. RIVF 2008. IEEE International Conference on*, pp. 71–78, IEEE, 2008.

[85] S. Qiu and T. Lane, "A framework for multiple kernel support vector regression and its applications to siRNA efficacy prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 6, no. 2, pp. 190–199, 2009.

[86] J. Kandola, J. Shawe-Taylor, and N. Cristianini, "Optimizing kernel alignment over combinations of kernels," tech. rep., Department of Computer Science, University of London, 2002.

[87] J. He, S.-F. Chang, and L. Xie, "Fast kernel learning for spatial pyramid matching," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–7, IEEE, 2008.

[88] C. H. Nguyen and T. B. Ho, "An efficient kernel matrix evaluation measure," *Pattern Recognition*, vol. 41, no. 11, pp. 3366–3372, 2008.

[89] Y. Ying, K. Huang, and C. Campbell, "Enhanced protein fold recognition through a novel data integration approach," *BMC Bioinformatics*, vol. 10, no. 1, p. 267, 2009.

[90] M. Varma and B. R. Babu, "More generality in efficient multiple kernel learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1065–1072, ACM, 2009.

[91] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao, "A new multiple kernel approach for visual concept learning," in *International Conference on Multimedia Modeling*, pp. 250–262, Springer, 2009.

[92] J. Yang, Y. Li, Y. Tian, L.-Y. Duan, and W. Gao, "Per-sample multiple kernel approach for visual concept learning," *Journal on Image and Video Processing*, vol. 2010, p. 2, 2010.

[93] M. Girolami and S. Rogers, "Hierarchic Bayesian models for kernel learning," in *Proceedings of the 22nd International Conference on Machine Learning*, pp. 241–248, ACM, 2005.

[94] K. P. Bennett, M. Momma, and M. J. Embrechts, "MARK: A boosting algorithm for heterogeneous kernel models," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 24–31, ACM, 2002.

[95] J. Bi, T. Zhang, and K. P. Bennett, "Column-generation boosting methods for mixture of kernels," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 521–526, ACM, 2004.

[96] J. Zhuang, J. Wang, S. C. Hoi, and X. Lan, "Unsupervised multiple kernel learning," *Journal of Machine Learning Research-Proceedings Track*, vol. 20, pp. 129–144, 2011.

[97] S. Ji, L. Sun, R. Jin, and J. Ye, "Multi-label multiple kernel learning," in *Advances in Neural Information Processing Systems*, pp. 777–784, 2009.

[98] S. K. Pal, S. Bandyopadhyay, and S. S. Ray, "Evolutionary computation in bioinformatics: a review," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 36, no. 5, pp. 601–615, 2006.

[99] A. E. Hassanien, M. G. Milanova, T. G. Smolinski, and A. Abraham, "Computational Intelligence in Solving Bioinformatics Problems: Reviews, Perspectives, and Challenges," in *Computational Intelligence in Biomedicine and Bioinformatics, Current Trends and Applications*, pp. 3–47, 2008.

[100] H.-J. Bockenhauer and D. Bongartz, *Algorithmic Aspects of Bioinformatics (Natural Computing Series)*. Springer, 2007.

[101] M. F. Sanders and J. L. Bowman, *Genetic Analysis: An Integrated Approach*. Pearson, 2015.

[102] C. Gibas and P. Jambeck, *Developing bioinformatics computer skills - an introduction to software tools for biological applications*. O'Reilly, 2001.

[103] K. Kadam, S. Sawant, U. Kulkarni-Kale, and V. K. Jayaraman, "Prediction of protein function based on machine learning methods: an overview," *Genomics III Methods, Techniques and Applications*, pp. 1–38, 2014.

[104] W.-R. Qiu, B.-Q. Sun, X. Xiao, Z.-C. Xu, and K.-C. Chou, "iPTM-mLys: identifying multiple lysine PTM sites and their different types," *Bioinformatics*, vol. 32, no. 20, pp. 3116–3123, 2016.

[105] J. Jia, Z. Liu, X. Xiao, B. Liu, and K.-C. Chou, "iCar-PseCp: identify carbonylation sites in proteins by Monto Carlo sampling and incorporating sequence coupled effects into general PseAAC," *Oncotarget*, vol. 7, no. 23, pp. 34558–34570, 2016.

[106] H. Lv, J. Han, J. Liu, J. Zheng, R. Liu, and D. Zhong, "CarSPred: a computational tool for predicting carbonylation sites of human proteins," *PloS One*, vol. 9, no. 10, p. e111478, 2014.

[107] Y. Xu, X. Wang, Y. Wang, Y. Tian, X. Shao, L.-Y. Wu, and N. Deng, "Prediction of posttranslational modification sites from amino acid sequences with kernel methods," *Journal of Theoretical Biology*, vol. 344, pp. 78–87, 2014.

[108] Y. Y. Nancy, J. R. Wagner, M. R. Laird, G. Melli, S. Rey, R. Lo, P. Dao, S. C. Sahinalp, M. Ester, L. J. Foster, *et al.*, "Psortb 3.0: improved protein subcellular localization prediction with refined localization subcategories and predictive capabilities for all prokaryotes," *Bioinformatics*, vol. 26, no. 13, pp. 1608–1615, 2010.

[109] N. Xiao, D.-S. Cao, M.-F. Zhu, and Q.-S. Xu, "protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences," *Bioinformatics*, p. btv042, 2015.

[110] K.-C. Chou, "Prediction of protein cellular attributes using pseudo-amino acid composition," *Proteins: Structure, Function, and Bioinformatics*, vol. 43, no. 3, pp. 246–255, 2001.

[111] X. Qu, D. Wang, Y. Chen, S. Qiao, and Q. Zhao, "Predicting the subcellular localization of proteins with multiple sites based on multiple features fusion," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 13, no. 1, pp. 36–42, 2016.

[112] K.-C. Chou and H.-B. Shen, "A new method for predicting the subcellular localization of eukaryotic proteins with both single and multiple sites: Euk-mPLoc 2.0," *PLoS One*, vol. 5, no. 4, p. e9931, 2010.

[113] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[114] W.-Z. Lin, J.-A. Fang, X. Xiao, and K.-C. Chou, "iLoc-Animal: a multi-label learning classifier for predicting subcellular localization of animal proteins," *Molecular BioSystems*, vol. 9, no. 4, pp. 634–644, 2013.

[115] K.-C. Chou, Z.-C. Wu, and X. Xiao, "iLoc-Euk: a multi-label classifier for predicting the subcellular localization of singleplex and multiplex eukaryotic proteins," *PloS One*, vol. 6, no. 3, p. e18258, 2011.

[116] K.-C. Chou, "Structural bioinformatics and its impact to biomedical science," *Current Medicinal Chemistry*, vol. 11, no. 16, pp. 2105–2134, 2004.

[117] A. A. Schäffer, L. Aravind, T. L. Madden, S. Shavirin, J. L. Spouge, Y. I. Wolf, E. V. Koonin, and S. F. Altschul, "Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements," *Nucleic Acids Research*, vol. 29, no. 14, pp. 2994–3005, 2001.

[118] C. T. Walsh, S. Garneau-Tsodikova, and G. J. Gatto, "Protein posttranslational modifications: the chemistry of proteome diversifications," *Angewandte Chemie International Edition*, vol. 44, no. 45, pp. 7342–7372, 2005.

[119] C. Walsh, *Posttranslational modification of proteins: expanding nature's inventory*. Roberts and Company Publishers, 2006.

[120] J. Hu, Y. Guo, and Y. Li, "Research progress in protein post-translational modification," *Chinese Science Bulletin*, vol. 51, no. 6, pp. 633–645, 2006.

[121] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, *et al.*, "The sequence of the human genome," *Science*, vol. 291, no. 5507, pp. 1304–1351, 2001.

[122] G. A. Khoury, R. C. Baliban, and C. A. Floudas, "Proteome-wide post-translational modification statistics: frequency analysis and curation of the swiss-prot database," *Scientific Reports*, vol. 1, p. 90, 2011.

[123] H. Kaufmann, J. E. Bailey, and M. Fussenegger, "Use of antibodies for detection of phosphorylated proteins separated by two-dimensional gel electrophoresis," *Proteomics*, vol. 1, no. 2, pp. 194–199, 2001.

[124] Y. Xu, Y.-X. Ding, J. Ding, Y.-H. Lei, L.-Y. Wu, and N.-Y. Deng, "iSuc-PseAAC: predicting lysine succinylation in proteins by incorporating peptide position-specific propensity," *Scientific Reports*, vol. 5, 2015.

[125] Y. Xu, J. Ding, and L.-Y. Wu, "iSulf-Cys: Prediction of S-sulfenylation Sites in Proteins with Physicochemical Properties of Amino Acids," *PloS One*, vol. 11, no. 4, p. e0154237, 2016.

[126] Y. Xu, Y.-X. Ding, J. Ding, L.-Y. Wu, and Y. Xue, "Mal-Lys: prediction of lysine malonylation sites in proteins integrated sequence-based features with mRMR feature selection," *Scientific Reports*, vol. 6, p. 38318, 2016.

[127] K. Veropoulos, C. Campbell, N. Cristianini, *et al.*, "Controlling the sensitivity of support vector machines," in *Proceedings of the International Joint Conference on AI*, pp. 55–60, 1999.

[128] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *European Conference on Machine Learning*, pp. 39–50, Springer, 2004.

[129] R. Batuwita and V. Palade, "Efficient resampling methods for training support vector machines with imbalanced datasets," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pp. 1–8, IEEE, 2010.

[130] Z. Ju, J.-Z. Cao, and H. Gu, "Predicting lysine phosphoglycerylation with fuzzy SVM by incorporating k-spaced amino acid pairs into Chou's general PseAAC," *Journal of Theoretical Biology*, vol. 397, pp. 145–150, 2016.

[131] K.-C. Chou, "Some remarks on protein attribute prediction and pseudo amino acid composition," *Journal of Theoretical Biology*, vol. 273, no. 1, pp. 236–247, 2011.

[132] Y. Xu, Y.-X. Ding, N.-Y. Deng, and L.-M. Liu, "Prediction of sumoylation sites in proteins using linear discriminant analysis," *Gene*, vol. 576, no. 1, pp. 99–104, 2016.

[133] B. Liu, Y. Liu, X. Jin, X. Wang, and B. Liu, "iRSpot-DACC: a computational predictor for recombination hot/cold spots identification based on dinucleotide-based auto-cross covariance," *Scientific Reports*, vol. 6, 2016.

[134] Z. Liao, Y. Ju, and Q. Zou, "Prediction of G protein-coupled receptors with SVM-prot features and random forest," *Scientifica*, vol. 2016, 2016.

[135] H.-B. Shen and K.-C. Chou, "Gneg-mPLoc: a top-down strategy to enhance the quality of predicting subcellular localization of Gram-negative bacterial proteins," *Journal of Theoretical Biology*, vol. 264, no. 2, pp. 326–333, 2010.

[136] E. Camon, M. Magrane, D. Barrell, D. Binns, W. Fleischmann, P. Kersey, N. Mulder, T. Oinn, J. Maslen, A. Cox, *et al.*, "The gene ontology annotation (GOA) project: implementation of GO in SWISS-PROT, TrEMBL, and InterPro," *Genome Research*, vol. 13, no. 4, pp. 662–672, 2003.

[137] T. Ishii, S. Ito, S. Kumazawa, T. Sakurai, S. Yamaguchi, T. Mori, T. Nakayama, and K. Uchida, "Site-specific modification of positively-charged surfaces on human serum albumin by malondialdehyde," *Biochemical and Biophysical Research Communications*, vol. 371, no. 1, pp. 28–32, 2008.

[138] A. G. Madian, N. Diaz-Maldonado, Q. Gao, and F. E. Regnier, "Oxidative stress induced carbonylation in human plasma," *Journal of Proteomics*, vol. 74, no. 11, pp. 2395–2416, 2011.

[139] H. Mirzaei and F. Regnier, "Affinity chromatographic selection of carbonylated proteins followed by identification of oxidation sites using tandem mass spectrometry," *Analytical Chemistry*, vol. 77, no. 8, pp. 2386–2392, 2005.

[140] J. Chen, H. Liu, J. Yang, and K.-C. Chou, "Prediction of linear B-cell epitopes using amino acid pair antigenicity scale," *Amino Acids*, vol. 33, no. 3, pp. 423–428, 2007.

[141] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[142] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proceedings of the 23rd International Conference on Machine learning*, pp. 233–240, ACM, 2006.

[143] X. Zhao, Q. Ning, H. Chai, and Z. Ma, "Accurate in silico identification of protein succinylation sites using an iterative semi-supervised learning technique," *Journal of Theoretical Biology*, vol. 374, pp. 60–65, 2015.

[144] Z. Liu, Y. Wang, T. Gao, Z. Pan, H. Cheng, Q. Yang, Z. Cheng, A. Guo, J. Ren, and Y. Xue, "CPLM: a database of protein lysine modifications," *Nucleic Acids Research*, vol. 42, no. D1, pp. D531–D536, 2014.

[145] K.-C. Chou, "Some remarks on predicting multi-label attributes in molecular biosystems," *Molecular Biosystems*, vol. 9, no. 6, pp. 1092–1100, 2013.

[146] C. Huang and J.-Q. Yuan, "A multilabel model based on Chou's pseudo–amino acid composition for identifying membrane proteins with both single and multiple functional types," *The Journal of membrane biology*, vol. 246, no. 4, pp. 327–334, 2013.

[147] X. Xiao, P. Wang, W.-Z. Lin, J.-H. Jia, and K.-C. Chou, "iAMP-2L: a two-level multi-label classifier for identifying antimicrobial peptides and their functional types," *Analytical Biochemistry*, vol. 436, no. 2, pp. 168–177, 2013.

[148] J. Jia, L. Zhang, Z. Liu, X. Xiao, and K.-C. Chou, "pSumo-CD: Predicting sumoylation sites in proteins with covariance discriminant algorithm by incorporating sequence-coupled effects into general PseAAC," *Bioinformatics*, vol. 32, no. 20, pp. 3133–3141, 2016.

[149] K.-C. Chou, "Prediction of human immunodeficiency virus protease cleavage sites in proteins," *Analytical Biochemistry*, vol. 233, no. 1, pp. 1–14, 1996.

[150] S. Zhang, X. Xia, J. Shen, Y. Zhou, and Z. Sun, "DBMLoc: a Database of proteins with multiple subcellular localizations," *BMC Bioinformatics*, vol. 9, 2008.

[151] S. Briesemeister, J. Rahnenführer, and O. Kohlbacher, "YLoc-an interpretable web server for predicting subcellular localization," *Nucleic Acids Research*, vol. 38, no. suppl 2, pp. W497–W502, 2010.

[152] K. Duan, S. S. Keerthi, and A. N. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," *Neurocomputing*, vol. 51, pp. 41–59, 2003.

[153] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 795–828, 2012.

[154] W.-L. Huang, C.-W. Tung, S.-W. Ho, S.-F. Hwang, and S.-Y. Ho, "ProLoc-GO: utilizing informative Gene Ontology terms for sequence-based prediction of protein subcellular localization," *BMC Bioinformatics*, vol. 9, no. 1, p. 80, 2008.

[155] Z.-C. Wu, X. Xiao, and K.-C. Chou, "iLoc-Gpos: a multi-layer classifier for predicting the subcellular localization of singleplex and multiplex Gram-positive bacterial proteins," *Protein and Peptide Letters*, vol. 19, no. 1, pp. 4–14, 2012.

[156] A. Dehzangi, R. Heffernan, A. Sharma, J. Lyons, K. Paliwal, and A. Sattar, "Gram-positive and Gram-negative protein subcellular localization by incorporating evolutionary-based descriptors into Chou's general PseAAC," *Journal of Theoretical Biology*, vol. 364, pp. 284–294, 2015.

[157] H.-B. Shen and K.-C. Chou, "Gpos-mPLoc: a top-down approach to improve the quality of predicting subcellular localization of Gram-positive bacterial proteins," *Protein and Peptide Letters*, vol. 16, no. 12, pp. 1478–1484, 2009.

[158] E. Pacharawongsakda and T. Theeramunkong, "Predict subcellular locations of singleplex and multiplex proteins by semi-supervised learning and dimension-reducing general mode of Chou's PseAAC," *IEEE Transactions on Nanobioscience*, vol. 12, no. 4, pp. 311–320, 2013.

[159] K. Tanaka, M. Soeda, Y. Hashimoto, S. Takenaka, and M. Komori, "Identification of phosphorylation sites in Hansenula polymorpha Pex14p by mass spectrometry," *FEBS Open Bio*, vol. 3, no. 1, pp. 6–10, 2013.

[160] Y. Dou, B. Yao, and C. Zhang, "PhosphoSVM: prediction of phosphorylation sites by integrating various protein sequence attributes with a support vector machine," *Amino Acids*, vol. 46, no. 6, pp. 1459–1469, 2014.

[161] S. Mei, "Multi-label multi-kernel transfer learning for human protein subcellular localization," *PLoS One*, vol. 7, no. 6, p. e37716, 2012.

[162] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, 2006.

[163] M. A. Tahir, J. Kittler, and A. Bouridane, "Multilabel classification using heterogeneous ensemble of multi-label classifiers," *Pattern Recognition Letters*, vol. 33, no. 5, pp. 513–523, 2012.