The Hong Kong Polytechnic University

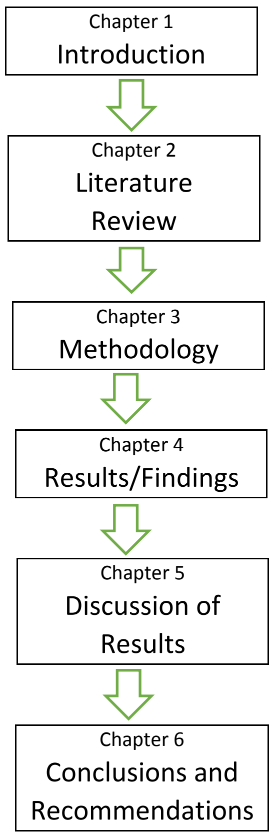
Department of Electronic and Information Engineering

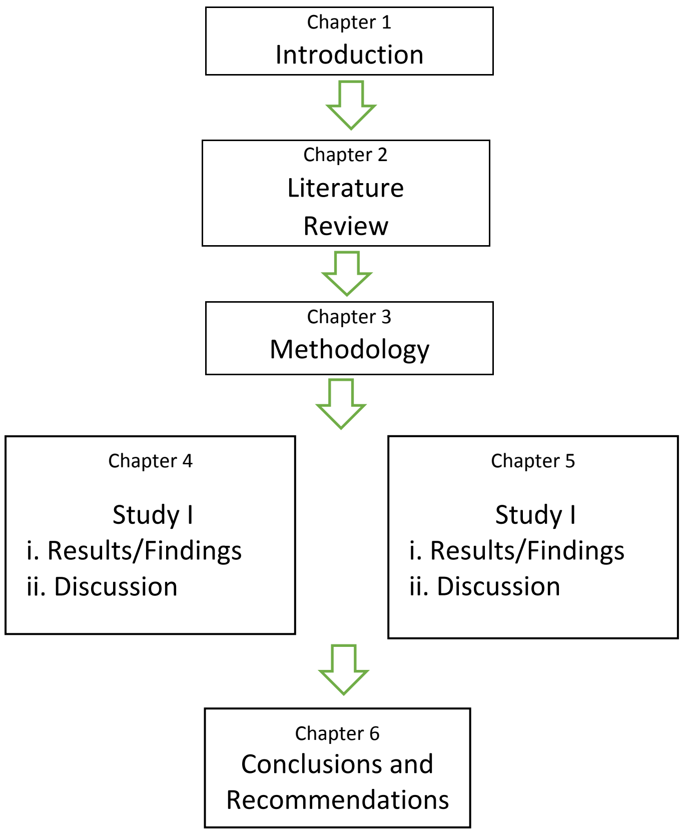
Protein Subcellular Localization: Gene Ontology Based Machine Learning Approaches

A thesis submitted in partial fulfillment of the requirements of the Degree of Doctor of Philosophy

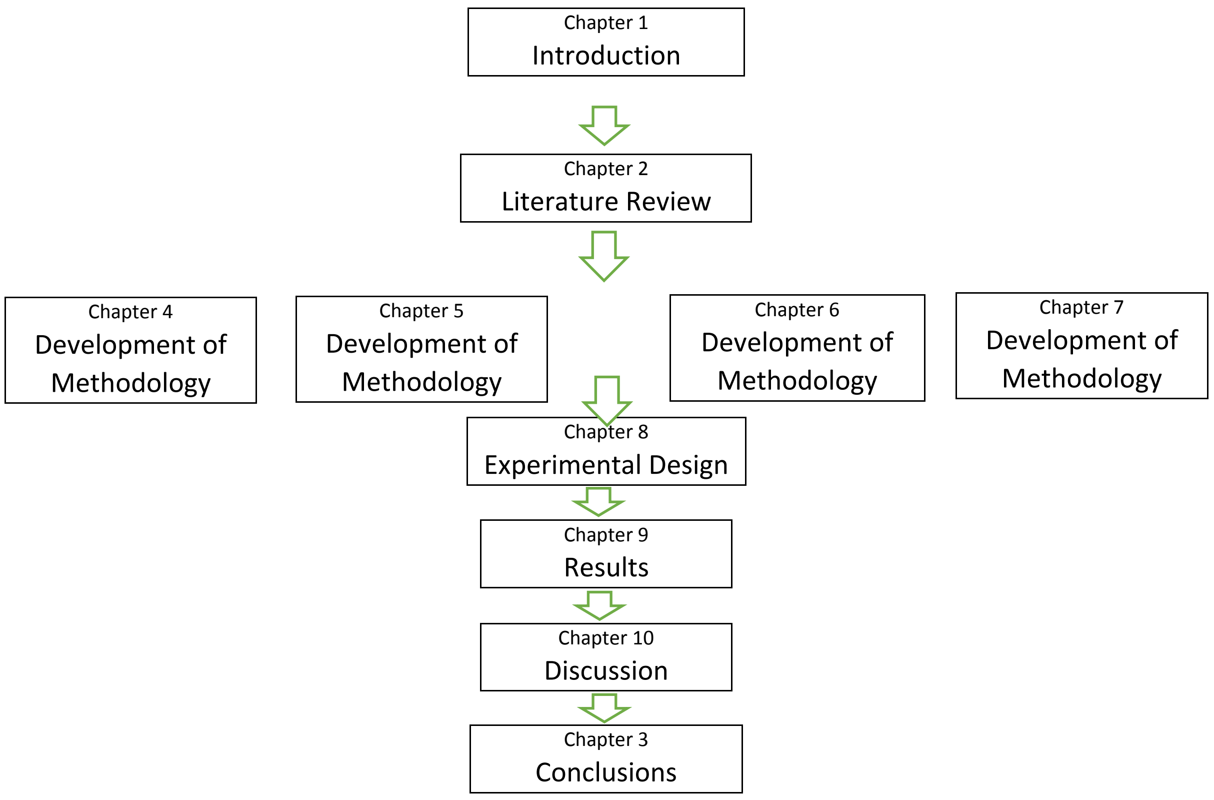
2014

Theses are usually organised in the following ways:





This thesis is organised as shown here:



Abstract

The abstract is the first chapter of a thesis and should be a short summary of the entire thesis. Its purpose is to attract people to read your thesis. Abstracts normally summarise aspects of the research including: the background, the importance of topic, the purpose of the research, the methodology used in the study, the key findings and the implications the findings will have. The abstract should be very short and precise.

This abstract is very effective partly because the writer includes the following:

Structure

Background	paragraph 1 sentence 1-3
Aims	paragraph 1, sentence 4
Methods	paragraph 2-4
Findings	paragraph 4, points 2-3

Content

- Explains key terms (e.g. paragraph 1, sentence 1)
- Highlights problem to be addressed (e.g. paragraph 1, sentence 2)
- Highlights suggested solution (e.g. paragraph 1, sentence 3-4)
- Introduces abbreviations of key terms that appear in the thesis (e.g. paragraph 1 sentence 5)
- Explains rationale for methodology used e.g. *to overcome come this limitation* (e.g. paragraph 4, point 2, sentence 1 and 2)

Language

- Uses adjectives to show importance of topic, e.g. *laborious, time-consuming, accurate, fast, reliable* (e.g. paragraph 1, sentence 4)
- Uses past tense and passive voice to describe the methodology and findings in this thesis, e.g. *was found* (e.g. paragraph 3, final sentence)
- Uses vocabulary to show importance of topic and findings e.g. *impressively reducing* (e.g. paragraph 4, point 3, final sentence)

To Consider

This abstract is effective but could be improved in the following aspects.

- Highlight the gap in the current understanding of the problem.
- Outline the scope of the research.
- Use a new paragraph for the summary of each investigation.
- State the importance of the work.
- Highlight how the findings can be used in further research.
- Explain how the findings will aid the wider community.
- Explain the limitations of the research.
- Avoid the overuse sentences starting with *for*, e.g. *For the prediction of single-location proteins,* (e.g. paragraph 2, sentence 1). It is better to state the main point first.
- Avoid including too much detail in the abstract.

Abstract

Proteins, which are essential macromolecules for organisms, need to be located in appropriate physiological contexts within a cell to exhibit tremendous diversity of biological functions. Aberrant protein subcellular localization may lead to a broad range of diseases. Knowing where a protein resides within a cell can give insights on drug target discovery and drug design. Computational methods are required to assist the laborious and time-consuming conventional wet-lab experiments for accurate, fast, reliable and large-scale predictions in proteomics research. This thesis proposes several Gene Ontology (GO) based machine learning approaches for the prediction of subcellular localization of both single-location and multi-location proteins.

For the prediction of single-location proteins, two GO-based single-label predictors, namely GOASVM and FusionSVM, are proposed. GOASVM exploits GO information from the gene ontology annotation (GOA) database while FusionSVM extracts GO information from InterProScan and then combines GO information with profile alignment information. It was found that GOASVM (extracting GO from the GOA database) performs significantly better than FusionSVM (extracting GO from InterProScan). Moreover, GOASVM also remarkably outperforms existing state-of-the-art single-label predictors.

For the prediction of multi-location proteins, an efficient multi-label predictor, namely mGOASVM, is proposed. mGOASVM extends GOASVM from single-location prediction to multi-location prediction. It possesses the following desirable properties: (1) it uses the frequency of occurrences of GO terms instead of 1-0 values; (2) it uses a more efficient

multi-label SVM classifier to handle multi-label problems; and (3) it selects a relevant GO-vector subspace by finding distinct GO terms instead of using the full GO-vector space; (4) it adopts a successive-search strategy to incorporate more useful homologous information for classification. It was found that these properties make mGOASVM outperform other GO-based multi-label predictors.

Based on mGOASVM, several more advanced multi-label predictors are proposed. These predictors further improve the performance of mGOASVM by enhancing the following aspects of the prediction process:

1. **Classification Refinement.** The classifier adopted by mGOASVM to tackle multi-label problems is rather primitive, thus refining the classification process is necessary. To this end, two multi-label predictors, namely AD-SVM and mPLR-Loc, are proposed. The former adopts an adaptive decision scheme for multi-label SVM classification. The scheme essentially converts the linear SVMs in the classifier into piecewise linear SVMs, which effectively reduces the over-prediction instances while having little influence on the correctly predicted ones, thus improving the prediction performance. The latter adopts a multi-label penalized logistic regression classifier equipped with an adaptive decision scheme, which can also boost the performance.

2. **Deeper Feature Extraction.** mGOASVM only considers the frequency of occurrences of GO terms, which may not be sufficient for accurate prediction. To overcome this limitation, a multi-label predictor called SS-Loc, which further exploits the semantic similarity over GO, is proposed. Based on SS-Loc, an even more advanced predictor called HybridGO-Loc, which uses both GO frequency features and GO semantic similarity features, is developed. Experimental results demonstrate that HybridGO-Loc performs the best among all of the proposed multi-label

predictors as well as other existing GO-based predictors.

3. **Dimensionality Reduction.** Although a relevant GO-vector subspace has been selected, the feature vectors in mGOASVM are still of high dimensionality. To address the problem of the curse of high dimensionality, an ensemble method based on random projection (RP) is applied to construct two dimensionality-reduction multi-label predictors, namely RP-SVM and R3P-Loc. The former uses multi-label SVM classifiers and the latter uses multi-label ridge regression classifiers. Experimental results suggest that both predictors outperform mGOASVM as well as other state-of-the-art predictors while at the same time impressively reducing the dimensions.

Contents

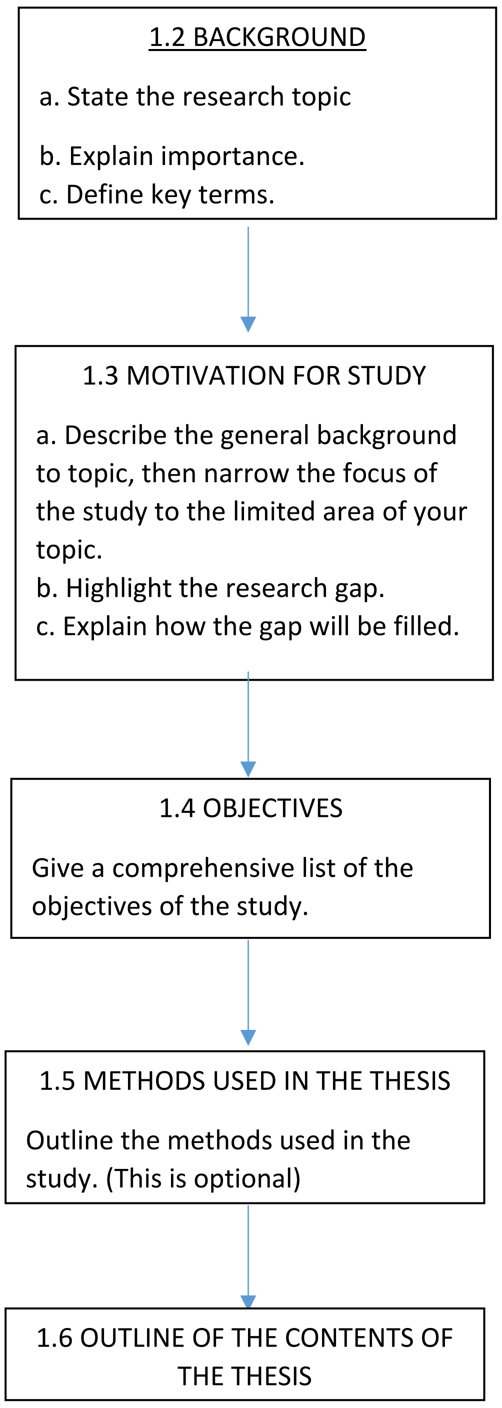
List of Figures

List of Tables

The Introduction

The Introduction is usually organised in the following way:

1.1 INTRODUCTION



1.2 BACKGROUND

a. State the research topic

b. Explain importance.
c. Define key terms.

1.3 MOTIVATION FOR STUDY

a. Describe the general background to topic, then narrow the focus of the study to the limited area of your topic.
b. Highlight the research gap.
c. Explain how the gap will be filled.

1.4 OBJECTIVES

Give a comprehensive list of the objectives of the study.

1.5 METHODS USED IN THE THESIS

Outline the methods used in the study. (This is optional)

1.6 OUTLINE OF THE CONTENTS OF THE THESIS

1.7 SUMMARY

Chapter 1: Introduction

The introduction chapter is designed to help the reader better understand the technical sections of the thesis by providing an overview of the research. It normally includes the following: the reasons for the research, the scope, the scientific importance of the research, the introduction, explanation and definition of key terms, an introduction of the most important current studies in your field, an overview of the methodology used and an overview of the way the thesis is organized with a short summary of each chapter.

This introduction is very effective partly because the writer includes the following:

Structure

Introduction	Chapter 1
Background	Section 1.1
Key terms	Section 1.1
Motivation for Study	Section 1.2.1
Current Methods	Section 1.2.2
Research Gap	Section 1.2.3
Methodology	Section 1.2.3
Outline of the Thesis	Section 1.3
(Summary)	Not included

Content

- Describes background, highlighting the importance of the topic (paragraph 1,
  Explains key terms (e.g. Section 1.1, paragraph 1, sentence 1-4)
- Gives an example to aid explanation (e.g. Section 1.1, paragraph 1, sentence 5)
- Develops explanation from more general background on key terms (e.g. Section 1.1, paragraph 1) to more specific detail (e.g. Section 1.1, paragraph 3)
- Cites key studies in the field (e.g. Section 1.1, paragraph 1, sentence 4)
- Describes problem thesis seeks to address (e.g. Section 1.2)
- States benefits study will bring to the wider community (e.g. Section 1.2.1, paragraph 1, sentence 4)
- Introduces key techniques used in existing research (e.g. Section 1.2.2)

- Refers to information provided in previous section when developing the argument to identify the research gap (e.g. Section 1.2.3, Paragraph 1, sentence 1)
- Highlights limitations of current methodology (e.g. Section 1.2.3, paragraph 1, first and final sentences)
- Develops a coherent argument for approach to the research (e.g. Section 1.2.3)
- States why the proposed methodology is required (e.g. Section 1.2.3, paragraph 2, sentence 1-2)

Language

- Uses present perfect to describe the background to the topic, e.g. *have witnessed* (e.g.paragraph 1, sentence 2)
- Introduces subsection 1.2 with a short introductory paragraph (e.g. Section 1.2, paragraph 1)
- Uses adjectives to highlight the importance of the topic *essential and indispensable* (e.g. Section 1.1, paragraph 1, sentence 1 and 4)
- Outlines the structure of the thesis (e.g. Section 1.3)
- Introduces key words and abbreviations, e.g. *gene ontology (GO)* (e.g. Section 1.3 sentence 2)
- Uses short clear subheadings

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

💡Define the scope of the research.

💡Use one short paragraph for the outline of each chapter, rather than one paragraph for all chapters (e.g.Section 1.3, paragraph 1).

💡Provide a short summary paragraph at the end of the chapter which links to the next chapter.

💡Avoid using spoken expressions (e.g. ‘What’s more’, Section 1.2.3, paragraph 1, sentence 8). A better option would be ‘furthermore’.

Avoid using contractions (e.g. What’s more). They are inappropriate for academic writing.

Chapter 1

Introduction

Protein subcellular localization is one of the most essential and indispensable topics in proteomics research. Recent years have witnessed the incredibly fast development of molecular biology and computer science, which makes it possible to utilize computational methods to determine the subcellular locations of proteins. This chapter introduces the background knowledge about proteins, their subcellular locations as well as the significance of protein subcellular localization prediction.

1.1 Proteins and Their Subcellular Locations

Proteins, which are essential biological macromolecules for organisms, consist of one or more chains of amino acids residues which are encoded by genes. Proteins occur in great variety and exist in all cells and all parts of cells. Moreover, proteins exhibit tremendous diversity of biological functions and participate in virtually every process within cells. Proteins are important and indispensable in many biological processes. For example, enzymes are a special kind of proteins that participate in most of the reactions involved in metabolism catalyzing; membrane proteins are receptors for cell signalling, i.e., binding a signaling molecule and induce a biochemical response in the cell [1]; antibodies are proteins that are mainly responsible for identifying and neutralizing alien objects such as bacteria or viruses in immune systems; cell adhesion proteins, such as selectins, cadherins or integrins [2] are to bind a cell to a surface or substrate, which are essential for the pathogenesis of infectious organisms; and some proteins like digestive enzymes play important roles in chemical digestion to break down food into small molecules the body can use.

Most of the biological activities performed by proteins occur in cellular compartments, or subcellular locations. In eukaryotic cells, major subcellular locations include cytoplasm, mitochondria, chloroplast, nucleus, extracellular space, endoplasmic reticulum (ER), Golgi apparatus and plasma membrane. Cytoplasm takes up most of the cell volume, within which most cellular activities occur, such as cell division and metabolic pathways. Mitochondrion is a membrane-bound organelle found in most eukaryotic cells, which are mainly responsible for supplying energy for cellular activities. Chloroplast is an organelle existing in plant or algal cells whose role is to conduct photosynthesis to store energy from sunlight. Nucleus is a membrane-enclosed organelle which contains most of the genetic material for the cell and whose function is mainly to control the activities of the cell by regulating gene expression. Extracellular space refers to the space outside the plasma membrane which is occupied by fluid. ER is also a type of organelle that forms an interconnected membranous network of cistemae which serves the functions of folding protein molecules in cistemae and transporting synthesized proteins to Golgi apparatus. Golgi apparatus is an organelle which is particularly important in cell secretion. Plasma membrane or cell membrane is a biological membrane that separates the intracellular environment from extracellular space whose basic function is to protect the cell from its surroundings.

Some proteins locate in peroxisome, vacuole, cytoskeleton, nucleoplasm, lysosome, acrosome, cell wall, centrosome, cyanelle, endosome, hydrogenosome, melanosome, microsome, spindle pole body, synapse, etc.[1] For the virus species, viral proteins are usually located within host cells, which are distributed in subcellular locations such as host cytoplasm, host nucleus, host cell membrane, host ER, host nucleus as well as viral capsid.

1.2 Why Computationally Predicting Protein Subcellular Localization?

As an essential and indispensable topic in proteomics research and molecular cell biology, protein subcellular localization is critically important for protein function annotation, drug target discovery, and drug design [3, 4, 5]. To tackle the exponentially growing number of newly found protein sequences in the post-genomic era, computational methods are developed to assist biologists to deal with large-scale protein subcellular localization.

1.2.1 Significance of Subcellular Localization of Proteins

Proteins located in appropriate physiological contexts within a cell are of paramount importance to exert their biological functions. Subcellular localization of proteins is essential to the functions of proteins and has been suggested as a means to maximize functional diversity and economize on protein design and synthesis [6]. Aberrant protein subcellular localization is closely correlated to a broad range of human diseases, such as Alzheimer's disease [7], kidney stone [8], primary human liver tumors [9], breast cancer [10], pre-eclampsia [11] and Bartter syndrome [12]. Knowing where a protein resides within a cell can give insights on drug targets identification and drug design [13, 14].

[1]http://www.uniprot.org/locations/?query=*

1.2.2 Conventional Wet-lab Techniques

Although many proteins are synthesized in the cytoplasm, how proteins are transported to specific cellular organelles often remains unclear. Conventional wet-lab methods use genetic engineering techniques to assess subcellular locations of proteins. There are three main wet-lab techniques:

1. *Fluorescent microscopy imaging.* A useful technique is to create a fusion protein consisting of the natural protein of interest linked to a 'reporter', such as green fluorescent proteins [15]. The subcellular position of the fused protein can be clearly and efficiently visualized using microscopy [16].

2. *Immunoelectron microscopy.* This technique is regarded as a gold-standard method which uses antibodies conjugated with colloidal gold particles to make high-resolution localization of cellular compartments [17].

3. *Fluorescent tagging with biomarkers.* This technique requires the use of known compartmental markers for regions such as mitochondria, chloroplasts, plasma membrane, Golgi apparatus, ER, etc. It uses fluorescently tagged versions of these markers of antibodies to known markers to identify the localization of a protein of interest [18].

Wet-lab experiments are the gold standard for validating subcellular localization and are essential for the design of high quality localization databases such as The Human Protein Atlas.[2]

[2]http://www.proteinatlas.org/

1.2.3 Computational Prediction of Protein Subcellular Localization

Although various wet-lab experiments mentioned in Section. 1.2.2 can be used to determined the subcellular localization of a protein, solely conducting wet-lab experiments to acquire this kind of knowledge is costly, time-consuming and laborious. With the avalanche of newly discovered protein sequences in the post-genomic era, large-scale localization of proteins within cells by conventional wet-lab techniques is by no means wise and tractable. Table 1.1 shows the growth of protein sequences in the UniProt database[3] in the past decade. The UniProt database includes two databases, SwissProt whose protein sequences are reviewed and TrEMBL whose protein sequences are not reviewed. As can be seen, the number of protein sequence entries in Swiss-Prot in 2004 was only 137,916, but the figure was increased to 542,503 in 2014, which means the number of reviewed protein sequences has quadrupled in the past decade. More importantly, during this period, the number of unreviewed protein sequences has increased by almost 59 times, from 895,002 in 2004 to 52,707,211 in 2014. This suggests that the number of unreviewed protein sequences increases significantly faster than that of the reviewed ones. What's more, the ratio of the number of reviewed protein sequences and that of the unreviewed ones has been remarkably widen from 1:6 to 1:97. This suggests that the gap between the number of reviewed protein sequences and discovered but unreviewed protein sequences becomes larger and larger. Therefore, using wet-lab experiments alone to determine the subcellular localization of such a huge number of protein sequences amounts to a 'mission impossible'.

Under such circumstances, computational methods are required to assist biologists to deal with large-scale proteomic data to determine the subcellular localization of proteins.

[3]http://www.uniprot.org/

Table 1.1: Growth of protein sequences in the UniProt database. The UniProt database includes the Swiss-Prot database whose protein sequences are reviewed and the TrEMBL database whose protein sequences are unreviewed. Note that from 23-March-2010, the UniProt release has changed to the 'year_month' model.

Date	UniProt Release	No. of sequence entries		
		Swiss-Prot	TrEMBL	UniProt
02/Feb/2004	1.2	137,916	895,002	1,032,918
15/Feb/2005	4.1	166,613	1,389,215	1,555,828
07/Feb/2006	7.0	204,930	2,042,049	2,246,979
06/Feb/2007	9.6	255,667	3,078,259	3,333,926
05/Feb/2008	12.8	347,458	4,776,500	5,123,958
10/Feb/2009	14.8	408,238	6,592,465	7,000,703
09/Feb/2010	15.14	512,824	9,749,524	10,262,348
08/Feb/2011	2011_02	523,646	12,857,824	13,381,470
22/Feb/2012	2012_02	534,395	19,547,369	20,081,764
06/Feb/2013	2013_02	539,045	29,468,959	30,008,004
19/Feb/2014	2014_02	542,503	52,707,211	53,249,714

With the rapid progress of machine learning coupled with an increasing number of proteins with experimentally-determined localization, accurate prediction of protein subcellular localization by computational methods has become achievable and promising.

A protein has four distinct hierarchical structures: (1) primary structure, or the amino acid sequence; (2) secondary structure, or regularly repeating local structures, such as α-helix, β-sheet and turns; (3) tertiary structure, or the overall shape of a single protein molecule and (4) quaternary structure, or the structure formed by several protein molecules. Since the primary structure, namely the amino acid sequence is easier to obtain by the high-throughput sequencing technology, protein subcellular localization prediction usually refers to a problem of determining in which part a protein resides within a cell, given the amino acid sequence of the protein. In other words, computational methods for protein subcellular localization are equivalent to designing a model or a predictor, with the amino acid sequence information of query proteins as input and the subcellular location(s) of the protein as output.

1.3 Organization of The Thesis

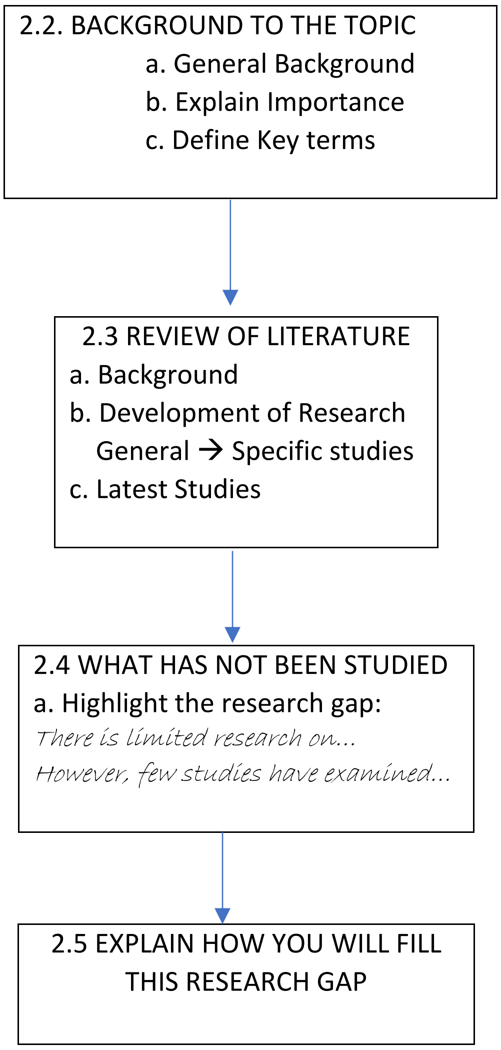
The next chapter reviews different kinds of computational methods for protein subcellular localization prediction proposed in the past decades and points out the limitations of these approaches. Chapter 3 details the legitimacy of using gene ontology (GO) information for predicting subcellular localization of proteins. Then in Chapter 4, two predictors, namely GOASVM and FusionSVM, which are both based on GO information, are proposed for single-location protein subcellular localization. Subsequently, multi-location protein subcellular localization is focused in Chapter 5. In this chapter, several multi-label predictors, including mGOASVM, AD-SVM and mPLR-Loc, which are developed based on different classifiers, are introduced for accurate prediction of subcellular localization of both single-location and multi-location proteins. Next, from the perspectives of mining deeper GO information, two more predictors, namely SS-Loc and HybridGO-Loc, are presented in Chapter 6. These predictors incorporate the information of semantic similarity over GO terms. For large-scale protein subcellular localization, Chapter 7 introduces the ensemble random projection to construct two dimension-reduced multi-label predictors, namely RP-SVM and R3P-Loc. Besides, two compact databases (ProSeq and ProSeq-GO) are proposed to replace the conventional databases (Swiss-Prot and GOA) for fast and efficient feature extraction. Chapter 8 details the specific experimental setup, including datasets construction and performance metrics. Extensive experimental results and analyses for all the proposed predictors are detailed in Chapter 9. Further discussions are provided in Chapter 10. The thesis ends with conclusions in Chapter 11.

To allow other researchers to use the proposed predictors, several online web-servers have been developed and are detailed in Appendix A. A complementary proof for no bias during the performance measurement of leave-one-out cross-validation is provided in Appendix B. The derivatives for penalized logistic regressions are provided in Appendix C.

Literature Review

The Literature Review is usually organised in the following way:

2.1 INTRODUCTION



2.2. BACKGROUND TO THE TOPIC

a. General Background
b. Explain Importance
c. Define Key terms

2.3 REVIEW OF LITERATURE

a. Background
b. Development of Research
General → Specific studies
c. Latest Studies

2.4 WHAT HAS NOT BEEN STUDIED

a. Highlight the research gap:
There is limited research on…
However, few studies have examined…

2.5 EXPLAIN HOW YOU WILL FILL THIS RESEARCH GAP

2.6 SUMMARY

Chapter 2: Literature Review

Chapter 3: Legitimacy of Using Gene Ontology Information

The Literature Review discusses previous research in the field. It should be structured in a clear way with the previous studies grouped logically. It should also be organised so as to create a coherent account or a story of existing research and the research gap identified that calls for a solution – the current study. It should include: a summary of important previous findings, a synthesis of studies that are similar, a critical discussion of all current research that is relevant to your topic, an identification of the gap in current research that needs to be addressed and an explanation of how the thesis will fill the research gap.

This Literature Review is effective partly because the writer includes the following:

Structure

(Introduction)	Not included
Review of studies using method I	Section 2.1
Review of studies using method II	Section 2.2
Limitations of methods	Section 2.3
Motivation for proposed methods	Chapter 3
(Summary)	Not included

Content

- Introduces chapter with a short introductory paragraph (e.g. paragraph 1)
- States background to the topic (paragraph 1, sentence 1-2, and sentence 5)
- States the motivation for the research (paragraph 1, sentence 4-5)
- Outlines content of the chapter (paragraph 1, sentence 7.
- Gives a short introductory paragraph to longer sections (e.g. Section 2.1)
- Explains key theoretical background (e.g. Section 2.1.1, paragraph 1)
- Cites sources, grouping studies (e.g. Section 2.1.1, paragraph 1, sentence 2)
- Critiques studies cited (e.g. Section 2.1.2, paragraph 2, final sentence)
- Develops paragraphs in a logical way, e.g. Section 2.1.2, paragraph 2:

Introduces method	sentence 1
Explains method	sentence 2
Explains development of method	sentence 3

Highlights key feature of method	sentence 4-5
Compares to other methods	sentence 6

- Develops sections in a logical way, e.g. Section 2.2:

General Method	paragraph 1
Details Specific Method	paragraph 2
Use of Specific Method	paragraph 3

- Cites important studies by name (e.g. Section 2.1.1, paragraph 2, sentence 1)
- Defines key terms (e.g. Section 2.2, paragraph 2, sentence 1)
- Details advantages and disadvantages of key methodologies reviewed (e.g. Section 2.2.1, paragraph 1, sentence 7-8)
- Discusses limitations of the methodology chosen (e.g. Section 2.2.1, paragraph 1, sentence 8, paragraph 2, sentence 6)

Language

- Uses present perfect to describe background, e.g. *has witnessed*
- Uses reporting verbs with correct meaning, e.g. *pioneered* (e.g. Section 2.1.1, paragraph 2, sentence 1), *extended* (e.g. Section 2.1.1, paragraph 2, sentence 4)
- Uses clear topic sentences (e.g. Section 2.1.2, paragraph 1, sentence 1)
- Refers reader to discussions elsewhere in the thesis (e.g. Section 2.3.2, paragraph 2, sentence 1)
- Links subsections (e.g. Section 3.1 paragraph 4)

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

💡 Include more in the Literature Review. The following chapters also include reviews of previous studies that could be included in the literature even though they focus on methodological detail. Much of the content of Chapter 3 could also appear in Chapter 2, Literature Review

💡 Sum up longer paragraphs with a final summery sentence

💡 Indicate what is lacking in previous research

Give a short summary or concluding subsection at the end of the chapter

Avoid using a spoken expression next to a formal expression e.g. *Let us* (e.g. Section 3.1, paragraph 2, sentence 3). *Obviously, it is not, fairly speaking* (e.g. Section 3.1, paragraph 2, sentence 3)

Avoid using ambiguous sub-headings (e.g. 3.3 equivalent to homologous transfer). It does not specify the focus of the question (what is being investigated to determine equivalence to homologous transfer).

Chapter 2

Literature Review

Protein subcellular localization prediction is to determine the cellular compartment(s) that a protein will be transported to. Traditionally, this problem is solved by purely experimental means through time-consuming and laborious laboratory tests [19]. However, the number of newly found protein sequences has been growing rapidly in the post-genomic era. Therefore, more reliable, efficient and automatic methods are highly required for the prediction of where a protein resides in a cell. The knowledge thus obtained can help biologists to use these newly discovered protein sequences for both basic biological research and drug design [14]. Recent decades have witnessed remarkable progress of *in-silico* methods for predicting subcellular localization of proteins, which can be roughly divided into sequence-based and knowledge-based. Both methods will be reviewed in this chapter, and then the limitations of existing methods are discussed.

2.1 Sequence-Based Methods

Sequence-based methods only use the amino acid sequence of the query protein as input. Proteins consist of linear polymers built from series of 20 different amino acids. The *ab initio* methods made efforts to find the correlations between the amino acid sequences and the subcellular locations. This kinds of methods can be generally divided into three categories described below.

2.1.1 Composition-Based Methods

Composition-based methods are one of the earliest methods for subcellular localization prediction. This category focuses on the relationship between subcellular locations and the information embedded in the amino acid sequences such as amino-acid compositions (AA) [20],[13], amino-acid pair compositions (PairAA) [20], and gapped amino-acid pair compositions (GapAA) [21] [22].

Nakashima and Nishikawa [23] pioneered the prediction of proteins by using a simple odds-ratio statistics to discriminate between soluble intracellular and extracellular proteins based on AA and PairAA information. In the AA method, each sequence can be represented by a 20-dimensional AA composition vector for subsequent classification. It was found that a simple odds-ratio statistics based on amino-acid composition and residue-pair frequencies can be used to discriminate between soluble intracellular and extracellular proteins. Later, Cedano et. al. [24] extended this method to five protein classes: integral membrane proteins, anchored membrane proteins, extracellular proteins, intracellular proteins and nuclear proteins. Reinhardt and Hubbard [25] further used the amino acid composition for subcellular localization of both prokaryotic and eukaryotic proteins. Also based on the amino acid composition, Chou and Elrod [26] designed an algorithm for predicting 12 organelles and subcellular compartments. A review [27] detailed the correlation between the amino acid compositions and subcellular locations of proteins.

To further include the sequence-order information in the sequence vectors, PairAA [20] has also been used in the prediction. This method incorporates the information of co-occurrence frequencies of adjacent dipeptide in the protein sequences for the prediction. Garg et. al. [28] designed a predictor called HSCLPred which uses both AA and PairAA as features for human protein subcellular localization. Later, Park and Kanehisa [21] used GapAA method to obtain more sequential information. This method extends PairAA to count the frequencies of amino acid pairs whose residues are separated by one or more residue positions (gaps). Chou and Cai [13] combined AA, PairAA and GapAA to construct a feature vector with several thousand dimensions for predicting protein localization in budding yeast. Later, Lee et. al. [22] did similar feature extraction for prediction of imbalanced and overlapped datasets.

Based on these early approaches, Chou [29] proposed a method called pseudo amino-acid composition (PseAA) using a sequence-order correlation factor to discover more biochemical properties, i.e., hydrophobicity, hydrophilicity and side-chain mass of amino acids from protein sequences. Some other modes, such as physicochemical distance mode [30] or amphiphilic pattern mode [31] have also been proposed to derive different types of pseudo amino acid compositions. Later, many classifiers [32, 33, 34, 35, 36, 37, 38, 39, 40, 13] based on PseAA have been proposed for protein subcellular localization. A web-server called 'PseAAC' [41] was also established which has incorporated as many as 63 different kinds of PseAA compositions.

2.1.2 Sorting Signals Based Methods

Sorting-signals based methods predict the localization by recognizing N-terminal sorting signals in amino acid sequences [42]. After a protein is synthesized, it will either be transported to an intracellular organelle or be secreted to the extracellular space through a secretory pathway [43]. The information of where the protein will be transported can be found in a short segment of the amino acid sequence, which are generally known as sorting signals, signal peptides or targeting sequences. The sorting signals of a protein will be cleaved off by a signal peptidase after it is translocated across the cell membrane. These cleavable peptides contain the information about where the protein should be transported, either to the secretory pathway (in which case they are called signal peptides) or to mitochondria and chloroplast (in which they are called transit peptides). The secretory signal peptide (SP) is an N-terminal peptide which targets a protein for translocation across the endoplasmic reticulum (ER) membrane in eukaryotes [44]. Similar to secretory SPs, the transit peptides are also N-terminal peptides [45], whose sequence motifs, however, are less conserved than those of secretory SPs. The chloroplast transit peptide (cTP), which is rich in hydroxylated residues and rarely has acidic residues, directs nuclear-encoded proteins into the chloroplast [46]. The mitochondrial targeting peptide (mTP), which rarely has negatively charged residues, directs nuclear-encoded proteins into the mitochondria [47].

Nakai and Kanehisa in 1991 [48] proposed the earliest predictor using sorting signals—PSORT, and in 2006 they extended PSORT to WoLF PSORT [49]. PSORT is a knowledge-based program for predicting protein subcellular localization, and WoLF PSORT utilizes the information contained in sorting signals, amino acid composition and functional motifs to convert amino acid sequences into numerical features. Later, methods using signal peptides, mitochondrial targeting peptides and chloroplast transit peptides have also been proposed [50, 51]. Among these predictors, TargetP [52], which uses Hidden Markov Models (HMMs) and neural networks to learn the relationship between subcellular locations and amino acid sequences, is the most popular. It uses the N-terminal sequence as input and utilizes two binary predictors, SignalP [51] and ChloroP [53]. Compared to other methods, predictors based on sorting signals are more similar to mimicking the de facto information processing in cells.

2.1.3 Homology Based Methods

Homology-based methods use the fact that homologous sequences are more likely to reside in the same subcellular location. In this group of methods, a query sequence is first used to search through a protein database for homologs [54, 55], and then the subcellular location of this query sequence is determined as the one to which the homologs belong. This kind of methods can achieve a very high accuracy as long as the homologs of the query sequences can be found in protein databases [56].

Over the years, a number of homology-based predictors have been proposed. For example, Proteome Analyst [57] computes the feature vectors for classification by using the presence or absence of some tokens from certain fields of the homologous sequences in the Swiss-Prot database. Kim et al. [58] demonstrates that feature vectors can be created by aligning an unknown protein sequence with every training sequence (with known subcellular locations). Recently, a predictor called PairProSVM was proposed by Mak et al. [59], which applies profile alignment to detect weak similarity between protein sequences. For each query sequence, a profile can be generated by PSI-BLAST [60]. Then the obtained profile is aligned with the profile of each training sequence to form a score vector, which is classified by SVMs. It was found that profile alignment is more sensitive to detecting the weak similarity between protein families than sequence alignment. Recently, Wang and Li [61] proposed a random label selection method based on the homology information for multi-label protein subcellular localization. It applied the auto covariance transformation [62] to each column of the profile of each protein to extract sequence evolutionary information.

2.2 Knowledge-Based Methods

Knowledge-based methods use information from knowledge databases, such as using Gene Ontology (GO)[1] terms [63, 40, 64, 65, 66, 67], Swiss-Prot keywords [68, 69], or PubMed abstracts [70, 71]. This kind of methods make use of the correlation between the knowledge or annotations of a protein and its subcellular localization. Recent decades have witnessed the significant improvement and enrichment of knowledge databases, or annotation databases for known proteins, which makes the prediction based on using knowledge database feasible and attractive. This kind of methods extract the feature information of the training proteins from related knowledge databases, which is then used to train a statistical model for prediction of novel proteins. Among them, GO-based methods are more attractive [64, 72, 73].

Gene Ontology (GO) is a set of standardized vocabularies that annotate the function of genes and gene products across different species. The term ‘ontology’ originally refers to a systematic account of existence. In the GO database, the annotations of gene products are organized in three related ontologies: cellular components, biological processes, and molecular functions. Cellular components refer to the substances that constitute cells and living organisms. Example substances are proteins, nucleic acids, membranes, and organelles. Majority of these substances are located within the cells, but there are also substances locating outside the cells (extracellular areas). A biological process is a sequence of events achieved by one or more ordered assemblies of molecular functions. A molecular function is achieved by activities that can be performed by individual or by assembled complexes of gene products at the molecular level.

GO-based methods make use of the well-organized biological knowledge about genes

[1]http://www.geneontology.org

and gene products in the GO databases. The GO-based methods can be viewed from the following two perspectives:

2.2.1 From the Perspective of GO-Term Extraction

From the perspectives of GO terms extraction, the GO-based predictors can be classified into three categories. The first category uses a program called InterProScan [74] to search against a set of protein signature databases to look for relevant GO terms. InterProScan is an online tool that scans a given set of protein sequences against the protein signatures of the InterPro [75, 76] member databases, including PROSITE [77], PRINTS [78], Pfam [79], ProDom [80], SMART [81], etc. InterPro can provide an integrated layer on top of the most commonly used protein signature databases, of which each InterPro entry corresponds to one GO number. Many predictors [40, 82, 63, 83] thus used InterProScan to find function-related GO terms for feature information extraction and then utilized different classifiers for predicting protein subcellular localization. Recently, Tung et. al. [84] enlarged the GO term coverage by transferring the GO terms of physically interacting partners in yeast interacting network to the target protein. The essence of this kind of methods is to transfer the GO terms of signature proteins which can be retrieved by InterProScan to the target proteins. This kind of methods can be applied to all protein sequences; but usually they can retrieve only a small number of GO terms, which may not be sufficient for accurate prediction of subcellular localization.

The second method uses the accession numbers (ACs) of proteins to search against the Gene Ontology Annotation (GOA) database[2] to retrieve GO terms. This method directly associates the protein accession numbers with GO entries, which correspond to some predefined biological processes, molecular functions or cellular components. Typical

[2]http://www.ebi.ac.uk/GOA

predictors using this approach include Euk-OET-PLoc [64], Hum-PLoc [85], Euk-mPLoc [86] and Gneg-PLoc [38]. Euk-OET-PLoc, proposed by Chou et al. [64], demonstrates that this category can achieve a higher performance than any other existing methods. In [85], a sequence is mapped into the GO database so that a feature vector can be formed by determining which GO terms the sequence holds. These predictors perform better than the ones based on InterProScan, but they are not applicable to proteins that have not been functionally annotated.

The third method uses BLAST [60] to obtain the ACs of homologs of the query proteins and then uses these ACs to search against the GOA database. With the input of query protein sequence, BLAST search can produce the ACs of the homologous proteins to the target protein by sequence alignment, which can be used as keys to retrieve GO terms in the GOA database. This enables the extension of GO-based methods to prediction of novel or newly discovered proteins. Typical predictors include ProLoc-GO [72], iLoc-Plant [87], iLoc-Hum [88], iLoc-Gpos [89], iLoc-Euk [90], iLoc-Gneg [91], Cell-PLoc [92], iLoc-Virus [93] and Cell-PLoc 2.0 [94]. ProLoc-GO [72] uses a searching algorithm called GOmining to discover the informative GO terms and classify them into instructive GO terms and essential GO terms to leverage the information in the GO database. This method is applicable to all protein sequences and is able to retrieve more GO terms, which are essential for good prediction performance.

2.2.2 From the Perspective of GO-Vector Construction

After retrieving the GO terms, the ways of constructing the GO feature vectors are also of high significance. From the perspectives of GO-Vector Construction, GO-based methods can be classified into two categories.

The first category considers each GO term as a canonical basis of a Euclidean space where the coordinates can be equal to either 0 or 1. In other words, this method uses the presence or absence of some predefined GO terms as the information of feature vectors. Representative methods in this category include Euk-OET-PLoc [64], Hum-PLoc [85], Gneg-PLoc [38] and Gpos-PLoc [95]. Recently, a modified binary feature vector construction method was proposed to deal with many sets of GO terms for one protein [94, 96]. Specifically, instead of using 1-0 value, each element of the feature vectors is represented by the percentage of homologous proteins containing the corresponding GO term. This category of methods provides a large coverage of GO terms, but many of them may be irrelevant to the classification task. Besides, this category of methods ignores the fact that a GO term may be used to annotate the same protein multiple times under different entries in the GOA database.

The second category uses genetic algorithms to select the most informative GO terms, such as ProLoc-GO [72] and PGAC [97]. This category of methods selects some informative GO terms which are the essential GO terms annotating subcellular compartments. One problem of this type of methods is that it may select only a small number of GO terms, increasing the chance of having a null GO vector for a test protein.

Lei and Dai [98] predicted protein subnuclear localization by using the semantic similarity between GO terms. The similarity of two GO terms is defined by the depth of the most recent common ancestor GO terms. This information is definitely insufficient for accurate prediction, which is proved by the poor performance in [98]. Lee et. al. [99] proposed using only GO terms in molecular functions and biological processes combined with protein-protein interactions and sequence-based features, such as amino acid compositions and biochemical properties, for the prediction. This sophisticated strategy improves the performance, but it is difficult to guarantee that all of the features can be obtained for every protein, which may limit the application of this method.

2.3 Limitations of Existing Methods

2.3.1 Limitations of Sequence-based Methods

Among the sequence-based methods mentioned above, composition-based methods are easy to implement and have obvious biological reasoning; but in most cases these methods perform poorly, which demonstrates that amino acid sequence information is not sufficient for protein subcellular localization.

Sorting-signal based methods can determine the subcellular locations of proteins from the sequence segments containing the localization information, leading these methods to be more biologically plausible and robust. However, this type of methods could only deal with proteins that contain signal sequences. For example, the popular TargetP [52, 100] could only detect three locations: chloroplast, mitochondria and secretory pathway (extracellular).

Homology-based methods, on the other hand, can theoretically detect as many locations as appeared in the training data and can achieve comparatively high accuracy [101]. But when the training data contains sequences with low sequence similarity or the numbers of samples in different classes are imbalanced, the performance is still very poor. While the functional-domain based methods can often outperform sequence-based methods (as they can leverage the annotations in functional domain databases), they can only be applied to datasets where the sequences possess the required information as so far not all sequences are functionally annotated. Thus, they must be complemented by other types of methods.

2.3.2 Limitations of Knowledge-based Methods

Compared to sequence-based methods, GO-based methods are found to be superior [64, 72, 73, 102]. However, existing GO-based methods are not without disadvantages and limitations.

From the perspectives of GO terms extraction, as mentioned in Section 2.2.1, methods using InterProScan to retrieve GO terms can only produce a limited number of GO terms, which are not sufficient for accurate prediction. In some cases, it is possible that no GO terms can be retrieved for some proteins from InterProScan because these proteins have not been functionally annotated.

The second category directly associates the protein accession numbers with GO entries in the GOA database, which can possibly generate many useful GO terms for classification. However, this is not applicable to newly discovered proteins, which have not been annotated, let alone be assigned with accession numbers.

The third category uses BLAST to transfer the GO information from homologs of the target proteins, which can both produce many GO terms and also be applicable to novel proteins. However, directly using the top homologs of the query proteins to retrieve GO terms cannot guarantee the availability of GO information because it is also possible that no corresponding GO terms can be found for their top homologs in the GOA database.

From the perspectives of GO vector construction, as mentioned in Section 2.2.2, the first category of methods, namely 1-0 value method, is simple and logically plausible, but some information will be inevitably lost because it quantizes the frequency of occurrences of GO terms to either 0 or 1. The second category of methods constructs the feature vectors based on essential GO terms, which may be directly associated with subcellular localization. However, it is also liable to retrieving insufficient GO terms for accurate prediction.

In summary, existing methods have advantages and disadvantages. In the following chapters, we will propose our predictors which make full use of state-of-the-art feature information, i.e. GO information, and present new approaches to retrieving GO terms and constructing GO vectors.

Chapter 3

Legitimacy of Using Gene Ontology Information

Before we propose subcellular-location predictors based on Gene Ontology (GO) information, in this chapter we will address some concerns about the legitimacy of using GO information for protein subcellular localization. There are mainly three kinds of concerns about using GO information: (1) Can the GO-based methods be replaced by a lookup table using the cellular component GO terms as the keys and the component categories as the hashed values? (2) Are cellular components GO terms the only information necessary for protein subcellular localization? (3) Are GO-based methods equivalent to transferring annotations from BLAST homologs? These concerns are explicitly addressed in the following sections.

3.1 Direct Table Lookup?

For those who are skeptical about the GO-based prediction methods, the following question is prone to be raised: If a protein has already been annotated by cellular component GO terms, is it still necessary to predict its subcellular localization? The GO comprises three orthogonal categories whose terms describe the cellular components, biological processes, and molecular functions of gene products. This sounds like a legitimate question because the GO terms already suggest the subcellular localization and therefore it is merely a procedure of converting the annotation into another format. In other words, all we need is to create a lookup table (hash table) using the cellular component GO terms as the keys and the component categories as the hashed values.

To answer this question, let us provide some facts here. Most of the existing 'non-GO predictors' were established based on the proteins in the Swiss-Prot database in which the subcellular locations are experimentally determined. Is it logical to consider that all of these methods have nothing to predict? Obviously, it is not. Fairly speaking, as long as the input is a query protein sequence and the output is its subcellular location(s), the predictor is deemed to be a valid protein subcellular-location predictor. In fact, most of the existing GO predictors, such as iLoc-Euk [90] and iLoc-Hum [88], use protein sequence information only to predict the subcellular locations, without adding any GO information to the input. That is to say, these GO predictors use the same input as the non-GO predictors. Therefore, GO-based predictors should also be regarded as valid predictors.

Here, we explain why the simple table-lookup method mentioned above is undesirable. Although the cellular component ontology is directly related to the subcellular localization, we cannot simply use its GO terms to determine the subcellular locations of proteins. The reason is that some proteins do not have cellular component GO terms. Even for proteins annotated with cellular-component GO terms, it is inappropriate to use these terms only to determine their subcellular localizations. The reason is that a protein could have multiple cellular-component GO terms that map to different subcellular localizations, which are highly likely to be inconsistent with the true subcellular locations of proteins. Another reason is that, according to [85], proteins with annotated subcellular localization in Swiss-Prot may still be marked as 'Cellular Component Unknown' in the GO database. Because of this limitation, it is necessary to use the other two ontologies as well because they are also relevant (although not directly) to the subcellular localization of proteins.

To further exemplify the analysis above, we created lookup tables for protein subcellular prediction of both single-label case and multi-label case, respectively, which are specified in the following subsections.

3.1.1 Table-Lookup Procedure for Single-Label Prediction

To exemplify the discussion above for the single-location case, we created a lookup table (Table 3.1) and developed a table-lookup procedure to predict the subcellular localization of the proteins in the EU16 dataset (Table 8.1). Table 3.1 has two types of GO terms: essential GO terms and child GO terms. As the name implies, the essential GO terms, as identified by Huang et al. [72], are GO terms that are essential or critical for the subcellular localization prediction. In addition to the essential GO terms, their direct descendants (known as child terms) also possess direct localization information. The relationships between child terms and their parent terms include 'is a', 'part of' and 'occurs in' [103]. The former two correspond to cellular component GO terms and the third one typically corresponds to biological process GO terms. As we are more interested in cellular component GO terms, the 'occurs in' relationship will not be considered. For ease of reference, we refer to both essential GO terms and their child terms as 'explicit GO terms'.

For each class in Table 3.1, the child terms were obtained by presenting the corresponding essential GO term to the QuickGO server [104], followed by excluding those child terms that do not appear in the proteins of the EU16 dataset. Note that if we use

Table 3.1: Explicit GO terms for the EU16 dataset. Explicit GO terms include essential GO terms and their child terms that appear in the proteins of the dataset. The definition of essential GO terms can be found in [72]. Here the relationship only includes 'is a' and 'part of', because only cellular component GO terms are analyzed here. *CC*: cellular components, including cell wall (CEL), centriole (CEN), chloroplast (CHL), cyanelle (CYA), cytoplasm (CYT), cytoskeleton (CYK), endoplasmic reticulum (ER), extracellular (EXT), Golgi apparatus (GOL), lysosome (LYS), mitochondrion (MIT), nucleus (NUC), peroxisome (PER), plasma membrane (PM), plastid (PLA) and vacuole (VAC); *Relationship*: the relationship between child terms and their parent essential GO terms; *No. of Terms*: the total number of explicit GO terms in a particular class.

Class	CC	Explicit GO Terms		No. of Terms
		Essential Terms	Child Terms (Relationship)	
1	CEL	GO:0005618	GO:0009274 (Is a), GO:0009277 (Is a), GO:0009505 (Is a), GO:0031160 (Is a)	5
2	CEN	GO:0005814	None	1
3	CHL	GO:0009507	None	1
4	CYA	GO:0009842	GO:0034060 (Part of)	2
5	CYT	GO:0005737	GO:0016528 (Is a), GO:0044444 (Part of)	3
6	CYK	GO:0005856	GO:0001533 (Is a), GO:0030863 (Is a), GO:0015629 (Is a), GO:0015630 (Is a), GO:0045111 (Is a), GO:0044430 (Part of)	7
7	ER	GO:0005783	GO:0005791 (Is a), GO:0044432 (Part of)	3
8	EXT	GO:0030198	None	1
9	GOL	GO:0005794	None	1
10	LYS	GO:0005764	GO:0042629 (Is a), GO:0005765 (Part of), GO:0043202 (Part of)	4
11	MIT	GO:0005739	None	1
12	NUC	GO:0005634	GO:0043073 (Is a), GO:0045120 (Is a), GO:0044428 (Part of)	4
13	PER	GO:0005777	GO:0020015 (Is a), GO:0009514 (Is a)	3
14	PM	GO:0005886	GO:0042383 (Is a), GO:0044459 (Part of)	3
15	PLA	GO:0009536	GO:0009501 (Is a), GO:0009507 (Is a), GO:0009509 (Is a), GO:0009513 (Is a), GO:0009842 (Is a)	6
16	VAC	GO:0005773	GO:0000322 (Is a), GO:0000323 (Is a), GO:0005776 (Is a)	4

the cellular-component names as the searching keys, QuickGO will give us more than 49 cellular-component GO terms, suggesting that the 49 explicit GO terms are only a tiny subset of all relevant GO terms (in our method, we have more than 5000 relevant GO terms). Even for such a small number of explicit GO terms, many proteins have explicit GO terms spanning several classes.

Given a query sequence, we first obtain its 'GO-term' set from the GO annotation database. Then, if only one of the terms in this set matches an essential GO term in Table 3.1, the subcellular location of this query protein is predicted to be the one corresponding to this matched GO term. For example, if the set of GO terms contains GO:0005618, then this query protein is predicted as 'Cell Wall'. Further, if none of the terms in this set matches any essential GO terms but one of the terms in this set matches any child terms in Table 3.1, then the query protein is predicted as belonging to the class associated with this child GO term. For example, if no essential GO terms can be found in the set but GO:0009274 is found, then the query protein is predicted as 'Cell Wall'.

3.1.2 Table-Lookup Procedure for Multi-Label Prediction

To exemplify the above discussion for the multi-location case, we created a lookup table (Table 3.2) and developed a table-lookup procedure to predict the subcellular localization of the proteins in the virus dataset (see Table 8.5(a)). Similar to the single-location case, Table 3.2 has two types of GO terms: essential GO terms and child GO terms. As the name implies, the essential GO terms [72] are GO terms that are essential or critical for the subcellular localization prediction. In addition to the essential GO terms, their direct descendants (known as child terms) also possess direct localization information. The relationships between child terms and their parent terms include 'is a', 'part of' and 'occurs in' [103]. The former two correspond to cellular component GO terms and the

Table 3.2: Explicit GO terms for the virus dataset. Explicit GO terms include essential GO terms and their child terms. The definition of essential GO terms can be found in [72]. Here the relationship includes 'is a' and 'part of' only, because only cellular component GO terms are analyzed here. *CC*: cellular components, including viral capsid (VC), host cell membrane (HCM), host endoplasmic reticulum (HER), host cytoplasm (HCYT), host nucleus (HNUC) and secreted (SEC); *Relationship*: the relationship between child terms and their parent essential GO terms; *No.*: the total number of explicit GO terms in a particular class.

Class	CC	Explicit GO Terms		No.
		Essential Terms	Child Terms (Relationship)	
1	VC	GO:0019028	GO:00046727 (Part of), GO:0046798 (Part of), GO:0046806 (Part of), GO:0019013 (Part of), GO:0019029 (Is a), GO:0019030 (Is a)	7
2	HCM	GO:0033644	GO:0044155 (Part of), GO:0044084 (Part of), GO:0044385 (Part of), GO:0044160 (Is a), GO:0044162 (Is a), GO:0085037 (Is a), GO:0085042 (Is a), GO:0085039 (Is a), GO:0020002 (Is a), GO:0044167 (Is a), GO:0044173 (Is a), GO:0044175 (Is a), GO:0044178 (Is a), GO:0044384 (Is a), GO:0033645 (Is a), GO:0044231 (Is a), GO:0044188 (Is a), GO:0044191 (Is a), GO:0044200 (Is a)	20
3	HER	GO:0044165	GO:0044166 (Part of), GO:0044167 (Part of), GO:0044168 (Is a), GO:0044170 (Is a)	5
4	HCYT	GO:0030430	GO:0033655 (Part of)	2
5	HNUC	GO:0042025	GO:0044094 (Part of)	2
6	SEC	GO:0005576	GO:0048046 (Is a), GO:0044421 (Part of)	3

third one typically corresponds to biological process GO terms. As we are more interested in cellular component GO terms, the 'occurs in' relationship will not be considered. For ease of reference, we refer to both essential GO terms and their child terms as 'explicit GO terms'.

For each class in Table 3.2, the child terms were obtained by presenting the corresponding essential GO term to the QuickGO server [104]. In our method, we have more than 300 relevant GO terms for the virus dataset. Even for such a small number of explicit GO terms, many proteins have explicit GO terms spanning several classes.

Given a query sequence, we first obtain its 'GO-term' set from the GO annotation database. Then, if one (or more than one) of the terms in this set matches an essential GO term in Table 3.2, the subcellular location set of this query protein is predicted to be the one (or the ones) corresponding to the matched GO term(s). For example, if the set of GO terms contains GO:0019028, then this query protein is predicted as 'Viral capsid'; or if the set of GO terms contains both GO:0030430 and GO:0042025, then this query protein is predicted as 'host cytoplasm' and 'host nucleus'. Further, if none of the terms in this set matches any essential GO terms but one (or more than one) of the terms in this set match(es) any child terms in Table 3.2, then the query protein is predicted as belonging to the class(es) associated with the child GO term(s). For example, if no essential GO terms can be found in the set but GO:0019030 is found, then the query protein is predicted as 'Viral capsid'; or if GO:0044155, GO:0044166 and GO:0033655 are found, then the query protein is predicted as 'host cell membrane', 'host endoplasmic reticulum' and 'host cytoplasm'.

3.1.3 Problems of Table Lookup

A major problem of this table lookup procedure is that the GO terms of a query protein may contain many essential GO terms and/or having child terms spanning across more classes than the number of true subcellular locations, causing over-prediction or inconsistent classification decisions. For example, in the EU16 single-location dataset, 713 (out of 2423) proteins have explicit GO terms that map to more than one class, and 513 (out of 2423) proteins do not have any explicit GO terms. This means that about 51% (1226/2423) of the proteins in the dataset cannot be predicted using only explicit GO terms. Among the 2423 proteins in the dataset, only 1197 (49%) of them have explicit GO terms that map to unique (consistent) subcellular locations. While in the multi-label virus dataset, 69, 14 and 3 (out of 207) proteins have explicit GO terms that map to two, three and four locations, and 121 (out of 207) proteins have explicit GO terms that map to one location. By comparing with the true locations, there are totally 139 proteins whose explicit GO terms are consistent with their true locations, of which there are 107 single-label proteins, 30 two-label proteins and 2 three-label proteins. This means that only about 67% (139/207) proteins are likely to be predicted correctly.

Note that this table-up procedure only incorporates the explicit GO terms. If more cellular-component GO terms and even GO terms from the other two ontologies are used to infer the subcellular locations, more proteins are likely to be over-predicted. This analysis suggests that direct table lookup is not a desirable approach and this motivates us to develop machine learning methods for GO-based subcellular localization prediction.

3.2 Only Using Cellular Component GO Terms?

Some people disprove the effectiveness of GO-based methods by claiming that only cellular component GO terms are necessary and GO terms in the other two categories play no role in determining the subcellular localization. They argue that cellular component GO terms directly associate with the predicting labels, and only these GO terms are useful for determination of protein subcellular localization.

This concern has been explicitly and directly addressed by Lu and Hunter [105], who demonstrated that GO molecular function terms are also predictive of subcellular localization, particularly for nucleus, extracellular space, membrane, mitochondrion, endoplasmic reticulum and Golgi apparatus. The in-depth analysis of the correlation between the molecular function GO terms and localization provide an explanation of why GO-based methods outperform sequence-based methods. Mei et al. [83] also did extensive experiments on Multiloc [106] dataset, BaCelLo [107] dataset and Euk-mPLoc [86] dataset to show that not only cellular component GO terms play significantly roles in estimating the kernel weights of the proposed classifier and training the prediction model, but also GO terms in categories of molecular functions and biological processes make considerable contributions on final predictions. This is also understandable because although GO terms in molecular functions and biological processes have no direct implications of protein subcellular localization, proteins can only properly exert their functions in particular physiological contexts and participate in certain biological processes within amenable cellular compartments. Therefore, it is logically acceptable that all categories of GO terms should be considered for accurate prediction of protein subcellular localization.

3.3 Equivalent to Homologous Transfer?

Even though GO-based methods can predict novel proteins based on the GO information obtained from their homologous proteins [108, 102], some researchers still argue that the prediction is equivalent to using the annotated localization of the homologs (i.e., using BLAST [60] with homologous transfer). They argue that GO-based methods are in fact equivalent to mining the annotations of homologous proteins retrieved by BLAST, whose subcellular localization information are well annotated or experimentally determined. Namely, they consider GO-based methods have nothing to do with machine learning; instead they think these methods simply assign the subcellular localization information of the homologs to the target proteins.

This claim is clearly proved to be untenable in Table 9.4 of Chapter 9, which demonstrates that GO-based methods remarkably outperform methods that only use BLAST and homologous transfer. More details of the procedures can be found in Section 9.1.4 of

Chapter 9. Besides, Briesemeister et al. [109] also suggested that using BLAST alone is not sufficient for reliable prediction.

3.4 More Reasons for Using GO Information

As suggested by Chou [110], as long as the input of query proteins for predictors is the sequence information without any GO annotation information and the output is the subcellular localization information, there is no difference between non-GO based methods and GO-based methods, which should be regarded as equally legitimate for subcellular localization.

Some other papers [111, 92] also provide strong arguments supporting the legitimacy of using GO information for subcellular localization. In particular, as suggested by [92], the good performance of GO-based methods is due to the fact that the feature vectors in the GO space can better reflect their subcellular locations than those in the Euclidean space or any other simple geometric space.

Chapter 4: Single-Location Protein Subcellular Localization

Chapter 5: From Single-Location to Multi-Location

The Methodology chapter is a description of the methodological approach(es) taken and an explanation of why they were chosen. It often includes: a general introduction restating the central aims, the details of any part of the methodology that may be unfamiliar to the readers, an explanation of how the results will be analyzed, and an explanation of any limitations with the methodology.

Chapter4 and 5 describe a number of different methodologies for two the two research foci for protein subcellular localization (single location and multi-location subcellular localization).

Chapter 4 is organised in the following way:

Structure

(Introduction)	Not included
Describes one prediction process	Section 4.1
Details the second process	Section 4.2
Summarizes chapter and compares methods	Section 4.3

Chapter 5 is organized in the following way:

Structure

(Introduction)	Not included
Describes key background including existing methods	Section 5.1-5.2
Details three proposed methods	Section 5.3-5.5
Summarizes chapter and compares models	Section 5.6

Chapter 4 and 5 are effective partly because the writer includes the following:

Content

- Gives clear introductory paragraphs (e.g. Chapter 4 and 5)
- Explains theoretical basis for the development of the model (e.g. Chapter 4, paragraph 1)
- Outlines the content of the chapter (e.g. Chapter 4, paragraph 2)
- Uses a short introductory paragraph to subsections (e.g. Section 4.1, paragraph 1)
- Explains key terminology, e.g. *are referred to as* (e.g. Section 5.2 paragraph 1, sentence 4)
- Critiques other methodologies (e.g. Section 5.2.1, paragraph 3, final sentence)
- Develops sections logically, e.g. Section 4.1.2:

Method	paragraph 1
Drawback with method	paragraph 2 sentence 1-2
Possible solution	paragraph 2 sentence 3-5
Evaluation	paragraph 2 sentence 5

- Defines key terms (e.g. Section 5.4.3, paragraph 1, sentence 2)
- Provides a summary subsection (e.g. Section 4.3, 5.6)

Language

- Uses present simple tense and passive voice to describe processes e.g. *are linked, is preserved* (e.g. Section 5.2.2, paragraph 4, sentence 2 and 3)
- Links subsections, e.g. *which is elaborated below* (e.g. Section 5.3.1, final sentence)
- Uses tentative language when predicting e.g. *better performance may be obtained* (e.g. Section 4.2.1.1, paragraph 1, sentence 1)

To consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

Be consistent in the use of *we*. If you use *we* in a description of methodology and results, it should be used constantly. The passive voice and past tense is preferred by a number of academics, e.g. *was used* rather than *we used*.

Explain where a figure can be found if it is not on the same page as it is referred to in the text, e.g. *Fig. 4.5 illustrates* (e.g. Section 4.2.3, paragraph 1, sentence 1) would be better written as, *Fig 4.5 on page 66 illustrates.*

- Avoid using too many subsections. Subsections are to help the reader understand, having too many numerals can confuse the reader, e.g. *4.2.1.1.*
- End the summary with a transitional sentence linking the chapter to the following chapter, e.g. *This problem is discussed in the following chapter*.
- Avoid contractions and short form, e.g. *Let's* (e.g. Section 5.4.3 paragraph 1, sentence 1).

Chapter 4

Single-Location Protein Subcellular Localization

According to a recent comprehensive review [112], the establishment of a statistical protein predictor involves the following five steps: (i) construction of a valid dataset for training and testing the predictor; (ii) formulation of effective mathematical expressions for converting proteins' characteristics to feature vectors that are relevant to the prediction task; (iii) development of classification algorithm for discriminating the feature vectors; (iv) evaluation of cross-validation tests for measuring the performance of the predictor; and (v) deployment of a user-friendly, publicly accessible web-server for other researchers to use and validate the prediction method. These steps are further elaborated in the following chapters.

This chapter will focus on predicting single-location protein subcellular localization. Single-location proteins refer to those proteins that are located in one subcellular compartment. It is well known that most proteins stay only at one subcellular location [113]. Therefore, predicting the subcellular localization of single-label proteins is of great significance. In this chapter, two GO-based predictors will be presented, namely GOASVM and FusionSVM.

4.1 GOASVM: Extracting GO from Gene Ontology Annotation Database

The GOASVM predictor uses either accession numbers (ACs) or amino acid (AA) sequences as input. The prediction process is divided into two stages: feature extraction (vectorization) and pattern classification. For the former, the query proteins are "vectorized" to high-dim GO vectors. For the latter, the GO vectors are classified by one-vs-rest linear support vector machines (SVMs).

4.1.1 Gene Ontology Annotation Database

GOASVM extracts the GO information from the Gene Ontology annotation (GOA) database.[1] The database uses standardized GO vocabularies to systematically annotate non-redundant proteins of many species in the UniProt Knowledgebase (UniProtKB) [115], which comprises Swiss-Prot [116], TrEMBL [116] and PIR-PSD [117]. The large-scale assignment of GO terms to UniProtKB entries (or ACs) was done by converting a portion of the existing knowledge held within the UniProKB database into GO terms [114]. The GOA database also includes a series of cross-references to other databases. For example, the majority of UniProtKB entries contain cross-references to InterPro identification numbers in the InterPro database maintained by the European Bioinformatics Institute (EBI) [118]. The GO-term assignments are released monthly, in accordance with a format standardized by the GO Consortium. As a result of the Gene Ontology (GO)[2] Consortium annotation effort, the GOA database has become a large and comprehensive resource for proteomics research [114].

Because the proteins in the GOA database have been systematically annotated by GO

[1]http://www.ebi.ac.uk/GOA
[2]http://www.geneontology.org

terms, it is possible to exploit the relationship between the accession numbers of proteins and GO terms for subcellular localization. Specifically, given the accession number of a protein, a set of GO terms can be retrieved from the GOA database file.[3] In UniProKB, each protein has a unique accession number (AC), and in the GOA database, each AC may be associated with zero, one or more GO terms. Conversely, one GO term may be associated with zero, one, or many different ACs. This means that the mappings between ACs and GO terms are many-to-many.

4.1.2 Retrieval of GO Terms

Given a query protein, GOASVM can handle two possible cases: (1) the AC is known and (2) the AA sequence is known. For proteins with known ACs, their respective GO terms are retrieved from the GOA database using the ACs as the searching keys. For a protein without an AC, its AA sequence is presented to BLAST [60] to find its homologs, whose ACs are then used as keys to search against the GOA database.

While the GOA database allows us to associate the AC of a protein with a set of GO terms, for some novel proteins, neither their ACs nor the ACs of their top homologs have any entries in the GOA database; in other words, the GO vectors constructed in Section 4.1.3 will contain all-zero, which are meaningless for further classification. In such case, the ACs of the homologous proteins, as returned from BLAST search, will be successively used to search against the GOA database until a match is found. Specifically, for the proteins whose top homologs do not have any GO terms in the GOA database, we used the second-top homolog to find the GO terms; similarly, for the proteins whose top and 2-nd homologs do not have any GO terms, the third-top homolog was used; and so on until all the query proteins can correspond to at least one GO term. With the rapid

[3]ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/UNIPROT/

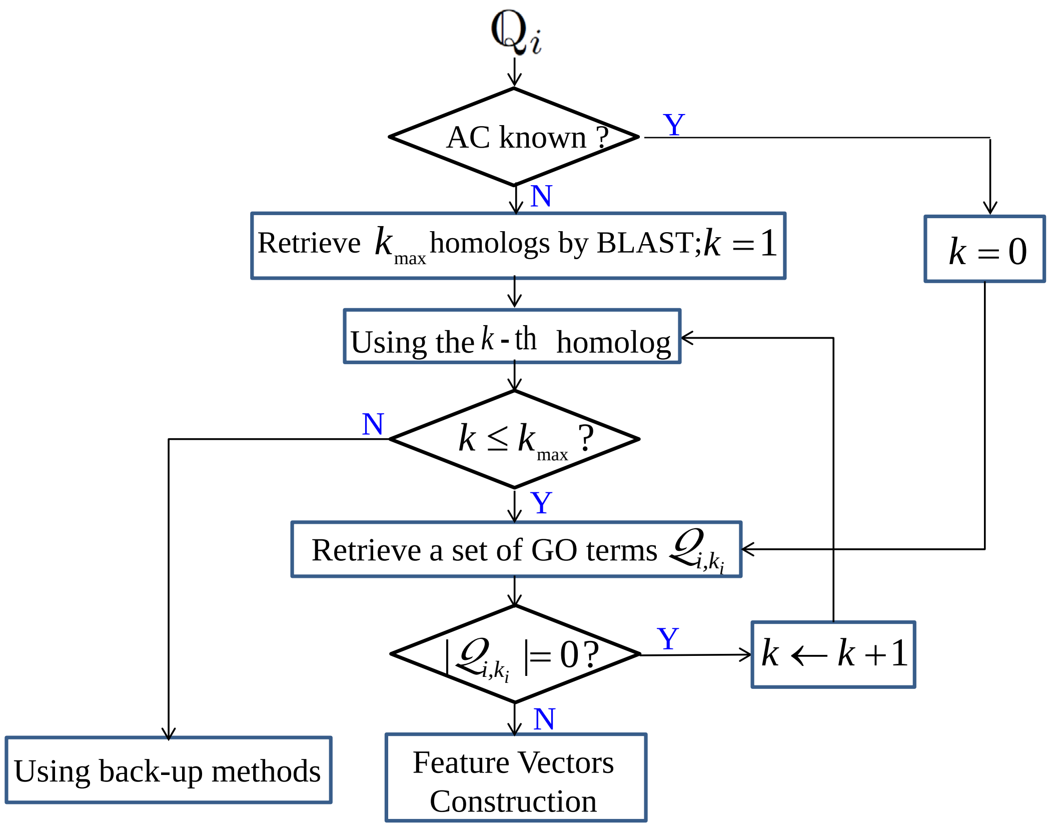

$\mathbb{Q}_i$
AC known ?
Y
N
Retrieve $k_{\max}$ homologs by BLAST; $k=1$
$k=0$
Using the k-th homolog
N
$k \le k_{\max}$?
Y
Retrieve a set of GO terms $\mathcal{Q}_{i,k_i}$
$|\mathcal{Q}_{i,k_i}| = 0$?
Y
$k \leftarrow k+1$
N
Using back-up methods
Feature Vectors Construction

Figure 4.1: **Procedures of retrieving GO terms**. $\mathbb{Q}_i$: the i-th query protein; $k_{\max}$: the maximum number of homologs retrieved by BLAST with the default parameter setting; $\mathcal{Q}_{i,k_i}$: the set of GO terms retrieved by BLAST using the k_i-th homolog for the i-th query protein $\mathbb{Q}_i$; k_i: the k_i-th homolog used to retrieve the GO terms.

progress of the GOA database [119], it is reasonable to assume that the homologs of the query proteins have at least one GO term [66]. Thus, it is not necessary to use back-up methods to handle the situation where no GO terms can be found. The procedures are outlined in Fig. 4.1.

4.1.3 Construction of GO Vectors

According to Eq. 6 of [112], the characteristics of any proteins can be represented by the general form of Chou's pseudo amino acid composition [31, 29]:

$$\mathbf{q}_i = [\phi_{i,1}, \ldots, \phi_{i,u}, \ldots, \phi_{i,W}]^{\mathsf{T}}, \tag{4.1}$$

where T is a transpose operator, W is the dimension of the feature vector $\mathbf{q}_i$, and the definitions of the W feature components $\phi_{i,u}$ $(u = 1, \ldots, W)$ depend on the feature extraction approaches elaborated below.

Given a dataset, we used the procedure described in Section 4.1.2 to retrieve the GO terms of all of its proteins. Let $\mathbb{W}$ denotes a set of distinct GO terms corresponding to a data set. $\mathbb{W}$ is constructed in two steps: (1) identifying all of the GO terms in the dataset and (2) removing the repetitive GO terms. Suppose W distinct GO terms are found, i.e., $|\mathbb{W}| = W$; these GO terms form a GO Euclidean space with W dimensions. For each sequence in the dataset, a GO vector is constructed by matching its GO terms against $\mathbb{W}$, using the number of occurrences of individual GO terms in $\mathbb{W}$ as the coordinates. We have investigated four approaches to determining the elements of the GO vectors.[4]

1. **1-0 value.** In this approach, each of the W GO terms represents one canonical basis of a Euclidean space, and a protein is represented by a point in this space with coordinates equal to either 0 or 1. Specifically, the GO vector of the i-th protein $\mathbb{Q}_i$ is denoted as:

$$\mathbf{q}_i = \begin{bmatrix} a_{i,1} \\ \vdots \\ a_{i,u} \\ \vdots \\ a_{i,W} \end{bmatrix} \text{ where } a_{i,u} = \begin{cases} 1 & , \text{GO hit} \\ 0 & , \text{otherwise} \end{cases} \tag{4.2}$$

[4]Note that these four types of features are used independently, rather than being combined for classification. Through performance evaluation and experimentation, we would like to find out which type of the four features performs the best.

where 'GO hit' means that the u-th GO term appears in the GOA-search result using the AC of the i-th protein as the searching key.

2. **Term-Frequency (TF).** This approach is similar to the 1-0 value approach in that a protein is represented by a point in the W-dim Euclidean space. However, unlike the 1-0 approach, it uses the number of occurrences of individual GO terms as the coordinates. Specifically, the GO vector $\mathbf{q}_i$ of the i-th protein is defined as:

$$\mathbf{q}_i = \begin{bmatrix} b_{i,1} \\ \vdots \\ b_{i,u} \\ \vdots \\ b_{i,W} \end{bmatrix} \text{ where } b_{i,u} = \begin{cases} f_{i,u} & \text{, GO hit} \\ 0 & \text{, otherwise} \end{cases} \tag{4.3}$$

where $f_{i,u}$ is the number of occurrences of the u-th GO term (term-frequency) in the i-th protein. The rationale is that the term-frequencies may also contain important information for classification and therefore should not be quantized to either 0 or 1. Note that $b_{i,u}$'s are analogous to the term-frequencies commonly used in document retrieval.

3. **Inverse Sequence-Frequency (ISF).** In this approach, a protein is represented by a point with coordinates determined by the existence of GO terms and the inverse sequence-frequency (ISF). Specifically, the GO vector $\mathbf{q}_i$ of the i-th protein is defined as:

$$\mathbf{q}_i = \begin{bmatrix} c_{i,1} \\ \vdots \\ c_{i,u} \\ \vdots \\ c_{i,W} \end{bmatrix}, \; c_{i,u} = a_{i,u} \log\left(\frac{N}{|\{k : a_{k,u} \neq 0\}|}\right) \tag{4.4}$$

where N is the number of protein sequences in the training dataset. The denominator inside the logarithm is the number of GO vectors (among all GO vectors in the dataset) having a non-zero entry in their u-th element, or equivalently the number of sequences with the u-th GO term as determined in Section 4.1.2. Note that the logarithmic term in Eq. 4.4 is analogous to the inverse document frequency commonly used in document retrieval. The idea is to emphasize (resp. suppress) the GO terms that have a low (resp. high) frequency of occurrences in the protein sequences. The reason is that if a GO term occurs in every sequence, it is not very useful for classification.

4. **Term-Frequency–Inverse Sequence-Frequency (TF-ISF).** This approach combines term-frequency (TF) and inverse sequence frequency (ISF) mentioned above. Specifically, the GO vector $\mathbf{q}_i$ of the i-th protein is defined as:

$$\mathbf{q}_i = \begin{bmatrix} d_{i,1} \\ \vdots \\ d_{i,u} \\ \vdots \\ d_{i,W} \end{bmatrix}, \; d_{i,u} = b_{i,u} \log \left(\frac{N}{|\{k : b_{k,u} \neq 0\}|} \right) \tag{4.5}$$

where $b_{i,u}$ is defined in Eq. 4.3.

By correlating Eqs. 4.2–4.5 with the general form of pseudo amino acid composition (Eq. 4.1), we notice that W is the number of distinct GO terms of the given dataset, and $\phi_{i,u}$'s in Eq. 4.1 correspond to $a_{i,u}$, $b_{i,u}$, $c_{i,u}$ and $d_{i,u}$ in Eqs. 4.2–4.5, respectively.

4.1.4 Multi-class SVM Classification

Support Vector Machines (SVMs) were originally proposed by Vapnik [120] to tackle binary classification problems. An SVM classifier maps a set of input patterns into a high-dimensional space and then finds the optimal separating hyperplane and the margin of separations in that space. The obtained hyperplane is able to classify the patterns into two categories and maximize their distance from the hyperplane. To tackle the multi-class problems, the one-vs-rest approach described below is typically used.

GO vectors are used for training one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent SVMs are trained, one for each class. Denote the GO vector created by using the true AC of the i-th query protein as $\mathbf{q}_{i,0}$ and the GO vectors created by using the AC of the k-th homolog as $\mathbf{q}_{i,k}$, $k = 1, \ldots, n$, where n is the number of homologs retrieved by BLAST with the default parameter setting. Then, given the i-th query protein $\mathbb{Q}_i$, the score of the m-th SVM is:

$$s_m(\mathbb{Q}_i) = \sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r, \mathbf{q}_{i,h}) + b_m, \tag{4.6}$$

where

$$h = \min \Big\{ k \in \{0, \ldots, n\} \text{ s.t. } ||\mathbf{q}_{i,k}||_0 \neq 0 \Big\}, \tag{4.7}$$

and $\mathcal{S}_m$ is the set of support vector indexes corresponding to the m-th SVM, $y_{m,r} = 1$ when $\mathbf{p}_r$ belongs to class m and $y_{m,r} = -1$ otherwise, $\alpha_{m,r}$ are the Lagrange multipliers, and $K(\cdot, \cdot)$ is a kernel function. In this work, linear kernels were used, i.e., $K(\mathbf{p}_r, \mathbf{q}_{i,k}) = \langle \mathbf{p}_r, \mathbf{q}_{i,k} \rangle$. The predicted class of the i-th query protein is given by

$$m^* = \arg \max_{m=1}^{M} s_m(\mathbb{Q}_i). \tag{4.8}$$

Note that $\mathbf{p}_r$'s in Eq. 4.6 represent the GO training vectors, which may include the GO vectors created by using the true ACs of the training proteins or their homologous ACs. We have the following two cases:

1. If the true ACs are available, $\mathbf{p}_r$'s represent the GO training vectors created by using the true ACs only.

2. If only the AA sequences are known, then only the ACs of the homologous sequences

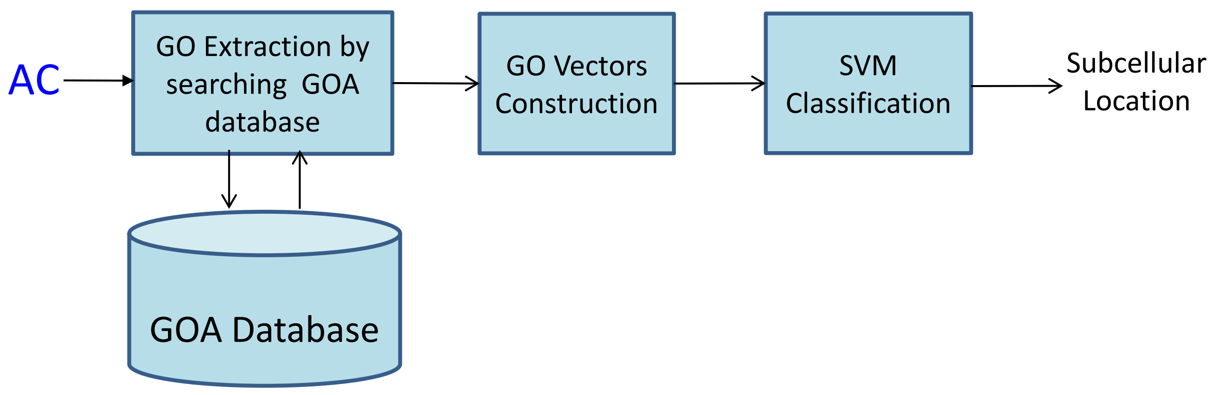


Figure 4.2: Flowchart of GOASVM that uses protein accession numbers (AC) only as input.

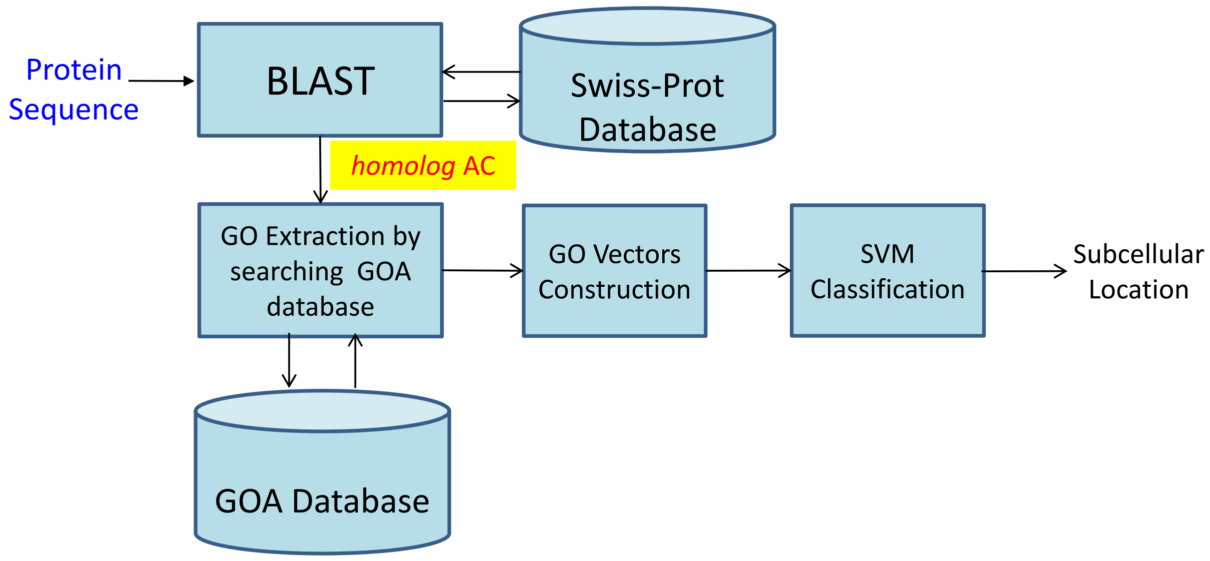


Figure 4.3: Flowchart of GOASVM that uses protein sequences only as input. *AC*: Accession Number.

can be used for training the SVM and for scoring. In that case, $\mathbf{p}_r$'s represent the GO training vectors created by using the homologous ACs only.

Fig. 4.2 and Fig. 4.3 illustrate the prediction process of GOASVM using protein accession numbers (ACs) and protein sequences as input, respectively.

4.2 FusionSVM: Fusion of Gene Ontology and Homology-Based Features

This section introduces a fusion predictor, namely FusionSVM, which integrates GO features and homology-based features for classification.

4.2.1 InterProSVM: Extracting GO from InterProScan

Similar to GOASVM, the prediction process is also divided into two stages: feature extraction (vectorization) and pattern classification. However, unlike GOASVM which retrieves GO terms from the GOA database, InterProSVM extracts GO terms from a program called InterProScan,[5] which does not need ACs of proteins nor BLAST, and may produce more GO terms correlated with molecular functions.

The construction of GO vectors is divided into two steps. First, a collection of distinct GO terms is obtained by presenting all of the sequences in a dataset to InterProScan. For each query sequence, InterProScan returns a file containing the GO terms found by various protein-signature recognition algorithms (we used all available algorithms in this work). Using the dataset described in Table 8.4 of Chapter 8, we found 1203 distinct GO terms, from GO:0019904 to GO:0016719. These GO terms form a GO Euclidean space with 1203 dimensions.

In the second step, for each sequence in the dataset, we constructed a GO vector by matching its GO terms to all of the 1203 GO terms determined in the first step. Similar to GOASVM, the four GO-vector construction methods have been investigated.

[5]http://www.ebi.ac.uk/Tools/pfa/iprscan/#

4.2.1.1 Post-processing of GO Vectors

Although the raw GO vectors can be directly applied to support vector machines (SVMs) for classification, better performance may be obtained by post-processing the raw vectors before SVM classification. Here we introduce two post-processing methods: (1) vector norm and (2) geometric mean.

1. **Vector Norm**. Given the i-th GO training vector $\mathbf{p}_i$, the vector is normalized as:

$$\mathbf{x}_i^{(v)} = [x_{i,1}^{(v)}, \ldots, x_{i,1203}^{(v)}]^\mathsf{T} \text{ where } x_{i,j}^{(v)} = \frac{p_{i,j}}{\|\mathbf{p}_i\|} \tag{4.9}$$

   where the superscript (v) stands for vector norm, and $p_{i,j}$ is the j-th element of $\mathbf{p}_i$. In case $\|\mathbf{p}_i\|= 0$, we set all the element of $x_{i,j}^{(v)} = 0$. Similarly, given the i-th test vector $\mathbf{q}_i$, the GO test vector is normalized as:

$$\mathbf{x}_i^{(v)\prime} = \left[x_{i,1}^{(v)\prime}, \ldots, x_{i,1203}^{(v)\prime}\right]^\mathsf{T} \text{ where } x_{i,j}^{(v)\prime} = \frac{q_{i,j}}{\|\mathbf{q}_i\|} \tag{4.10}$$

2. **Geometric Mean**. This method involves pairwise comparison of GO vectors, followed by normalization.

   -Pairwise Comparison: Denote $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_T]^\mathsf{T}$ as a $T \times 1203$ matrix whose rows are the raw GO vectors of T training sequences. Given the i-th GO training vector $\mathbf{p}_i$, we compute the dot products between $\mathbf{p}_i$ and each of the training GO vectors to obtain a T-dim vector:

$$\mathbf{x}_i = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_T]^\mathsf{T} \mathbf{p}_i = \mathbf{P}\mathbf{p}_i \ , \ i = 1, \ldots, T. \tag{4.11}$$

   During testing, given the i-th test vector $\mathbf{q}_i$, we compute

$$\mathbf{x}_i' = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_T]^\mathsf{T} \mathbf{q}_i = \mathbf{P}\mathbf{q}_i \ , \ i = 1, \ldots, T' \tag{4.12}$$

where T' is the number of test vectors (sequences).

-Normalization: The j-th elements of $\mathbf{x}_i$ is divided by the geometric mean of the i-th element of $\mathbf{x}_i$ and the j-th element of $\mathbf{x}_j$, leading to the normalized vectors:

$$\mathbf{x}_i^{(g)} = [x_{i,1}^{(g)}, \ldots, x_{i,T}^{(g)}]^\mathsf{T} \text{ where } x_{i,j}^{(g)} = \frac{x_{i,j}}{\sqrt{x_{i,i}x_{j,j}}} \tag{4.13}$$

where the superscript (g) stands for geometric mean. Note that we selected those proteins which have at least one GO term (See Section 8.1.1.2), therefore pairwise comparison can guarantee that the elements $x_{i,i}$ and $x_{j,j}$ exist for $i, j = 1, \ldots, T$.

4.2.1.2 Multiclass SVM Classification

After GO vector construction and post-processing, the vectors $\mathbf{p}_i$, $\mathbf{x}_i^{(v)}$, or $\mathbf{x}_i^{(g)}$ can be used for training one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent SVMs are trained. During testing, given the i-th test protein $\mathbb{Q}_i$, the output of the m-th SVM is

$$s_m^{\text{GO}}(\mathbb{Q}_i) = \sum_{r \in \text{SV}_m^{\text{GO}}} \alpha_{m,r}^{\text{GO}} y_{m,r}^{\text{GO}} K^{\text{GO}}(\mathbf{p}_r, \mathbf{q}_i) + b_m^{\text{GO}}, m = 1, \ldots, M \tag{4.14}$$

where SV_m^{GO} is the set of support vector indexes corresponding to the m-th SVM, $y_{m,r}^{\text{GO}} = 1$ when $\mathbf{p}_r$ belongs to class m and $y_{m,r}^{\text{GO}} = -1$ otherwise, $\alpha_{m,r}^{\text{GO}}$ are the Lagrange multipliers, and $K^{\text{GO}}(\mathbf{p}_r, \mathbf{q}_i)$ is a kernel function. The form of $K^{\text{GO}}(\mathbf{p}_r, \mathbf{q}_i)$ depends on the post-processing method being used. For example, if vector norm is used for normalization, the kernel becomes

$$K^{\text{GO}}(\mathbf{p}_r, \mathbf{q}_i) = \left\langle \mathbf{x}_r^{(v)}, \mathbf{x}_i^{(v)'} \right\rangle \tag{4.15}$$

The SVM score $s_m^{\text{GO}}(\mathbf{p}')$ can be combined with the score of the profile alignment SVM described next.

4.2.2 PairProSVM: A Homology-Based Method

Kernel techniques based on profile alignment have been used successfully in detecting remote homologous proteins [121] and in predicting subcellular locations of eukaryotic proteins [59]. Instead of extracting feature vectors directly from sequences, profile alignment methods train an SVM classifier by using the scores of local profile alignment.

This method, namely PairProSVM, extracts the features from protein sequences by aligning the profiles of the sequences with each of the training profiles [59]. A profile is a matrix in which elements in a column (sequence position) specify the frequency of individual amino acids appeared in the corresponding position of some homologous sequences. Given a sequence, a profile can be derived by aligning it with a set of similar sequences. The similarity score between a known and an unknown sequence can be computed by aligning the profile of the known sequence with that of the unknown sequence [121]. Since the comparison involves not only two sequences but also their closely related sequences, the score is more sensitive to detecting weak similarity between protein families.

The profile of a sequence can be obtained by presenting the sequence to PSI-BLAST [122] that searches against a protein database for homologous sequences. The information pertaining to the aligned sequences is represented by two matrices: position-specific scoring matrix (PSSM) and position-specific frequency matrix (PSFM). Each entry of a PSSM represents the log-likelihood of the residue substitutions at the corresponding position in the query sequence. The PSFM contains the weighted observation frequencies of each position of the aligned sequences.

Fig. 4.4 illustrates the flow of the profile alignment method for subcellular localization. Given a query sequence, we first obtain its profile by presenting it to PSI-BLAST. Then we align it with the profile of each training sequence to form an alignment score vector,

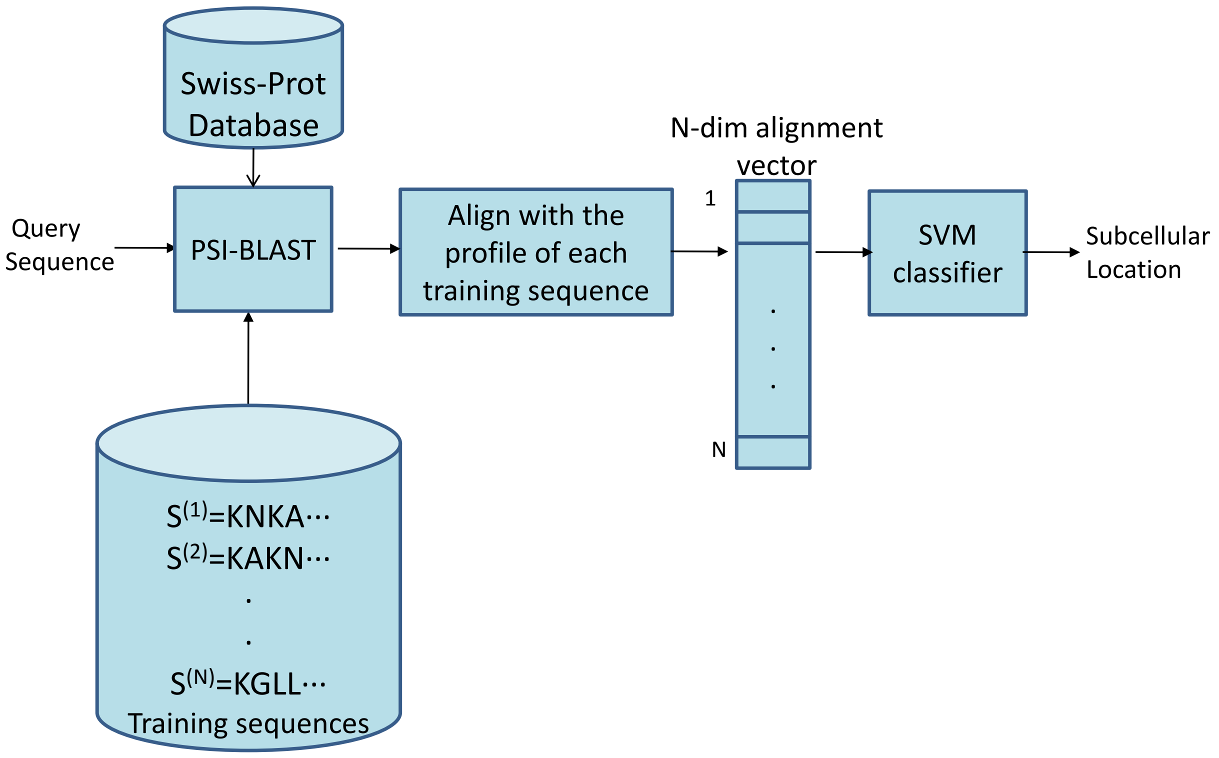


Figure 4.4: Flowchart of profile alignment method.

which is further used as inputs to an SVM classifier for classification. Mathematically, given the i-th test protein sequence, we align its profile with each of the training profiles to obtain a profile-alignment test vector $\mathbf{q}_i$, whose elements are then normalized by the geometric mean as follows:

$$\mathbf{q}_i^{(g)} = [q_{i,1}^{(g)}, \ldots, q_{i,T}^{(g)}]^{\mathsf{T}}, \text{ where } q_{i,j}^{(g)} = \frac{q_{i,j}}{\sqrt{q_{i,i}q_{j,j}}}. \tag{4.16}$$

Similar to the GO method, a one-versus-rest SVM classifier was used to classify the profile-alignment vectors. Specifically, the score of the m-th profile-alignment SVM for the i-th test protein $\mathbb{Q}_i$ is

$$s_m^{\mathrm{PA}}(\mathbb{Q}_i) = \sum_{r \in \mathrm{SV}_m^{\mathrm{PA}}} \alpha_{m,r}^{\mathrm{PA}} y_{m,r}^{\mathrm{PA}} K^{\mathrm{PA}}(\mathbf{p}_r, \mathbf{q}_i) + b_m^{\mathrm{PA}}, m = 1, \ldots, M, \tag{4.17}$$

which is to be fused with the score of the GO SVM.

4.2.3 Fusion of InterProSVM and PairProSVM

Fig. 4.5 illustrates the fusion of InterProGOSVM and PairProSVM. The GO and profile alignment scores produced by the GO and profile alignment SVMs are normalized by Z-norm:

$$\tilde{s}_m^{\mathrm{GO}}(\mathbb{Q}_i) = \frac{s_m^{\mathrm{GO}}(\mathbb{Q}_i) - \mu_m^{\mathrm{GO}}}{\sigma_m^{\mathrm{GO}}} \text{ and } \tilde{s}_m^{\mathrm{PA}}(\mathbb{Q}_i) = \frac{s_m^{\mathrm{PA}}(\mathbb{Q}_i) - \mu_m^{\mathrm{PA}}}{\sigma_m^{\mathrm{PA}}}, \quad m = 1, \ldots, M, \tag{4.18}$$

where $(\mu_m^{\mathrm{GO}}, \sigma_m^{\mathrm{GO}})$ and $(\mu_m^{\mathrm{PA}}, \sigma_m^{\mathrm{PA}})$ are respectively the mean and standard derivation of the GO and profile alignment SVM scores derived from the training sequences. The advantage of Z-norm is that estimating the normalization parameters can be done off-line during training [123]. The normalized GO and profile-alignment SVM scores are fused:

$$\tilde{s}_m^{\mathrm{Fuse}}(\mathbb{Q}_i) = w^{\mathrm{GO}} \tilde{s}_m^{\mathrm{GO}}(\mathbb{Q}_i) + w^{\mathrm{PA}} \tilde{s}_m^{\mathrm{PA}}(\mathbb{Q}_i) \tag{4.19}$$

where $w^{\mathrm{GO}} + w^{\mathrm{PA}} = 1$. Finally, the predicted class of the test sequence is given by

$$m^* = \arg \max_{m=1}^{M} \tilde{s}_m^{\mathrm{Fuse}}(\mathbb{Q}_i). \tag{4.20}$$

For ease of reference, the fusion predictor is referred to as FusionSVM.

4.3 Summary

This chapter has presented two predictors for single-location protein subcellular localization, namely GOASVM and FusionSVM. Both predictors use GO information as features and SVM as classifiers for prediction. Moreover, the ways to construct GO vectors are the same for these two predictors.

However, there are three differences between GOASVM and FusionSVM: (1) the former retrieves the GO terms from the GOA database while the latter from the InterProScan program; (2) the former does not post-process the GO vectors while the latter does; (3)

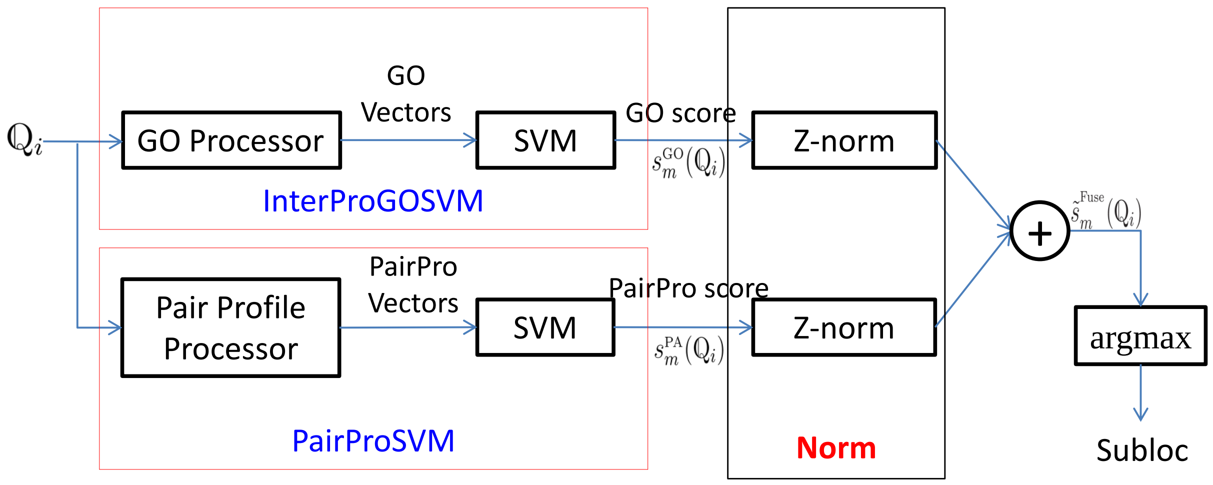


Figure 4.5: Flowchart of fusion of InterProGOSVM and PairProSVM.

the former uses only GO information as features and adopts a successive-search strategy to make sure this method is applicable to novel proteins, while the latter combines GO features and homology-based features for prediction.

Chapter 5

From Single-Location to Multi-Location

Instead of only determining subcellular localization of single-label proteins, this chapter will focus on predicting both single- and multi-location proteins. Biological significance of multi-location proteins will be first elaborated, followed by a brief introduction of existing algorithms for multi-label classification. Subsequently, three multi-label predictors, namely mGOASVM, AD-SVM and mPLR-Loc, are presented for multi-location protein subcellular location.

5.1 Significance of Multi-Location Proteins

Previous chapters show that remarkable progress in the development of computational methods has been made in the past decades for protein subcellular localization. Unfortunately, most of the existing methods are limited to the prediction of single-location proteins. These methods generally exclude the multi-label proteins or are based on the assumption that multi-location proteins do not exist. Besides, the focus on predicting single-location proteins may also be driven by the data available in public databases such as UniProt, where the majority of the proteins are typically assigned to a single location. However, there exist multi-location proteins that can simultaneously reside at, or move between, two or more different subcellular locations [124, 125, 126, 127]. Actually, proteins with multiple locations play important roles in some metabolic processes that take place in more than one compartment. For example, fatty acid β-oxidation in the peroxisome and mitochondria, and antioxidant defense in the cytosol, mitochondria and peroxisome[128]; GLUT4, a glucose transporter regulated by insulin, which is typically stored in the intracellular vesicles of adipocytes, has been found to translocate to the plasma membrane in response to insulin [129, 130].

5.2 Multi-Label Classification

Traditionally, pattern classification problems are concerned with learning from a set of patterns, where each pattern is associated with one of the known classes only. These problems are referred to as single-label, multi-class classification. However, many real-world problems are not limited to single-label classification. When more than one label are assigned to the data instance, the problems are referred to as multi-label multi-class classification. In the past decades, multi-label classification has received significant attention in a wide range of problem domains, such as functional genomics prediction [131, 132, 133], text categorization [134, 135, 136, 137, 138], music classification [139, 140], video segmentation [141], and semantic annotation of images [142]. In functional genomics prediction, a gene is likely to associate with many functions. In text categorization, a document describing the politics may involve other topics, such as sports or education. Similarly, in music classification, a song may belong to more than one genre.

Compared to single-label classification, multi-label classification is more complicated due to the large number of possible combinations of labels. Existing methods for multi-label classification can be grouped into two main categories: (1) algorithm adaptation and (2) problem transformation.

5.2.1 Algorithm-Adaptation Methods

Algorithm adaptation methods extend specific single-label algorithms to solve multi-label classification problems. Typical methods include AdaBoost.MH [134], multi-label C4.5 [143], and hierarchical multi-label decision trees [132].

AdaBoost.MH is an extension of AdaBoost for multi-label classification. It uses the one-vs-rest approach to convert an M-class problem into M 2-class AdaBoost problems in which an additional feature defined by the class labels is augmented to the input space.

In [143], a decision tree (C4.5) is used as a baseline algorithm and it extends the definition of entropy to include multi-label data by estimating the number of bits needed to describe the membership or non-membership of each class. One disadvantage of this algorithm is that it only learns a set of accurate rules, not a complete classification.

In [132], class labels are organized in a hierarchy and for each class, a binary decision tree is learned in a hierarchical way. An example can only belong to a class if it also belongs to the class's superclasses. This parent-children relationship enables the decision tree to predict multi-label instances.

5.2.2 Problem-Transformation Methods

Problem transformation methods transform a multi-label learning problem into one or more single-label classification problems [142] so that traditional single-label classifiers can be applied without modification. Typical methods include label powerset (LP) [144], binary relevance (BR) [145], ensembles of classifier chains (ECC) [146] and compressive sensing [147].

Label powerset method reduces a multi-label task to a single-label task by treating each possible multi-label subset as a new class in the single-label classification task. This method is simple, but is likely to generate a large number of classes, many of which are associated with a few examples only.

Binary relevance (BR) is a popular problem-transformation method. It transforms a multi-label task into many binary classification tasks, one for each label. Given a query instance, its predicted label(s) are the union of the positive-class labels output by these binary classifiers. BR is effective, but it neglects the correlation between labels, which may carry useful information for multi-label classification.

The classifier chain method is a variant of BR but it can take the correlation between labels into account. Similar to BR, a set of one-vs-rest binary classifiers are trained. But unlike BR, the classifiers are linked in a chain and the feature vectors presented to the i-th classifier in the chain are augmented with the binary values representing the label(s) of the feature vectors up to the $(i-1)$-th class. Therefore, label dependence is preserved through the feature space. Classification performance, however, depends on the chain order. This order-dependency can be overcome by ensembles of classifier chains [146].

The compressive sensing approach is motivated by the fact that when the number of classes is large, the actual labels are often sparse. In other words, a typical query instance will belong to a few classes only, even though the total number of classes is large. This approach exploits the sparsity of the output (label) space by means of compressive sensing to obtain a more efficient output coding scheme for large-scale multi-label learning problems.

Several algorithms based on support vector machines (SVM) [148] have been proposed to tackle multi-label classification problems. In [149], a ranking algorithm for multi-label classification is proposed. It uses the ranking loss [134] as the cost function, which is defined as the average fraction of pairs of labels that are ordered incorrectly. It follows the philosophy of SVMs, but one major disadvantage of this method is that it does not output a set of labels. In [150], the BR method is adopted to SVM classifiers by extending the original data set with some additional features indicating the relationship between classes.

Compared to algorithm adaptation methods, one advantage of problem transformation methods is that any algorithm which is not capable of dealing with multi-label classification problems can be easily extended to deal with multi-label classification via transformation. It should be pointed out that the multi-label classification methods are different from the multi-class classification methods, such as error-correcting-output-coding methods [151], pairwise comparison methods [152], and so on. There is probably no multi-class method that outperforms all others in all circumstances [153], so is the same case for multi-label methods.

5.2.3 Applications of Multi-Label Classification in Bioinformatics

In the past decades, multi-label classification methods have been increasingly applied to bioinformatics, especially in protein subcellular localization. Several multi-label predictors have been proposed to deal with the prediction of multi-label proteins in species of virus, plant and eukaryote, which are elaborated below.

There are a few predictors [154, 93, 155] specifically designed for predicting viral proteins, generated by viruses in various cellular compartments of the host cell or virus-infected cells. Studying the subcellular localization of viral proteins enables us to obtain the information about their destructive tendencies and consequences [155, 154, 93]. It is also beneficial to the annotation of the functions of viral proteins and the design of antiviral drugs. To the best of our knowledge, there are two predictors, namely Virus-mPLoc [154] and iLoc-Virus [93], capable of predicting multi-label viral proteins. iLoc-Virus performs better than Virus-mPLoc because the former has a better formulation for reflecting GO information and has a better way to predict the number of subcellular location sites of a query protein [93]. Recently, a method called KNN-SVM ensemble classifier [156] is proposed to deal with multi-label proteins, including viral proteins. It was found that the performance of the KNN-SVM predictor is comparable to iLoc-Virus and is better than Virus-mPLoc.

Conventional methods specializing for plant proteins, such as TargetP [52] and Predotar [157], can only deal with single-label proteins. Plant-mPLoc [96] and iLoc-Plant [87] are state-of-the-art predictors that can deal with single-label and multi-label proteins of plants. iLoc-Plant performs better than Plant-mPLoc because of the similar improvement as in iLoc-Virus versus Virus-mPLoc.

Prediction of eukaryotic proteins is one of the focal points in the past decades. Many predictors [72, 21, 25, 28, 42] were proposed to single-label predict eukaryotic protein subcellular localization. Euk-mPLoc 2.0 [158] and iLoc-Euk [90] are state-of-the-art predictors that can deal with both single-label and multi-label proteins of eukaryotes. Similar to iLoc-Virus versus Virus-mPLoc, iLoc-Euk performs better than Euk-mPLoc 2.0.

Among these multi-label predictors, Virus-mPLoc, Plant-mPLoc, Euk-mPLoc 2.0, iLoc-Virus, iLoc-Plant and iLoc-Euk use algorithm adaptation methods, while KNN-SVM uses problem transformation methods.

5.3 mGOASVM: A Predictor for Both Single- and Multi-Location Proteins

This section proposes an efficient multi-label predictor, namely mGOASVM, for multi-label protein subcellular localization prediction. Here, the prefix “m” stands for multiple, meaning that the predictor can deal with proteins with multiple subcellular locations. mGOASVM is different from other predictors in that (1) it adopts a new decision scheme

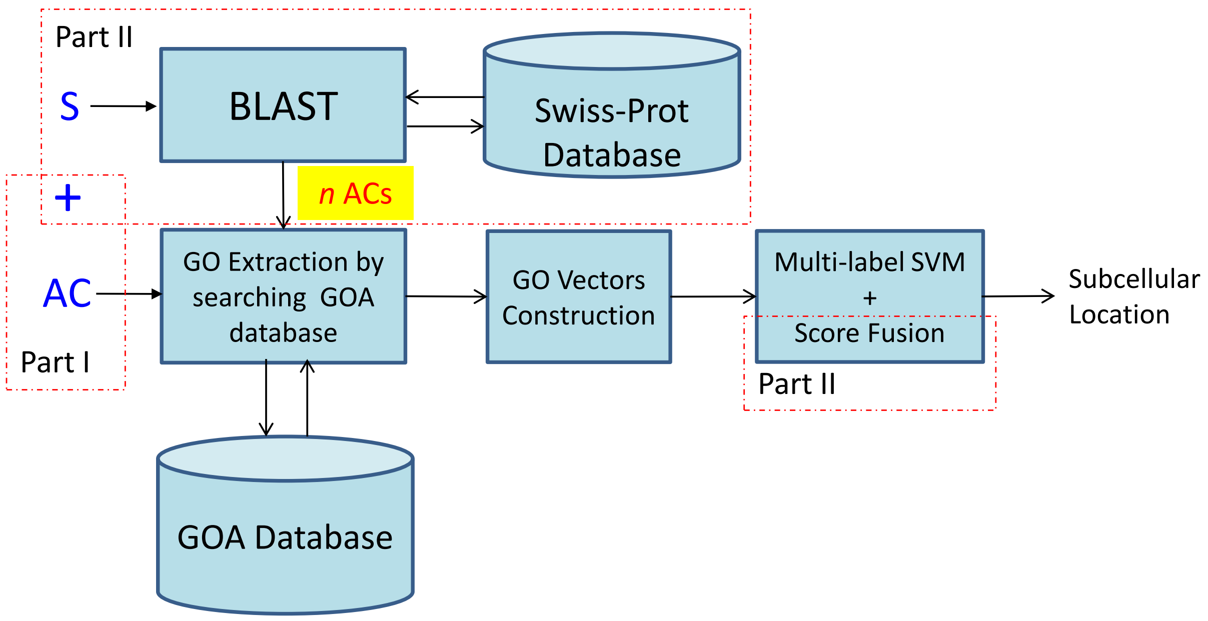


Figure 5.1: **Flowchart of mGOASVM for three cases: (1) using accession numbers only; (2) using sequences only; (3) using both accession numbers and sequences.** AC: Accession Number; S: Sequence. Part II does not exist for Case 1, and Part I does not exist for Case 2. Case 3 requires using both Part I and Part II. The score fusion implements Eq. 5.1.

for an SVM classifier so that it can effectively deal with datasets containing both single-label and multi-label proteins; (2) it selects a set of distinct, relevant GO terms to form a more informative GO subspace; (3) it constructs the GO vectors by using the frequency of occurrences of GO terms instead of using 1-0 values [63, 154, 96] for indicating the presence or absence of some predefined GO terms. The results in Chapter 9 on two benchmark datasets and a newly created dataset full of novel proteins demonstrate that these three properties enable mGOASVM to predict multi-location proteins and outperform the state-of-the-art predictors such as iLoc-Virus and iLoc-Plant.

5.3.1 Feature Extraction

The feature extraction part of mGOASVM is similar to GOASVM introduced in Chapter 4. mGOASVM adopts a method of retrieving GO terms similar to GOASVM, which was specified in Section 4.1.2 of Chapter 4. The only difference is that we tried using more than one homolog for classification. For the GO-vector construction, mGOASVM only adopts the term-frequency method introduced in Section 4.1.3 of Chapter 4, which has been demonstrated to be superior to other three construction methods in Chapter 9. However, compared to GOASVM, mGOASVM uses a more sophisticated multi-label multi-class classifier for multi-location protein subcellular localization, which is elaborated below.

5.3.2 Multi-label Multi-class SVM Classification

To predict the subcellular locations of datasets containing both single-label and multi-label proteins, a multi-label support vector machine (SVM) classifier is proposed in this section. GO vectors, which can be obtained from Eq. 4.3 in Section 4.1.3 of Chapter 4, are used for training the multi-label one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent binary SVMs are trained, one for each class. Denote the GO vector created by using the true accession number of the i-th query protein as $\mathbf{q}_{i,0}$ and the GO vectors created by using the n homologous accession numbers as $\mathbf{q}_{i,j}$, $j = 1, \ldots, n$. Then, the score of the m-th SVM given the i-th query protein is

$$s_m(\mathbb{Q}_i) = \sum_{j=0}^{n} w_j \left(\sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r, \mathbf{q}_{i,j}) + b_m \right) \tag{5.1}$$

where $\mathcal{S}_m$ is the set of support vector indexes corresponding to the m-th SVM, $\alpha_{m,r}$ are the Lagrange multipliers, $K(\cdot,\cdot)$ is a kernel function, and w_j's are fusion weights such that $\sum_{j=0}^{n} w_j = 1$. In this work, linear kernels were used, i.e., $K(\mathbf{p}_r, \mathbf{q}_{i,j}) = \langle \mathbf{p}_r, \mathbf{q}_{i,j} \rangle$. $y_{m,r} \in \{-1, +1\}$ are the class labels (here we call them "the transformed labels"), which are denoted as:

1. For single-label $\mathbf{p}_r$,
$$y_{m,r} = \begin{cases} +1 & , \text{if } \mathcal{L}(\mathbf{p}_r) = m \\ -1 & , \text{otherwise.} \end{cases} \tag{5.2}$$

2. For multi-label $\mathbf{p}_r$,
$$y_{m,r} = \begin{cases} +1 & , \text{if } \mathcal{L}(\mathbf{p}_r) \bigcap \{m\} \neq \emptyset \\ -1 & , \text{otherwise.} \end{cases} \tag{5.3}$$

where $m \in \{1, \ldots, M\}$. Note that unlike the single-label problem where each protein has one and only one positive transformed label, a multi-label protein can have more than one positive transformed label.

Then the subcellular location(s) of the i-th query protein will be predicted as:

$$\mathcal{M}^*(\mathbb{Q}_i) = \bigcup_{m=1}^{M} \{m : s_m(\mathbb{Q}_i) > 0\}. \tag{5.4}$$

As can be seen, $\mathcal{M}^*(\mathbb{Q}_i)$ is a predicted set that may have zero, one, or more than one element, which enables us to make multi-label prediction. In case Eq. 5.4 does not produce a class label, i.e., $\mathcal{M}^*(\mathbb{Q}_i) = \emptyset$, the number of subcellular locations is set to one and the location is given by

$$\mathcal{M}^*(\mathbb{Q}_i) = \arg \max_{m=1}^{M} s_m(\mathbb{Q}_i). \tag{5.5}$$

Note that $\mathbf{p}_r$'s in Eq. 5.1 represents the GO training vectors, which may include the GO vectors created by using the true accession numbers of the training proteins or their homologous accession numbers. We have the following three cases.

Case 1. If only the true accession numbers are available, then only $\mathbf{q}_{i,0}$'s exist and $\mathbf{p}_r$'s represent the GO training vectors created by using the true accession numbers only.

In that case, $\mathbf{q}_{i,j}$ $(j = 1, \ldots, n)$ do not exist; $w_0 = 1$ and $w_j = 0$ $(j = 1, \ldots, n)$. Then, Eq 5.1 can be written as:

$$s_m(\mathbb{Q}_i) = \sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r, \mathbf{q}_{i,0}) + b_m. \tag{5.6}$$

Case 2. If only the amino acid sequences are known, then only the accession numbers of the homologous sequences can be used for training the SVM and for scoring. In that case, $\mathbf{q}_{i,0}$ does not exist and $w_0 = 0$; moreover, $\mathbf{p}_r$'s represent the GO training vectors created by using the homologous accession numbers only.

Case 3. If both accession numbers and amino acid sequences are known, then both true accession numbers and the accession numbers of the homologous sequences are used for training the SVM and for scoring. Then, $\mathbf{q}_{i,j}$ $(j = 0, \ldots, n)$ exist, and $\mathbf{p}_r$'s represent the GO training vectors created by using both the true accession numbers and the homologous accession numbers.

In this work, 1, 2, 4 and 8 homologs were tried for the virus dataset, and 1 and 2 homologs were used for the plant dataset, respectively, i.e., $n \in \{1, 2, 4, 8\}$ and $n \in \{1, 2\}$ in Eq. 5.1, respectively. For convenience, equal weights for the true accession number and the accession numbers of homologs were adopted. Therefore, for Case 2, $w_0 = 0$ and $w_j = 1/n$, $j = 1, \ldots, n$; and for Case 3, we set $w_j = 1/(n + 1)$, $j = 0, \ldots, n$.

Fig. 5.1 illustrates the whole prediction process in mGOASVM for all the three cases: (1) using accession numbers only, (2) using sequences only and (3) using both accession numbers and sequences. Part II does not exist for Case 1, and Part I does not exist for Case 2. Both Part I and Part II exist for Case 3. Score fusion is the fusion of GO scores obtained from accession numbers of homologs in Case 2 or from both true accession numbers and accession numbers of homologs in Case 3.

5.4 AD-SVM: An Adaptive-decision Multi-Label Predictor

To determine the number of subcellular locations in which a protein will reside, mGOASVM introduced in Section 5.3 typically compares some pattern-matching scores with a fixed decision threshold. This simple strategy, however, may easily lead to over-prediction. To address this problem, this section proposes an adaptive decision (AD) scheme for multi-label SVM classifiers, which formulates a more powerful multi-label classifier, namely AD-SVM. AD-SVM extends binary relevance methods with an adaptive decision scheme that essentially converts the linear SVMs in the classifier into piecewise linear SVMs, which effectively reduces the over-prediction instances while having little influence on the correctly predicted ones.

AD-SVM uses similar feature extraction methods as mGOASVM. The feature vectors are classified by the proposed adaptive-decision multi-label classifier, which is elaborated below.

5.4.1 Multi-label SVM Scoring

GO vectors, as computed in Eq. 4.3, are used for training the multi-label one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent binary SVMs are trained, one for each class. Denote the GO vector created by using the true AC of the i-th query protein as $\mathbf{q}_{i,0}$ and the GO vector created by using the accession number of the k-th homolog as $\mathbf{q}_{i,k}$, $k = 1, \ldots, k_{\max}$, where $k_{\max}$ is the number of homologs retrieved by BLAST with the default parameter setting. Then, given the i-th query protein $\mathbb{Q}_i$, the score of the m-th SVM is:

$$s_m(\mathbb{Q}_i) = \sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r, \mathbf{q}_{i,h}) + b_m, \tag{5.7}$$

where

$$h = \min\left\{k \in \{0, \ldots, k_{\max}\} \text{ s.t. } ||\mathbf{q}_{i,k}||_0 \neq 0\right\}, \tag{5.8}$$

and $\mathcal{S}_m$ is the set of support vector indexes corresponding to the m-th SVM, $y_{m,r} \in \{-1, +1\}$ are the class labels, $\alpha_{m,r}$ are the Lagrange multipliers, $K(\cdot, \cdot)$ is a kernel function; here, the linear kernel is used. Note that $\mathbf{p}_r$'s in Eq. 5.7 represents the GO training vectors, which may include the GO vectors created by using the true AC of the training sequences or their homologous ACs.

5.4.2 Adaptive Decision for SVM (AD-SVM)

To predict the subcellular locations of datasets containing both single-label and multi-label proteins, an adaptive decision scheme for multi-label SVM classifiers is proposed. Unlike the single-label problem where each protein has one predicted label only, a multi-label protein could have more than one predicted labels. Thus, the predicted subcellular location(s) of the i-th query protein are given by:

If $\exists\ s_m(\mathbb{Q}_i) > 0$,

$$\mathcal{M}(\mathbb{Q}_i) = \bigcup_{m=1}^{M} \{\{m : s_m(\mathbb{Q}_i) > 1.0\} \cup \{m : s_m(\mathbb{Q}_i) \geq f(s_{\max}(\mathbb{Q}_i))\}\}, \tag{5.9}$$

otherwise,

$$\mathcal{M}(\mathbb{Q}_i) = \arg \max_{m=1}^{M} s_m(\mathbb{Q}_i). \tag{5.10}$$

In Eq. 5.9, $f(s_{\max}(\mathbb{Q}_i))$ is a function of $s_{\max}(\mathbb{Q}_i)$, where $s_{\max}(\mathbb{Q}_i) = \max_{m=1}^{M} s_m(\mathbb{Q}_i)$. In this work, we used a linear function as follows:

$$f(s_{\max}(\mathbb{Q}_i)) = \theta s_{\max}(\mathbb{Q}_i), \tag{5.11}$$

where $\theta \in [0.0, 1.0]$ is a parameter. Because $f(s_{\max}(\mathbb{Q}_i))$ is linear, Eq. 5.9 and Eq. 5.10 turn the linear SVMs into piecewise linear SVMs. Eq. 5.9 also suggests that the predicted

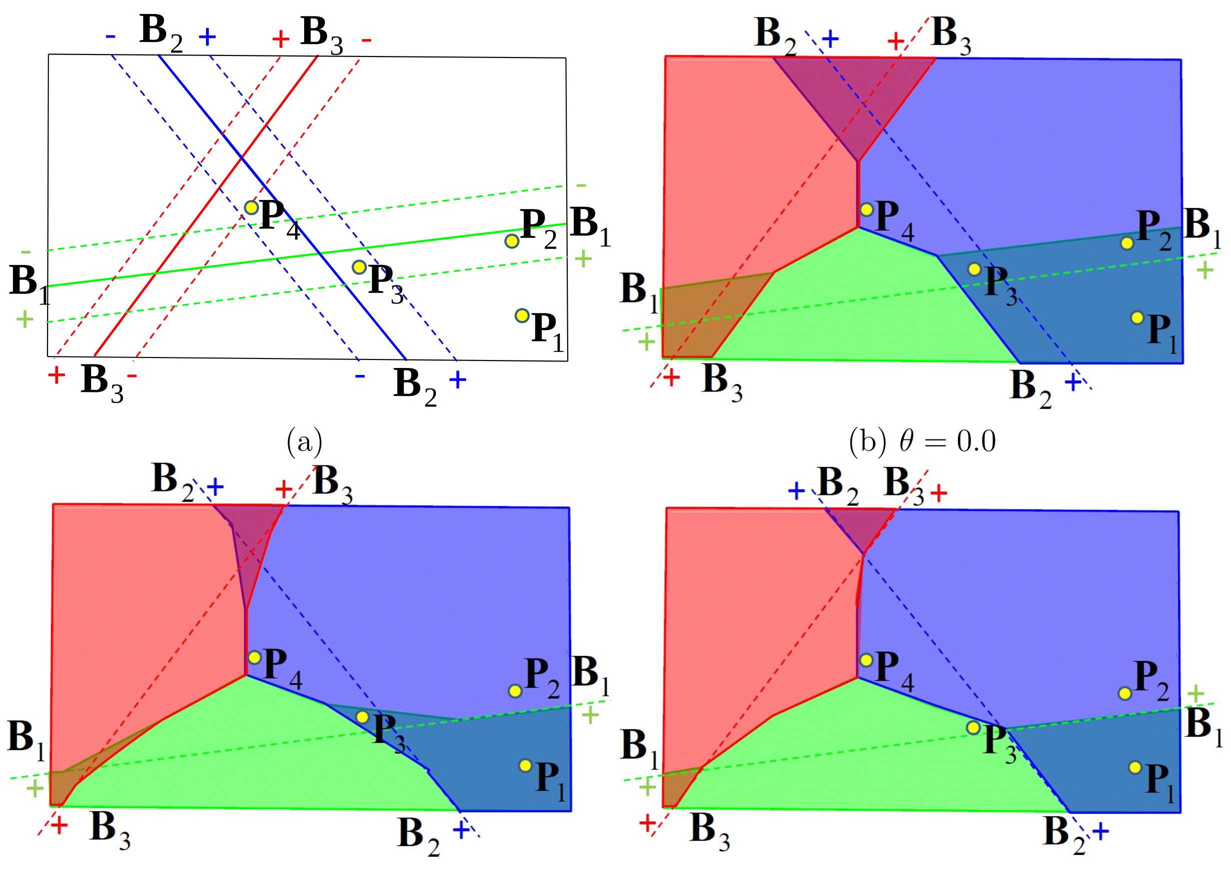


Figure 5.2: A 3-class example illustrating how the adaptive decision scheme changes the decision boundaries from linear to piecewise linear and how the resulting SVMs assign label(s) to test points when θ in Eq. 5.11 changes from 0 to 1. In (a), the solid and dashed lines respectively represent the decision boundaries and margins of individual SVMs. In (b)–(d), the input space is divided into three 1-label regions (green, blue and red) and three 2-label regions (green ∩ blue, blue ∩ red, and red ∩ green).

labels depend on $s_{\max}(\mathbb{Q}_i)$, a function of the test instance (or protein). This means that the decision and the corresponding threshold are adaptive to the test protein. For ease of reference, we refer to the proposed predictor as AD-SVM.

5.4.3 Analysis of AD-SVM

To facilitate discussion, let's define two terms: *over-prediction* and *under-prediction*. Specifically, over (under) prediction means that the number of predicted labels of a query protein is larger (smaller) than the ground-truth. In this chapter, both over- and under-predictions are considered as incorrect predictions, which will be reflected in the "overall actual accuracy (OAA)" to be defined in Section 8.2.3.

Conventional methods use a fixed threshold to determine the predicted classes. When the threshold is too small, the prediction results are liable to over-prediction; on the other hand, when the threshold is too large, the prediction results are susceptible to under-prediction. To overcome this problem, the adaptive decision scheme in the classifier uses the maximum score ($s_{\max}(\mathbb{Q}_i)$) among the one-vs-rest SVMs in the classifier as a reference. In particular, $s_{\max}(\mathbb{Q}_i)$ in Eq. 5.9 adaptively normalizes the scores of all one-vs-rest SVMs so that for SVMs to be considered as runner-ups, they need to have a sufficiently large score relative to the winner. This strategy effectively reduces the chance of over-prediction. The first condition in Eq. 5.9 ($s_m(\mathbb{Q}_i) > 1$) aims to avoid under-prediction when the winning SVM has very high confidence (i.e., $s_{\max}(\mathbb{Q}_i) \gg 1$) but the runners-up still have enough confidence ($s_m(\mathbb{Q}_i) > 1$) in making a right decision.[1] On the other hand, when the maximum score is small (say $0 < s_{\max}(\mathbb{Q}_i) \leq 1$), θ in the second term of Eq. 5.9 can strike a balance between over-prediction and under-prediction. When all of the SVMs have very low confidence (say $s_{\max}(\mathbb{Q}_i) < 0$), the classifier switches to single-label mode via Eq. 5.10.

To further illustrate how this decision scheme works, an example is shown in Fig. 5.2. Suppose there are 4 test data points ($\mathbf{P}_1, \ldots, \mathbf{P}_4$) which are possibly distributed into 3

[1]SVM scores larger than one means that the test proteins fall beyond the margin of separation; therefore, the confidence is fairly high.

classes: {green, blue, red}. The decision boundaries of individual SVMs and the 4 points are shown in Fig. 5.2(a). Suppose $s_m(\mathbf{P}_i)$ is the SVM score of $\mathbf{P}_i$ with respect to the class m, where $i = \{1, \ldots, 4\}$ and $m \in$ {green, blue, red}. Fig. 5.2(a) suggests the following conditions:

$$\begin{array}{rrr} s_{\text{green}}(\mathbf{P}_1) > 1, & s_{\text{blue}}(\mathbf{P}_1) > 1, & s_{\text{red}}(\mathbf{P}_1) < 0; \\ 0 < s_{\text{green}}(\mathbf{P}_2) < 1, & s_{\text{blue}}(\mathbf{P}_2) > 1, & s_{\text{red}}(\mathbf{P}_2) < 0; \\ 0 < s_{\text{green}}(\mathbf{P}_3) < 1, & 0 < s_{\text{blue}}(\mathbf{P}_3) < 1, & s_{\text{red}}(\mathbf{P}_3) < 0; \\ s_{\text{green}}(\mathbf{P}_4) < 0, & s_{\text{blue}}(\mathbf{P}_4) < 0, & s_{\text{red}}(\mathbf{P}_4) < 0. \end{array}$$

Note that points whose scores lie between 0 and 1 are susceptible to over-prediction because they are very close to the decision boundaries of the corresponding SVM. The decision scheme used in Eqs. 5.9–5.11 (i.e., $\theta = 0.0$) leads to the decision boundaries shown in Fig. 5.2(b). Based on these boundaries, $\mathbf{P}_1$, $\mathbf{P}_2$ and $\mathbf{P}_3$ will be assigned to class green $\cap$ blue , and $\mathbf{P}_4$ will be assigned to the class with the highest SVM score (using Eq. 5.10). If θ increases to 0.5, the results shown in Fig. 5.2(c) will be obtained. The assignments of $\mathbf{P}_1$, $\mathbf{P}_3$ and $\mathbf{P}_4$ remain unchanged but $\mathbf{P}_2$ will be changed from class green $\cap$ blue to class blue. Similarly, when θ increases to 1.0 (Fig. 5.2(d)), then the class of $\mathbf{P}_3$ will also be determined by the SVM with the highest score. This analysis suggests that when θ increases from 0 to 1, the decision criterion becomes more stringent, which has the effect of shrinking the 2-label regions in Fig. 5.2, thus reducing the over-prediction. Provided that θ is not close to 1, this reduction in over-prediction will not compromise the decisions made by the high scoring SVMs.

5.5 mPLR-Loc: A Multi-Label Predictor Based on Penalized Logistic-Regression

Logistic Regression (LR) is a powerful discriminative classifier which has an explicit probabilistic interpretation built into its model [159]. Traditional logistic regression classifiers, including penalized logistic regression classifiers [160, 161, 162], are only applicable to multi-class classification. This section elaborates an efficient penalized multi-label logistic regression classifier, namely mPLR-Loc, equipped with an adaptive decision scheme.

5.5.1 Single-label Penalized Logistic Regression

Suppose for a two-class single-label problem, we are given a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathcal{R}^{T+1}$ and $y_i \in \{0,1\}$. In our case, $\mathbf{x}_i = \begin{bmatrix} 1 \\ \mathbf{q}_i \end{bmatrix}$, where $\mathbf{q}_i$ is defined in Eq. 4.3. Denote $Pr(Y = y_i|X = \mathbf{x}_i)$ as the posterior probability of the event that X belongs to class y_i given $X = \mathbf{x}_i$. In logistic regression, the posterior probability is defined as:

$$Pr(Y = y_i|X = \mathbf{x}_i) = p(\mathbf{x}_i;\beta) = \frac{e^{\beta^\mathsf{T}\mathbf{x}_i}}{1+e^{\beta^\mathsf{T}\mathbf{x}_i}}, \tag{5.12}$$

where β is a $(T+1)$-dim parameter vector. When the number of training instances (N) is not significantly larger than the feature dimension $(T+1)$, using logistic regression without any regularization often leads to over-fitting. To avoid over-fitting, an L_2-regularization penalty term is added to the penalized cross-entropy error function as follows:

$$\begin{aligned} E(\beta) &= -\sum_{i=1}^{N}[y_i \log(p(\mathbf{x}_i;\beta)) + (1-y_i)\log(1-p(\mathbf{x}_i;\beta))] + \frac{1}{2}\rho\|\beta\|_2^2 \\ &= -\sum_{i=1}^{N}\left[y_i\beta^\mathsf{T}\mathbf{x}_i - \log(1+e^{\beta^\mathsf{T}\mathbf{x}_i})\right] + \frac{1}{2}\rho\beta^\mathsf{T}\beta \end{aligned} \tag{5.13}$$

where ρ is a user-defined penalty parameter to control the degree of regularization, and ρ can be determined by cross-validation.

To minimize $E(\beta)$, we may use the Newton-Raphson algorithm

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 E(\beta^{old})}{\partial\beta^{old}\partial(\beta^{old})^\mathsf{T}}\right)^{-1} \cdot \frac{\partial E(\beta^{old})}{\partial\beta^{old}}, \tag{5.14}$$

where

$$\frac{\partial E(\beta)}{\partial\beta} = -\mathbf{X}^\mathsf{T}(\mathbf{y}-\mathbf{p}) + \rho\beta \tag{5.15}$$

and

$$\frac{\partial^2 E(\beta)}{\partial\beta\partial\beta^\mathsf{T}} = \mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X} + \rho\mathbf{I} \tag{5.16}$$

See Appendix C for the derivations of Eq. 5.15 and Eq. 5.16. In Eqs. 5.15 and 5.16, $\mathbf{y}$ and $\mathbf{p}$ are N-dim vectors whose elements are $\{y_i\}_{i=1}^N$ and $\{p(\mathbf{x}_i;\beta)\}_{i=1}^N$, respectively, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]^\mathsf{T}$, $\mathbf{W}$ is a diagonal matrix whose i-th diagonal element is $p(\mathbf{x}_i;\beta)(1 - p(\mathbf{x}_i;\beta)), i = 1, 2, \ldots, N$.

Substituting Eqs. 5.15 and 5.16 into Eq. 5.14 gives the following iterative formula for estimating β:

$$\beta^{new} = \beta^{old} + (\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X} + \rho\mathbf{I})^{-1}(\mathbf{X}^\mathsf{T}(\mathbf{y} - \mathbf{p}) - \rho\beta^{old}). \tag{5.17}$$

5.5.2 Multi-label Penalized Logistic Regression

In an M-class multi-label problem, the training data set is written as $\{\mathbf{x}_i, \mathcal{Y}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathcal{R}^{T+1}$ and $\mathcal{Y}_i \subset \{1, 2, \ldots, M\}$ is a set which may contain one or more labels. M independent binary one-vs-rest LRs are trained, one for each class. The labels $\{\mathcal{Y}_i\}_{i=1}^N$ are converted to *transformed labels* (Similar to Eqs. 5.2–5.3) $y_{i,m} \in \{0, 1\}$, where $i = 1, \ldots, N$, and $m = 1, \ldots, M$. The 2-class update formula in Eq. 5.17 is then extended to:

$$\beta_m^{new} = \beta_m^{old} + (\mathbf{X}^\mathsf{T}\mathbf{W}_m\mathbf{X} + \rho\mathbf{I})^{-1}(\mathbf{X}^\mathsf{T}(\mathbf{y}_m - \mathbf{p}_m) - \rho\beta_m^{old}), \tag{5.18}$$

where $m = 1, \ldots, M$, $\mathbf{y}_m$ and $\mathbf{p}_m$ are vectors whose elements are $\{y_{i,m}\}_{i=1}^N$ and $\{p(\mathbf{x}_i;\beta_m)\}_{i=1}^N$, respectively, $\mathbf{W}_m$ is a diagonal matrix, whose i-th diagonal element is $p(\mathbf{x}_i;\beta_m)(1 - p(\mathbf{x}_i;\beta_m)), i = 1, 2, \ldots, N$.

Given the i-th GO vector $\mathbf{q}_i$ of the query protein $\mathbb{Q}_i$, the score of the m-th LR is given by:

$$s_m(\mathbb{Q}_i) = \frac{e^{\beta_m^\mathsf{T}\mathbf{x}_i}}{1 + e^{\beta_m^\mathsf{T}\mathbf{x}_i}}, \text{ where } \mathbf{x}_i = \begin{bmatrix} 1 \\ \mathbf{q}_i \end{bmatrix}. \tag{5.19}$$

The probabilistic nature of logistic regression enables us to assign confidence scores for the prediction decisions. Specifically, for the m-th location, its corresponding confidence score is $s_m(\mathbb{Q}_i)$. See Appendix B for the confidence scores produced by the mPLR-Loc server.

5.5.3 Adaptive Decision for LR (mPLR-Loc)

Because the LR scores of a binary LR classifier are posterior probabilities, the m-th class label will be assigned to $\mathbb{Q}_i$ only if $s_m(\mathbb{Q}_i) > 0.5$. To facilitate multi-label classification, the following decision scheme is adopted:

$$\mathcal{M}(\mathbb{Q}_i) = \bigcup_{m=1}^{M} \{\{m : s_m(\mathbb{Q}_i) > 0.5\} \cup \{m : s_m(\mathbb{Q}_i) \geq f(s_{\max}(\mathbb{Q}_i))\}\}, \tag{5.20}$$

where $f(s_{\max}(\mathbb{Q}_i))$ is a function of $s_{\max}(\mathbb{Q}_i)$ and $s_{\max}(\mathbb{Q}_i) = \max_{m=1}^{M} s_m(\mathbb{Q}_i)$. In this work, we used a linear function as follows:

$$f(s_{\max}(\mathbb{Q}_i)) = \theta s_{\max}(\mathbb{Q}_i), \tag{5.21}$$

where $\theta \in (0.0, 1.0]$ is a parameter that can be optimized by using cross-validation experiments. Note that θ cannot be 0.0, or otherwise all of the M labels will be assigned to $\mathbb{Q}_i$. This is because $s_m(\mathbb{Q}_i)$ is a posterior probability, which is always equal to or greater than zero. Clearly, Eq. 5.20 suggests that the predicted labels depend on $s_{\max}(\mathbb{Q}_i)$, a function of the test instance (or protein). This means that the decision and its corresponding threshold are adaptive to the test protein. For ease of reference, we refer to this predictor as mPLR-Loc.

5.6 Summary

This chapter mainly focuses on multi-location protein subcellular localization and presents three efficient multi-label classifiers for prediction, namely mGOASVM, AD-SVM and mPLR-Loc. All of the predictors use GO information as features for classification.

From the perspectives of multi-label classification, all of the three predictors adopt binary relevance methods to deal with multi-label problems. From the perspectives of classifiers, mGOASVM and AD-SVM use the SVM classifier, while mPLR-Loc adopts the logistic regression classifier. From the perspectives of decision schemes, AD-SVM and mPLR-Loc adopt an adaptive decision scheme based on the maximum score of one-vs-rest SVM or LR classifiers, while mGOASVM uses a fixed decision scheme for final decision-making. Compared to mGOASVM and AD-SVM, another noteworthy point for mPLR-Loc is that the LR posterior scores are probabilistic, which may have better biological meanings because it can be naturally regarded as one way to analyze how probable a protein will reside in each subcellular location.

Chapter 6: Mining Deeper on GO for Protein Subcellular Localization

Chapter 7: Ensemble Random Projection for Large-Scale Predictions

The Methodology chapter is a description of the methodological approach(es) taken and an explanation of why they were chosen. It often includes: a general introduction restating the central aims, the details of any part of the methodology that may be unfamiliar to the readers, an explanation of how the results will be analyzed, and an explanation of any limitations with the methodology. The chapter can also include a review of relevant literature to present the theoretical basis for the methodology adopted.

Chapter 6 and 7 further develop the methodology described in earlier chapters. In Chapter 6, two predictors for mining GO information are discussed. In Chapter 7, the method for constructing two-dimension reduced multi-level predictors is introduced.

These chapters are very effective partly because the writer includes the following:

Structure

(Introduction)	Not included
Review of key literature	Section6.1
Describes two predictive methods	Section 6.2-6.3
Summarises chapter and compares models	Section 6.4

Content

- Provides an introductory paragraph for both chapters
- Explains choice of methodology (e.g. Section 6.3.1, paragraph 2)
- Develops introduction effectively, e.g. Section 7.1, paragraph 1:

Background	sentence 1
Description of problem	sentence 2-3
Proposed solution	sentence 4-7

- Outlines the content of the chapter (e.g. Chapter 7 paragraph 2)
- Presents the relevant theoretical background for each methodological choice (e.g. Section 7.1, ‘Related Work’)
- Gives a short introductory paragraph for subsections stating aim of the subsection (e.g. Section 7.2, sentence 1)

- Refers the reader to other sections of the thesis (e.g. Chapter 9) where the performance of one predictor (HybridGO-LOC) is also identified (e.g. Section 6.4, final sentence)
- Makes comparison between equations used for two decision schemes (e.g. Section 7.2.3, paragraph 1, sentence 2)
- Highlights limitations of current methodology (e.g. Section 7.3.1)

Language

- Uses vocabulary to highlight uniqueness of approach, e.g. *novel* (e.g. Section 6.2, paragraph 1, sentence 1).
- Explains equations in text, e.g. *the lemma suggests that…* (e.g. Section 7.2.1, paragraph 2, sentence 1)
- Uses objective language for interpretations, e.g. *It is therefore reasonable to believe* (e.g. Section 6.3.2, paragraph 1, sentence 3)

To consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

💡Avoid vague language, e.g. *some previous works* (e.g. Chapter 6, introduction). It is better to use *A number of.*

💡Give a transitional statement at the end of the chapter that leads into the next chapter.

💡Avoid overusing prepositional structures at the start of paragraphs (e.g. Section 7.1, paragraph 1, paragraph2, paragraph 3).

💡Avoid starting sentences with time expressions, e.g. *so far* (e.g. Section 7.3, paragraph 1, sentence 1) and *recently* (e.g. Section 7.3.1, paragraph 1, sentence 1).

💡Avoid using overusing *in this work* to introduce paragraphs. It is often not needed, e.g. *in this work, we compared* (e.g. Section 6.3.3, paragraph 3, sentence 1) could be better written as *this worked compared* or *~~In this work~~, we compared.*

Chapter 6

Mining Deeper on GO for Protein Subcellular Localization

Previous chapters use Gene Ontology (GO) based information for single-label and multi-label protein subcellular localization prediction, which are demonstrated to perform impressively superior to methods based on other features (See Chapter 9 for details). However, they only focus on the occurrences of GO terms and disregard their relationships. This chapter will mine deeper into the GO database for protein subcellular localization, which leverages not only the GO term occurrences but also the inter-term relationships. Some previous works related with semantic similarity are first presented. Subsequently, a multi-label predictor, namely SS-Loc, based on semantic similarity over GO will be specified. Then, a hybrid-feature predictor, namely HybridGO-Loc, based on both GO term occurrences and semantic similarity will be elaborated.

6.1 Related Work

The GO comprises three orthogonal taxonomies whose terms describe the cellular components, biological processes, and molecular functions of gene products. The GO terms in each taxonomy are organized within a directed acyclic graph. These terms are placed within structural relationships, of which the most important being the 'is-a' relationship (*parent* and *child*) and the 'part-of' relationship (*part* and *whole*) [163, 164]. Recently, the GO consortium has been enriched with more structural relationships, such as 'positively-regulates', 'negatively-regulates' and 'has-part' [165, 166]. These relationships reflect that the GO hierarchical tree for each taxonomy contains redundant information, for which semantic similarity over GO terms can be found.

Since the relationship between GO terms reflects the association between different gene products, protein sequences annotated with GO terms can be compared on the basis of semantic similarity measures. The semantic similarity over GO has been extensively studied and have been applied to many biological problems, including protein function prediction [167, 168], subnuclear localization prediction [98], protein-protein interaction inference [169, 170, 171] and microarray clustering [172]. The performance of these predictors depends on whether the similarity measure is relevant to the biological problems. Over the years, a number of semantic similarity measures have been proposed, some of which have been used in natural language processing.

Semantic similarity measures can be applied at the GO-term level or the gene-product level. At the GO-term level, methods are roughly categorized as node-based and edge-based. The node-based measures basically rely on the concept of information content of terms, which was proposed by Resnik [173] for natural language processing. Later, Lord et al. [103] applied this idea to measure the semantic similarity among GO terms. Lin et al. [174] proposed a method based on information theory and structural information. Subsequently, more node-based measures [175, 176, 177] were proposed. Edge-based measures are based on using the length or the depth of different paths between terms and/or their common ancestors [178, 179, 180, 181]. At the gene-product level, two most common methods are pairwise approaches [182, 183, 184, 185, 67] and groupwise approaches [186, 187, 188, 189]. Pairwise approaches measure similarity between two gene products by combining the semantic similarities between their terms. Groupwise approaches, on the other hand, directly group the GO terms of a gene product as a set, a graph or a vector, and then calculate the similarity by set similarity techniques, graph matching techniques or vector similarity techniques. More recently, Pesquita et al. [190] reviewed the semantic similarity measures applied to biomedical ontologies, and Guzzi et al. [191] provides a comprehensive review on the relationship between semantic similarity measures and biological features.

6.2 SS-Loc: Using Semantic Similarity Over GO

This section proposes a novel predictor, namely SS-Loc, based on the GO semantic similarity for multi-label protein subcellular localization prediction. The predictor proposed is different from other predictors in that (1) it formulates the feature vectors by the semantic similarity over Gene Ontology which contains richer information than only GO terms; (2) it adopts a new strategy to incorporate richer and more useful homologous information from more distant homologs rather than using the top homologs only; (3) it adopts a new decision scheme for an SVM classifier so that it can effectively deal with datasets containing both single-label and multi-label proteins. Results on a recent benchmark dataset demonstrate that these three properties enable the proposed predictor to accurately predict multi-location proteins and outperform three state-of-the-art predictors.

SS-Loc adopts a way of retrieving GO terms similar to GOASVM and mGOASVM, which makes sure that each protein will correspond to at least one GO term. Subsequently, these sets of GO terms are measured based on semantic similarity elaborated below.

6.2.1 Semantic Similarity Measure

To obtain the GO semantic similarity between two proteins, we should start by introducing the semantic similarity between two GO terms. Semantic similarity (SS) is a measure for quantifying the similarity between categorical data (e.g., words in documents), where the notion of similarity is based on the likeliness of meanings in the data. It is originally developed by Resnik [173] for natural language processing. The idea is to evaluate semantic similarity in an 'is-a' taxonomy using the shared information contents of categorical data. In the context of gene ontology, the semantic similarity between two GO terms is based on their most specific common ancestor in the GO hierarchy. The relationships between GO terms in the GO hierarchy, such as 'is-a' ancestor-child, or 'part-of' ancestor-child can be obtained from the SQL database through the link: http://archive.geneontology.org/latest-termdb/go_daily-termdb-tables.tar.gz. Note here only the 'is-a' relationship is considered for semantic similarity analysis [174]. The semantic similarity between two GO terms x and y is defined as [173]:

$$sim(x, y) = \max_{c \in A(x,y)}[-\log(p(c))], \tag{6.1}$$

where $A(x, y)$ is the set of ancestor GO terms of both x and y, and $p(c)$ is the number of gene products annotated to the GO term c divided by the number of all the gene products annotated to the GO taxonomy.

To further incorporate structural information from the GO hierarchy, we used Lin's measures [174] to normalize the above measure. Then given two GO terms x and y, the similarity is calculated as:

$$sim(x, y) = \frac{2 \times \max_{c \in A(x,y)}[-\log(p(c))]}{-\log(p(x)) - \log(p(y))} \tag{6.2}$$

6.2.2 SS Vector Construction

Based on the semantic similarity (SS) between two GO terms, we adopted a continuous measure proposed in [172] to calculate the similarity of two proteins, which are functionally annotated by a set of GO terms. Given two proteins $\mathbb{P}_i$ and $\mathbb{P}_j$, which are annotated by two sets of GO terms $\mathcal{P}_i$ and $\mathcal{P}_j$ retrieved in Section 4.1.2,[1] we first computed $S(\mathcal{P}_i, \mathcal{P}_j)$ as follows:

$$S(\mathcal{P}_i, \mathcal{P}_j) = \sum_{x \in \mathcal{P}_i} \max_{y \in \mathcal{P}_j} sim(x, y), \tag{6.3}$$

where $sim(x, y)$ is defined in Eq 6.7.

Then, $S(\mathcal{P}_j, \mathcal{P}_i)$ is computed in the same way by swapping $\mathcal{P}_i$ and $\mathcal{P}_j$. Finally, the overall similarity between the two proteins is given by:

$$SS(\mathcal{P}_i, \mathcal{P}_j) = \frac{S(\mathcal{P}_i, \mathcal{P}_j) + S(\mathcal{P}_j, \mathcal{P}_i)}{S(\mathcal{P}_i, \mathcal{P}_i) + S(\mathcal{P}_j, \mathcal{P}_j)}. \tag{6.4}$$

Thus, for a testing protein $\mathbf{Q}_t$, a GO SS vector $\mathbf{q}_t$ can be formulated by performing pariwise comparisons with every training protein $\{\mathbb{P}_i\}_{i=1}^{N}$, where N is the number of training proteins. Then, $\mathbf{q}_t$ can be represented as:

$$\mathbf{q}_t = [SS(\mathcal{Q}_t, \mathcal{P}_1), \cdots, SS(\mathcal{Q}_t, \mathcal{P}_i), \cdots, SS(\mathcal{Q}_t, \mathcal{P}_N)]^{\mathsf{T}}, \tag{6.5}$$

where $\mathcal{Q}_t$ is the set of GO terms for the test protein $\mathbf{Q}_t$.

6.3 HybridGO-Loc: Hybridizing GO Frequency and Semantic Similarity Features

This section proposes a multi-label subcellular-localization predictor, namely HybridGO-Loc, that leverages not only the GO term occurrences but also the inter-term relationships.

[1]Strictly speaking, $\mathcal{P}_i$ should be $\mathcal{P}_{i,k_i}$, where k_i is the k_i-th homolog used to retrieve the GO terms in Section 4.1.2 for the i-th protein. To simplify notations, we write it as $\mathcal{P}_i$.

This is achieved by hybridizing the GO frequencies of occurrences specified in Section 4.1 and the semantic similarity between GO terms specified in Section 6.2. Given a protein, a set of GO terms are retrieved by searching against the gene ontology database, using the accession numbers of homologous proteins obtained via BLAST search as the keys. The frequency of GO occurrences and semantic similarity (SS) between GO terms are used to formulate frequency vectors and semantic similarity vectors, respectively, which are subsequently hybridized to construct fusion vectors. An adaptive-decision based multi-label support vector machine (SVM) classifier is proposed to classify the fusion vectors. Experimental results based on recent benchmark datasets and a new dataset containing novel proteins show that the proposed hybrid-feature predictor significantly outperforms predictors based on individual GO features as well as other state-of-the-art predictors.

6.3.1 Semantic Similarity Features

A semantic similarity measure was introduced in Section 6.2.1. To compare different kinds of semantic similarity measures, the most commonly used measures are specified here. Specifically, the semantic similarity between two GO terms x and y is defined as [173]:

$$sim(x, y) = \max_{c \in A(x,y)}[-\log(p(c))], \tag{6.6}$$

where $A(x, y)$ is the set of ancestor GO terms of both x and y, and $p(c)$ is the probability of the number of gene products annotated to the GO term c divided by the total number of gene products annotated in the GO taxonomy.

While Resnik's measure is effective in quantifying the shared information between two GO terms, it ignores the distance between the terms and their common ancestors in the GO hierarchy. To further incorporate structural information from the GO hierarchy into the similarity measure, we have explored three extension of Resnik's measure, namely

Lin's measure [174], Jiang's measure [192], and relevance similarity (RS) [175].

Given two GO terms x and y, the similarity by Lin's measure is:

$$sim_{Lin}(x,y) \equiv sim_1(x,y) = \max_{c \in A(x,y)} \left(\frac{2 \cdot [-\log(p(c))]}{-\log(p(x)) - \log(p(y))} \right) \tag{6.7}$$

The similarity by Jiang's measure is:

$$\begin{aligned} sim_{Jiang}(x,y) &\equiv sim_2(x,y) \\ &= \max_{c \in A(x,y)} \left(\frac{1}{1 - \log(p(x)) - \log(p(y)) + 2 \cdot [-\log(p(c))]} \right) \end{aligned} \tag{6.8}$$

The similarity by RS is calculated as:

$$sim_{RS}(x,y) \equiv sim_3(x,y) = \max_{c \in A(x,y)} \left(\frac{2 \cdot [-\log(p(c))]}{-\log(p(x)) - \log(p(y))} \cdot (1 - p(c)) \right) \tag{6.9}$$

Among the three measures, $sim_{Lin}(x,y)$ and $sim_{Jiang}(x,y)$ are relative measures that are proportional to the difference in information content between the terms and their common ancestors, which is independent of the absolute information content of the ancestors. On the other hand, $sim_{RS}(x,y)$ incorporates the probability of annotating the common ancestors as a weighing factor to Lin's measure. To simplify notations, we refer $sim_{Lin}(x,y)$, $sim_{Jiang}(x,y)$ and $sim_{RS}(x,y)$ as $sim_1(x,y)$, $sim_2(x,y)$ and $sim_3(x,y)$, respectively.

Based on the semantic similarity between two GO terms, we adopted a continuous measure proposed in [172] to calculate the similarity between two proteins. Specifically, given two proteins $\mathbb{P}_i$ and $\mathbb{P}_j$, we retrieved their corresponding GO terms $\mathcal{P}_i$ and $\mathcal{P}_j$ as described in Section 4.1.2.[2] Then, we computed the semantic similarity between two sets of GO terms $\{\mathcal{P}_i, \mathcal{P}_j\}$ as follows:

$$S_l(\mathcal{P}_i, \mathcal{P}_j) = \sum_{x \in \mathcal{P}_i} \max_{y \in \mathcal{P}_j} sim_l(x,y), \tag{6.10}$$

[2]Strictly speaking, $\mathcal{P}_i$ should be $\mathcal{P}_{i,k_i}$, where k_i is the k_i-th homolog used to retrieve the GO terms in Section 4.1.2 for the i-th protein. To simplify notations, we write it as $\mathcal{P}_i$.

where $l \in \{1,2,3\}$, and $sim_l(x,y)$ is defined in Eq. 6.7 to Eq. 6.9. $S_l(\mathcal{P}_j, \mathcal{P}_i)$ is computed in the same way by swapping $\mathcal{P}_i$ and $\mathcal{P}_j$. Finally, the overall similarity between the two proteins is given by:

$$\mathrm{SS}_l(\mathcal{P}_i, \mathcal{P}_j) = \frac{S_l(\mathcal{P}_i, \mathcal{P}_j) + S_l(\mathcal{P}_j, \mathcal{P}_i)}{S_l(\mathcal{P}_i, \mathcal{P}_i) + S_l(\mathcal{P}_j, \mathcal{P}_j)}, \tag{6.11}$$

where $l \in \{1,2,3\}$. In the sequel, we refer the SS measures by Lin, Jiang and RS to as SS1, SS2 and SS3, respectively.

Thus, for a testing protein $\mathbb{Q}_t$ with GO term set $\mathcal{Q}_t$, a GO semantic similarity (SS) vector $\mathbf{q}_t^{S_l}$ can be obtained by computing the semantic similarity between $\mathcal{Q}_t$ and each of the training protein $\{\mathbb{P}_i\}_{i=1}^N$, where N is the number of training proteins. Thus, $\mathbb{Q}_t$ can be represented by an N-dimensional vector:

$$\mathbf{q}_t^{S_l} = [\mathrm{SS}_l(\mathcal{Q}_t, \mathcal{P}_1), \cdots, \mathrm{SS}_l(\mathcal{Q}_t, \mathcal{P}_i), \cdots, \mathrm{SS}_l(\mathcal{Q}_t, \mathcal{P}_N)]^\mathsf{T}, \tag{6.12}$$

where $l \in \{1,2,3\}$. In other words, $\mathbf{q}_t^{S_l}$ represents the SS vector by using the l-th SS measure.

6.3.2 Hybridization of Two GO Features

As can be seen from Section 4.1.3 and Section 6.3.1, we know that the GO frequency features (Eq. 4.3) use the frequency of occurrences of GO terms, while GO SS features (Eq. 6.7 to Eq. 6.9) use the semantic similarity between GO terms. These two features are developed from two different perspectives. It is therefore reasonable to believe that these two kinds of information complement each other. Based on this assumption, we combine these two GO features and form a hybridized vector as:

$$\mathbf{q}_t^{H_l} = \begin{bmatrix} \mathbf{q}_t^F \\ \\ \mathbf{q}_t^{S_l} \end{bmatrix}, \tag{6.13}$$

where $l \in \{1, 2, 3\}$. In other words, $\mathbf{q}_t^{H_l}$ represents the hybridizing-feature vector by combining the GO frequency features and the SS features derived from the l-th SS measure. We refer them to as *Hybrid1*, *Hybrid2* and *Hybrid3*, respectively.

6.3.3 Multi-label Multi-class SVM Classification

The hybridized-feature vectors obtained in Section 6.3.2 are used for training multi-label one-vs-rest support vector machines (SVMs). Specifically, for an M-class problem (here M is the number of subcellular locations), M independent binary SVMs are trained, one for each class. Denote the hybrid GO vectors of the t-th query protein as $\mathbf{q}_t^{H_l}$, where the l-th SS measure is used in Section 6.3.1. Given the t-th query protein $\mathbb{Q}_t$, the score of the m-th SVM using the l-th SS measure is

$$s_{m,l}(\mathbb{Q}_t) = \sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r^{H_l}, \mathbf{q}_t^{H_l}) + b_m \tag{6.14}$$

where $\mathbf{q}_t^{H_l}$ is the hybrid GO vector derived from $\mathbb{Q}_t$ (See Eq. 6.13), $\mathcal{S}_{m,l}$ is the set of support vector indexes corresponding to the m-th SVM, $\alpha_{m,r}$ are the Lagrange multipliers, $y_{m,r} \in \{-1, +1\}$ indicates whether the r-th training protein belongs to the m-th class or not, and $K(\cdot, \cdot)$ is a kernel function. Here, the linear kernel was used.

Unlike the single-label problem where each protein has one predicted label only, a multi-label protein could have more than one predicted labels.

In this work, we compared the two decision schemes introduced in Section 5.3 and Section 5.4 for this multi-label problem. In the first scheme, the predicted subcellular location(s) of the i-th query protein are given by

$$\mathcal{M}_l^*(\mathbb{Q}_t) = \begin{cases} \bigcup_{m=1}^{M} \{m : s_{m,l}(\mathbb{Q}_t) > 0\}, & \text{when } \exists\ m \in \{1, \ldots, M\} \text{ s.t. } s_{m,l}(\mathbb{Q}_t) > 0; \\ \arg\max_{m=1}^{M} s_{m,l}(\mathbb{Q}_t), & \text{otherwise.} \end{cases} \tag{6.15}$$

The second scheme is an improved version of the first one in that the decision threshold is dependent on the test protein. Specifically, the predicted subcellular location(s) of the i-th query protein are given by:

If $\exists\ s_{m,l}(\mathbb{Q}_t) > 0$,

$$\mathcal{M}_l(\mathbb{Q}_t) = \bigcup_{m=1}^{M} \left(m : s_{m,l}(\mathbb{Q}_t) \geq \min\{1.0, f(s_{\max,l}(\mathbb{Q}_t))\}\right) \tag{6.16}$$

otherwise,

$$\mathcal{M}(\mathbb{Q}_t) = \arg\max_{m=1}^{M} s_{m,l}(\mathbb{Q}_t). \tag{6.17}$$

In Eq. 6.16, $f(s_{\max,l}(\mathbb{Q}_t))$ is a function of $s_{\max,l}(\mathbb{Q}_t)$, where $s_{\max,l}(\mathbb{Q}_t) = \max_{m=1}^{M} s_{m,l}(\mathbb{Q}_t)$. In this work, we used a linear function as follows:

$$f(s_{\max,l}(\mathbb{Q}_t)) = \theta s_{\max,l}(\mathbb{Q}_t), \tag{6.18}$$

where $\theta \in [0.0, 1.0]$ is a hyper-parameter that can be optimized through cross-validation.

In fact, besides SVMs, many other machine learning models, such as hidden Markov models (HMMs) and neural networks (NNs) [193, 194], have been used in protein subcellular-localization predictors. However, HMMs and NNs are not suitable for GO-based predictors because of the high dimensionality of GO vectors. The main reason is that under such condition, HMMs and NNs can be easily overtrained and thus lead to poor performance. On the other hand, linear SVMs can well handle high-dimensional data because even if the number of training samples is smaller than the feature dimension, linear SVMs are still able to find an optimal solution.

6.4 Summary

This chapter presents two predictors, namely SS-Loc and HybridGO-Loc, both of which extract deeper GO information, i.e. GO semantic similarity, for multi-location protein subcellular localization.

SS-Loc extends the inter-term relationship of GO terms to inter-group relationship of GO term groups, which are used to represent the similarity between proteins. And then the similarity vectors are predicted by multi-label SVM classifiers. Based on this, HybridGO-Loc combines the features of GO occurrences and GO semantic similarity to generate hybrid feature vectors. Several semantic similarity measures have been compared and two different decision schemes are tried. The superior performance of HybridGO-Loc (See Chapter 9) also demonstrates that these two features are complementary to each other.

Chapter 7

Ensemble Random Projection for Large-Scale Predictions

Typically, GO-based methods extract as many as thousands of GO terms to formulate GO vectors. The curse of dimensionality severely restricts the predictive power of GO-based multi-label classification systems. Besides, high-dimensional feature vectors may contain redundant or irrelevant information, causing the classification systems suffer from overfitting. To address this problem, this chapter presents a dimensionality-reduction method that applies random projection (RP) to construct an ensemble of multi-label classifiers. After feature extraction, the GO vectors are then projected onto lower-dimensional spaces by random projection matrices whose elements conform to a distribution with zero mean and unit variance. The transformed low-dimensional vectors are classified by an ensemble of one-vs-rest multi-label classifiers, each corresponding to one of the RP matrices. The scores obtained from the ensemble are then fused for predicting the subcellular localization of proteins.

Random projection-related works are first introduced in this chapter. Then, two multi-label classifiers using ensemble RP are presented, namely RP-SVM and R3P-Loc, one for multi-label SVM classifiers and one for multi-label ridge regression classifiers. Moreover, to further reduce redundant information and extract useful information, two compact databases are created for the R3P-Loc predictor.

7.1 Related Work

In machine learning, high-dimensional patterns are often mapped to a lower dimension subspace to avoid the curse of dimensionality [195]. Reducing the dimension of the input patterns can remove redundant or irrelevant information and allow for more reliable classification in the subspace. Actually, dimension reduction are imperative in various domains, such as text categorization [196], image retrieval [197] and gene expression microarray data analysis [198].

In the past three decades, random projection (RP) has emerged as a powerful method for dimension reduction. By using RP, the high dimensional feature vectors are transformed into a much lower-dimensional vectors, which preserve the original geometrical structure and contain less redundant, irrelevant or even detrimental information that might deteriorate classification performance. RP turns out to be a computationally efficient, yet sufficiently accurate method for dimensionality reduction of many high-dimensional datasets [199]. RP is particularly useful for sparse input data in high dimensions as the original data can be reconstructed almost perfectly from data in the lower-dimensional projected space [200]. RP has been widely used in various applications, such as preprocessing text data [201], indexing audio documents [202], processing images [199], learning high-dimensional Gaussian mixture models [203]. Recently, dynamic random projection [204, 205] is successfully applied in biometric template protection and privacy-preserving verification.

As stated in Chapter 2, conventional methods for subcellular-localization prediction can be roughly divided into sequence-based methods and knowledge-based methods. However, no matter using the sequence-based features or annotation-based features, a predominant scenario is that the dimension of available features is much larger than the number of training samples. For example, Lee et. al. [22] used amino-acid sequence-based features, whose dimension (11,992-dim) is remarkably larger than the number of proteins (3017); Xiao et. al. [93] used the Gene Ontology (GO)[1] information as the features and the dimension of features was 11,118 while the number of proteins was only 207. It is highly expected that the high-dimensional features contain redundant or irrelevant information, causing overfitting and worsening the prediction performance.

7.2 RP-SVM: A Multi-Label Classifier with Ensemble Random Projection

This section proposes an ensemble SVM classifier based on random projection (RP), namely RP-SVM, for predicting subcellular localization of multi-label proteins. By using RP, the high dimensional feature vectors are transformed into a much lower-dimensional vectors, which contain less redundant, irrelevant or even detrimental information that might deteriorate classification performance. To make the classifiers more robust, it is necessary to perform random projection of the feature vectors a number of times, each with a different projection matrix. The resulting projected vectors are then presented to an ensemble of one-vs-rest multi-label SVM classifiers.

7.2.1 Random Projection

The key idea of RP arises from Johnson-Lindenstrauss lemma [206]:

Lemma 1. (Johnson and Lindenstrauss [206]). *Given $\epsilon > 0$, a set $\mathcal{X}$ of N points in $\mathcal{R}^T$, and a positive integer d such that $d \geq d_0 = \mathcal{O}(\log N/\epsilon^2)$, there exists $f : \mathcal{R}^T \rightarrow \mathcal{R}^d$ such*

[1]http://www.geneontology.org

that

$$(1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

for all $u, v \in \mathcal{X}$.

The lemma suggests that if points in a high-dimensional space are projected onto a randomly selected subspace of suitable dimension, the distances between the points are approximately preserved. A proof can be found in [207].

Specifically, the original T-dimensional data is projected onto a d-dimensional ($d \ll T$) subspace, using a $d \times T$ random matrix $\mathbf{R}$ whose columns are of unit length. In our case, for the i-th protein, the GO vector $\mathbf{q}_i$ (Eq. 4.3 in Section 4.1.3 of Chapter 4) can be projected as:

$$\mathbf{q}_i^{RP} = \frac{1}{\sqrt{d}}\mathbf{R}\mathbf{q}_i, \tag{7.1}$$

where $1/\sqrt{d}$ is a scaling factor, $\mathbf{q}_i^{RP}$ is the projected vector after RP, and $\mathbf{R}$ is a random $d \times T$ matrix.

The choice of the random matrix $\mathbf{R}$ has been studied extensively. Practically, as long as the elements $r_{h,j}$ of $\mathbf{R}$ conforms to any distribution with zero mean and unit variance, $\mathbf{R}$ will give a mapping that satisfies the Johnson-Lindenstrauss lemma [199]. For computational simplicity, we adopted a simple distribution proposed by Achlioptas [208] for the elements $r_{h,j}$ as follows:

$$r_{h,j} = \sqrt{3} \times \begin{cases} +1 & \text{with probability } 1/6, \\ 0 & \text{with probability } 2/3, \\ -1 & \text{with probability } 1/6. \end{cases} \tag{7.2}$$

It is easy to verify that Eq. 7.2 conforms to a distribution with zero mean and unit variance [208].

7.2.2 Ensemble Multi-label Classifier

The projected GO vectors obtained from Eq. 7.1 are used for training multi-label one-vs-rest SVMs. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent binary SVMs are trained, one for each class. Denote the GO vector created by using the true AC of the i-th query protein as $\mathbf{q}_{i,0}$ and the GO vector created by using the accession number of the k-th homolog as $\mathbf{q}_{i,k}$, $k \in \{1, \ldots, k_{\max}\}$, where $k_{\max}$ is the number of homologs retrieved by BLAST with the default parameter setting. By Eq. 7.1, we obtained the corresponding projected vectors $\mathbf{q}_{i,0}^{RP}$ and $\mathbf{q}_{i,k}^{RP}$, respectively. Then, given the i-th query protein $\mathbb{Q}_i$, the score of the m-th SVM is:

$$s_m(\mathbb{Q}_i) = \sum_{r \in \mathcal{S}_m} \alpha_{m,r} y_{m,r} K(\mathbf{p}_r^{RP}, \mathbf{q}_{i,h}^{RP}) + b_m \tag{7.3}$$

where

$$h = \min\{\arg(|\mathbf{q}_{i,k}^{RP}| \neq 0)\}, k = 0, \ldots, k_{\max}, \tag{7.4}$$

and $\mathcal{S}_m$ is the set of support vector indexes corresponding to the m-th SVM, $y_{m,r} \in \{-1, +1\}$ are the class labels, $\alpha_{m,r}$ are the Lagrange multipliers, $K(\cdot, \cdot)$ is a kernel function; here, the linear kernel is used. Note that $\mathbf{p}_r^{RP}$'s in Eq. 7.3 represents the projected GO training vectors, which may include the projected GO vectors created by using the true AC of the training sequences or their homologous ACs.

Since $\mathbf{R}$ is a random matrix, the scores in Eq. 7.3 for each application of RP will be different. To construct a robust classifier, we fused the scores for several applications of RP and obtained an ensemble classifier, whose ensemble score of the m-th SVM for the i-th query protein is given as follows:

$$s_m^{en}(\mathbb{Q}_i) = \sum_{l=1}^{L} w_l \cdot s_m^{(l)}(\mathbb{Q}_i), \tag{7.5}$$

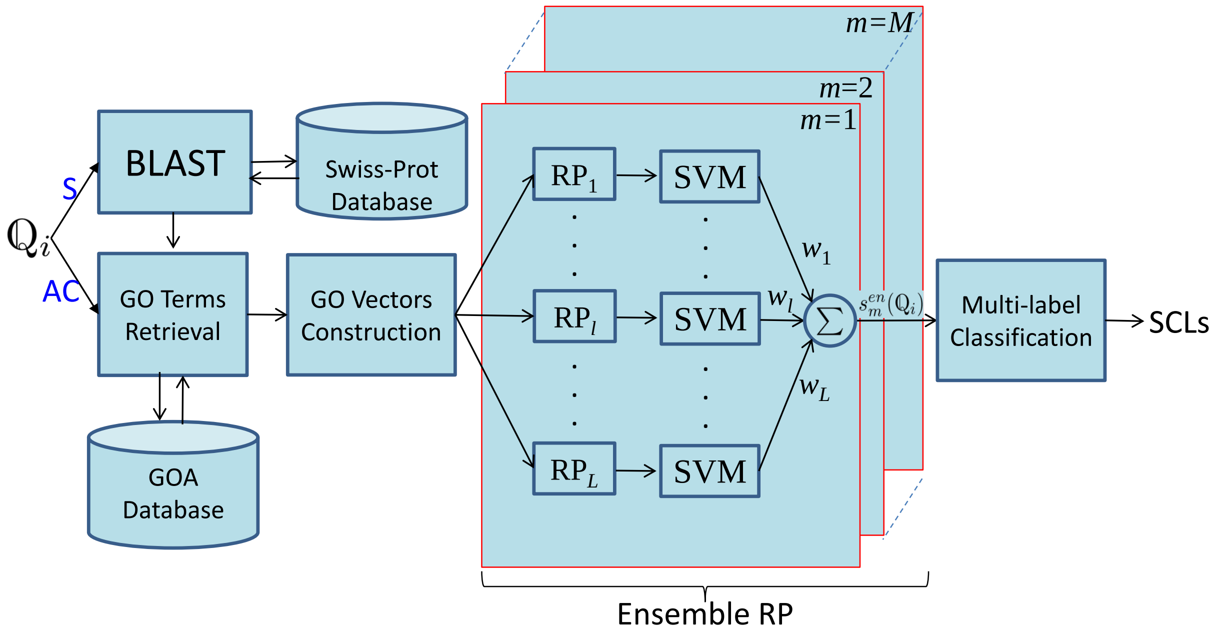


Figure 7.1: Flowchart of RP-SVM/RP-AD-SVM. $\mathbb{Q}_i$: the i-th query protein; S: protein sequence; AC: protein accession number; RP: random projection; SVM: SVM scoring (Eq. 7.3); *Ensemble RP*: ensemble random projection; w_1, w_l *and* w_L: the 1-st, l-th and L-th weights in Eq. 7.5; $s_m^{en}(\mathbb{Q}_i)$: the ensemble score in Eq. 7.5; *SCLs*: subcellular location(s).

where $\sum_{l=1}^{L} w_l = 1$, $s_m^{(l)}(\mathbb{Q}_i)$ represents the score of the m-th SVM for the i-th protein via the l-th application of RP, L is the total number of applications of RP, and $\{w_l\}_{l=1}^{L}$ are the weights. For simplicity, here we set $w_l = 1/L, l = 1, \ldots, L$. We refer L as 'ensemble size' in the sequel. Unless stated otherwise, the ensemble size was set to 10 in our experiments, i.e., $L = 10$. Note that instead of mapping the original data into an Ld-dim vector, the ensemble RP projects it into L d-dim vectors.

7.2.3 Multi-label Classification

To predict the subcellular locations of datasets containing both single-label and multi-label proteins, a decision scheme for multi-label SVM classifiers should be used. Unlike the single-label problem where each protein has one predicted label only, a multi-label protein should have more than one predicted labels. In this work, we evaluated two decision schemes. The first decision scheme is the same as that used in mGOASVM (Section 5.3 in Chapter 5). In this scheme, the predicted subcellular location(s) of the i-th query protein are given by:

$$\mathcal{M}^*(\mathbb{Q}_i) = \begin{cases} \bigcup_{m=1}^{M} \{m : s_m^{en}(\mathbb{Q}_i) > 0\}, \text{ where } \exists\ s_m^{en}(\mathbb{Q}_i) > 0\ ; \\ \arg\max_{m=1}^{M} s_m^{en}(\mathbb{Q}_i), \text{ otherwise.} \end{cases} \tag{7.6}$$

The second decision scheme is an adaptive decision improved upon the first one. This decision scheme is the same as that used in AD-SVM (Section 5.4 in Chapter 5). In this scheme, the predicted subcellular location(s) of the i-th query protein are given by:
If $\exists\ s_m^{en}(\mathbb{Q}_i) > 0$,

$$\mathcal{M}(\mathbb{Q}_i) = \bigcup_{m=1}^{M} \{\{m : s_m^{en}(\mathbb{Q}_i) > 1.0\} \cup \{m : s_m^{en}(\mathbb{Q}_i) \geq f(s_{\max}(\mathbb{Q}_i))\}\} \tag{7.7}$$

otherwise,

$$\mathcal{M}(\mathbb{Q}_i) = \arg\max_{m=1}^{M} s_m^{en}(\mathbb{Q}_i). \tag{7.8}$$

In Eq. 7.7, $f(s_{\max}(\mathbb{Q}_i))$ is a function of $s_{\max}(\mathbb{Q}_i)$, where $s_{\max}(\mathbb{Q}_i) = \max_{m=1}^{M} s_m^{en}(\mathbb{Q}_i)$. In this work, we used a linear function as follows:

$$f(s_{\max}(\mathbb{Q}_i)) = \theta s_{\max}(\mathbb{Q}_i), \tag{7.9}$$

where $\theta \in [0.0, 1.0]$ is a parameter that was optimized to achieve the best performance.

For ease of comparison, we refer to the proposed ensemble classifier with the first and the second decision scheme as RP-SVM and RP-AD-SVM, respectively. Fig. 7.1 illustrates the whole prediction process for RP-SVM and RP-AD-SVM. If we use the first decision scheme for *multi-label classification*, the diagram represents RP-SVM; if we use the second decision scheme, it represents RP-AD-SVM.

7.3 R3P-Loc: A Compact Predictor Based on Ridge Regression and Ensemble Random Projection

So far, we have presented many GO-based predictors which extract GO information from a knowledge database, i.e., Gene Ontology annotation (GOA) database. However, the predominant scenarios of GO-based methods are that (1) the GOA database has enormous size and are growing exponentially, (2) the GOA database contains redundant information, and (3) the number of extracted features from the GOA database is much larger than the number of data samples with ground-truth labels. These properties render the extracted features liable to redundant or irrelevant information, causing the prediction systems suffer from overfitting. To address these problems, this section proposes an efficient multi-label predictor, namely R3P-Loc, which uses two compact databases for feature extraction and applies random projection (RP) to reduce the feature dimensions of an ensemble ridge regression (RR) classifier. Two new compact databases are created from Swiss-Prot and GOA databases. These databases possess almost the same amount of information as their full-size counterparts but with much smaller size. Experimental results on two recent datasets (eukaryote and plant) suggest that R3P-Loc can reduce the dimensions by seven folds and significantly outperforms state-of-the-art predictors. It is also found that the compact databases reduce the memory consumption by 39 times without causing degradation in prediction accuracy.

This section is organized as follows. First, the limitation of using the knowledge databases is introduced. Then, the procedure of creating the two compact databases (i.e., ProSeq and ProSeq-GO) is specified. Subsequently, multi-label ridge regression classifiers equipped with random projection is presented.

7.3.1 Limitation of Using Current Databases

Recently, several state-of-the-art multi-label predictors have been proposed, such as Plant-mPLoc [96], Euk-mPLoc 2.0 [158], iLoc-Plant [87], iLoc-Euk [90], mGOASVM [108], HybridGO-Loc [209] and other predictors [210, 211, 212]. They all use the GO information as the features and apply different multi-label classifiers to tackle the multi-label classification problem. However, these GO-based methods are not without disadvantages. Currently the predominant scenarios of GO-based methods are that:

1. The gene ontology annotation (GOA) database,[2] from which these GO-based predictors extract the GO information for classification, is usually in enormous size and is also growing rapidly. For example, in October 2005, the GOA database contains 7,782,748 entries for protein annotations; in March 2011, GOA database contains 82,632,215 entries; and in July 2013, the number of entries increases to 169,603,862, which suggests that in less than 8 years, the number of annotations in GOA database increases 28 times. Even after compressing the GOA database released in July 2013 by removing the repeated pairing of accession numbers (ACs) and GO terms, the number of distinct pairs of AC–GO terms is still as high as 25,441,543. It is expected that searching a database with such a enormous and rapidly-growing size is computationally prohibitive, which makes large-scale subcellular localization by GO-based methods inefficient and even intractable.

2. The GOA database contains many redundant AC entries that will never be used by typical GO-based methods. This is because given a query protein, GO-based methods search for homologous ACs from Swiss-Prot and use these ACs as keys to search against the GOA database for retrieving relevant GO terms. Therefore,

[2]http://www.ebi.ac.uk/GOA

those ACs in the GOA database that do not appear in Swiss-Prot are redundant. Among all the ACs in the GOA database, more than 90% are in this category. This calls for a more compact GO-term database that excludes these redundant entries.

3. The number of extracted GO features from the GOA database is much larger than the number of proteins that are relevant to the prediction task. For example, Xiao et. al. [93] extracted GO information of 207 proteins from the GOA database; the resulting feature vectors have 11,118 dimensions, which suggests that the number of features is more than 50 times the number of proteins. It is likely that among the large number of features, many of them contain redundant or irrelevant information, causing the prediction systems suffer from overfitting and thus degrading the prediction performance.

To tackle the problems mentioned above, this section proposes an efficient and compact multi-label predictor, namely **R3P-Loc**, which uses **R**idge **R**egression and **R**andom **P**rojection for predicting subcellular **Loc**alization of both single-label and multi-label proteins. Instead of using the Swiss-Prot and GOA databases, R3P-Loc uses two newly-created compact databases, namely ProSeq and ProSeq-GO, for GO information transfer. The ProSeq database is a sequence database in which each amino acid sequence has at least one GO term annotated to it. The ProSeq-GO comprises GO terms annotated to the protein sequences in the ProSeq database. An important property of the ProSeq and ProSeq-GO databases is that they are much smaller than the Swiss-Prot and GOA databases, respectively.

Given a query protein, a set of GO-terms are retrieved by searching against the ProSeq-GO database using the accession numbers of homologous proteins as the searching keys, where the homologous proteins are obtained from BLAST searches, using ProSeq as the sequence database. The frequencies of GO occurrences are used to formulate frequency vectors, which are projected onto much lower-dimensional space by random matrices whose elements conform to a distribution with zero mean and unit variance. Subsequently, the dimension-reduced feature vectors are classified by a multi-label ridge regression classifier.

7.3.2 Creating Compact Databases

Typically, for a query protein, an efficient predictor should be able to deal with two possible cases: (1) the accession number (AC) is known and (2) only the amino acid sequence is known. For proteins with known ACs, their respective GO terms are retrieved from a database containing GO terms (i.e., GOA database) using the ACs as the searching keys. For a protein without an AC, its amino acid sequence is presented to BLAST [60] to find its homologs against a database containing protein amino acid sequences (i.e., Swiss-Prot), whose ACs are then used as keys to search against the GO-term database.

While the GOA database allows us to associate the AC of a protein with a set of GO terms, for some novel proteins, neither their ACs nor the ACs of their top homologs have any entries in the GOA database; in other words, no GO terms can be retrieved by their ACs or the ACs of their top homologs. In such case, some predictors use back-up methods that rely on other features, such as pseudo-amino-acid composition [29] and sorting signals [42]; some predictors [102, 108] use a successive-search strategy to avoid null GO vectors. However, these strategies may lead to poor performance and increase computation and storage complexity.

To address this problem, we created two small yet efficient databases: ProSeq and ProSeq-GO. The former is a sequence database and the latter is a GO-term database. The procedures of creating these databases are shown in Fig. 7.2. The procedure extracts accession numbers from two different sources: Swiss-Prot and GOA database. Specifically, all of the ACs in the Swiss-Prot database and the *valid* ACs in the GOA database are extracted. Here, an AC is considered valid when it has at least one GO term annotated to it. Then, the common ACs that appear in both sets are selected (the $\bigcap$ symbol in Fig. 7.2). These ACs are regarded as 'valid Swiss-Prot ACs'; each of them corresponds to at least one GO term in the GOA database. Next, using these valid ACs, their corresponding amino-acid sequences can be retrieved from the Swiss-Prot database, constituting a new sequence database, which we call 'ProSeq database'; similarly, using these valid ACs, their corresponding GO terms can be retrieved from the GOA database, constituting a new GO-term database, which we call 'ProSeq-GO database'. In this work, we created ProSeq and ProSeq-GO databases from the Swiss-Prot and GOA databases released in July 2013. The ProSeq-GO database has 513,513 entries while the GOA database has 25,441,543 entries; the ProSeq database has 513,513 protein sequences while the Swiss-Prot database has 540,732 protein sequences.

Similar to mGOASVM (Section 5.3 in Chapter 5), R3P-Loc uses GO frequencies as the feature information and the feature extraction part is the same as mGOASVM. After feature extraction, similar to RP-SVM, random projection is applied to the GO vectors (Eq. 7.1 in Section 7.2). Then, an ensemble multi-label ridge regression classifier is proposed for classification of the dimension-reduced vectors.

7.3.3 Single-Label Ridge Regression

Ridge regression (RR) is a simple yet effective linear regression model, which has been applied to many domains [213, 214, 215]. Here we apply RR into classification. Suppose for a two-class single-label problem, we are given a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathcal{R}^{T+1}$ and $y_i \in \{0, 1\}$. In our case, $\mathbf{x}_i = \begin{bmatrix} 1 \\ \mathbf{q}_i^{RP} \end{bmatrix}$, where $\mathbf{q}_i^{RP}$ is defined in Eq. 7.1.

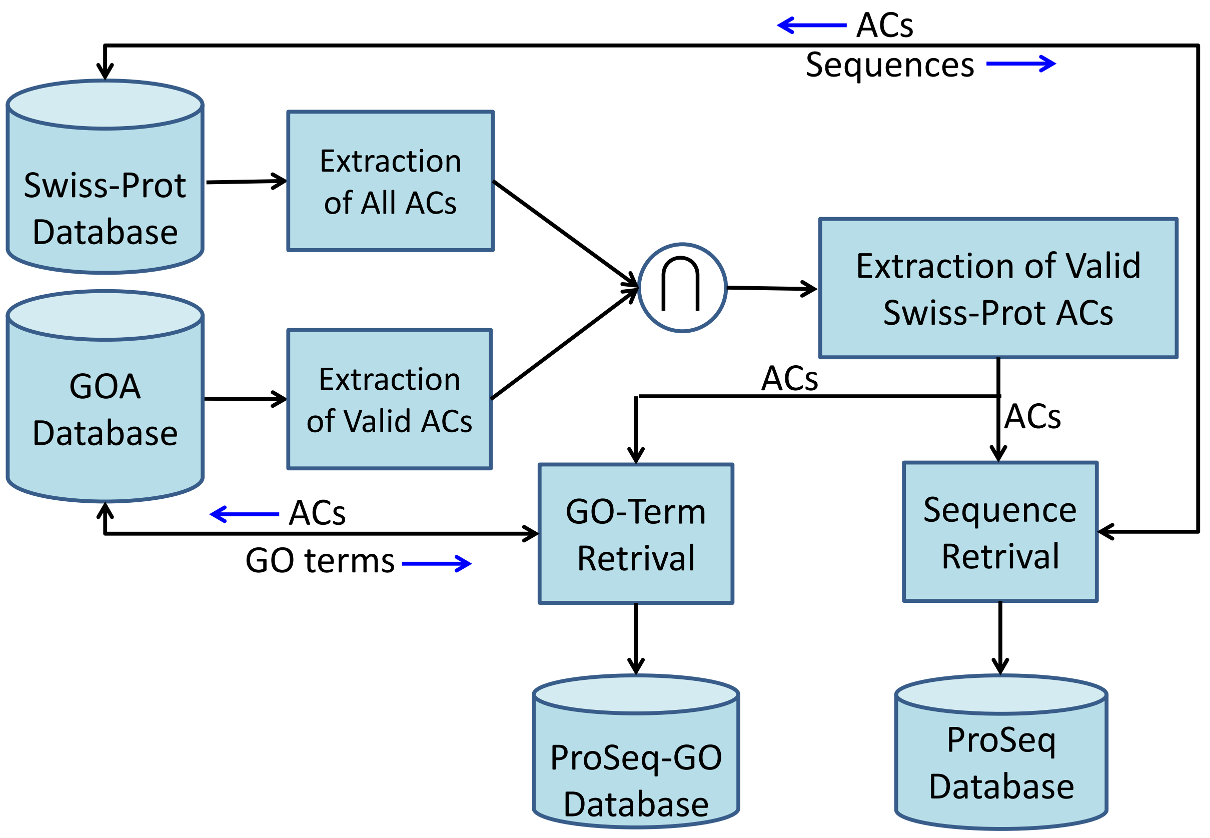


Figure 7.2: Procedures of creating compact databases (ProSeq and ProSeq-GO). *AC*: accession numbers; *GO*: gene ontology; *GOA database*: gene ontology annotation database.

Generally speaking, an RR model is to impose an L_2-style regularization to ordinary least squares (OLS), namely minimizing the empirical loss $l(\boldsymbol{\beta})$ as:

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 = \sum_{i=1}^{N}(y_i - \sum_{j=1}^{T+1}\beta_j x_{i,j})^2, \tag{7.10}$$

subject to

$$\sum_{j=1}^{T+1}\beta_j^2 \leq s,$$

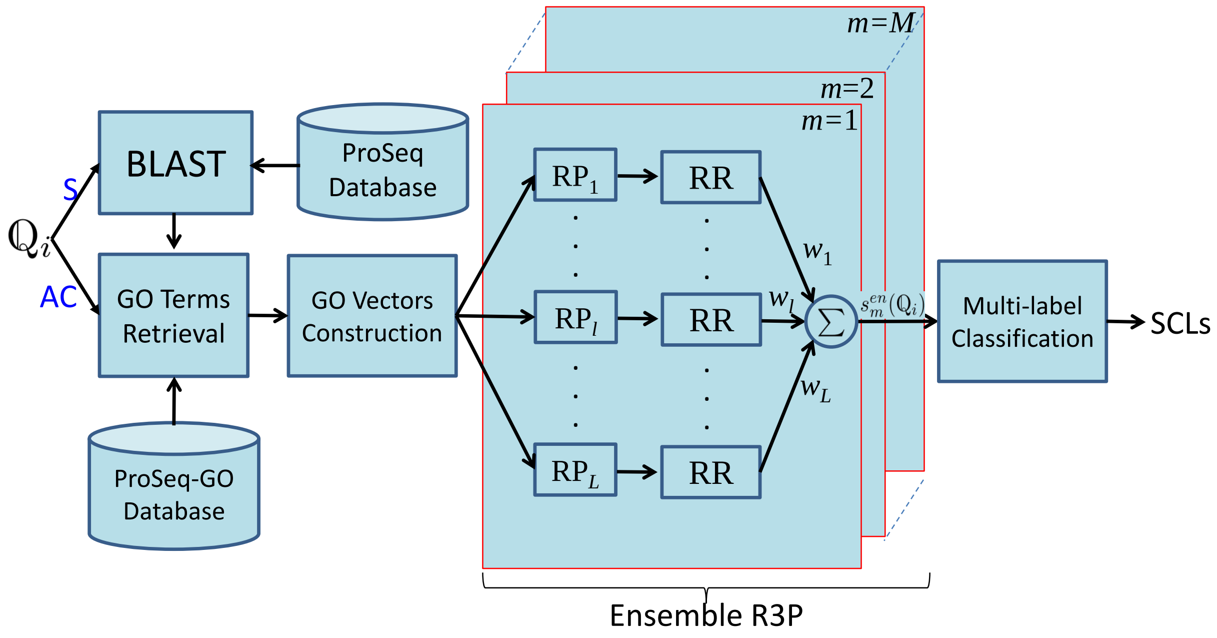


Figure 7.3: Flowchart of R3P-Loc. $\mathbb{Q}_i$: the i-th query protein; *S*: protein sequence; *AC*: protein accession number; *ProSeq/ProSeq-GO*: the proposed compact sequence and GO databases, respectively; *RP*: random projection; *RR*: ridge regression scoring (Eq. 7.14); *Ensemble R3P*: ensemble ridge regression and random projection; w_1, w_l *and* w_L: the 1-st, l-th and L-th weights in Eq. 7.15; $s_m^{en}(\mathbb{Q}_i)$: the ensemble score in Eq. 7.15; *SCLs*: subcellular location(s).

where $s > 0$, $x_{i,j}$ is the j-th element of $\mathbf{x}_i$ and $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_j, \ldots, \beta_{T+1}]^\mathsf{T}$ is the ridge vector to be optimized. Eq. 7.10 is equivalent to minimize the following equation:

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^\mathsf{T}\mathbf{x}_i)^2 + \lambda\boldsymbol{\beta}^\mathsf{T}\boldsymbol{\beta}, \tag{7.11}$$

where $\lambda > 0$ is a penalized parameter to control the degree of regularization. Then after optimization, $\boldsymbol{\beta}$ is given as:

$$\boldsymbol{\beta} = (\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}\mathbf{y}, \tag{7.12}$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N]^\mathsf{T}$,$\mathbf{y} = [y_1, \ldots, y_i, \ldots, y_N]^\mathsf{T}$, and $\mathbf{I}$ is a $(T+1) \times (T+1)$ identity matrix.

7.3.4 Multi-Label Ridge Regression

In an M-class multi-label problem, the training data set is written as $\{\mathbf{x}_i, \mathcal{Y}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathcal{R}^{T+1}$ and $\mathcal{Y}_i \subset \{1, 2, \ldots, M\}$ is a set which may contain one or more labels. M independent binary one-vs-rest RRs are trained, one for each class. The labels $\{\mathcal{Y}_i\}_{i=1}^N$ are converted to *transformed labels* (Similar to Eqs. 5.2–5.3) $y_{i,m} \in \{-1, 1\}$, where $i = 1, \ldots, N$, and $m = 1, \ldots, M$. Then, Eq. 7.12 is extended to:

$$\boldsymbol{\beta}_m = (\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda \mathbf{I})^{-1}\mathbf{X}\mathbf{y}_m, \tag{7.13}$$

where $m = 1, \ldots, M$, $\mathbf{y}_m$ are vectors whose elements are $\{y_{i,m}\}_{i=1}^N$.

The projected GO vectors obtained from Eq. 7.1 are used for training multi-label one-vs-rest ridge regression (RR) classifiers. Specifically, for an M-class problem (here M is the number of subcellular locations), M independent binary RRs are trained, one for each class. Then, given the i-th query protein $\mathbb{Q}_i$, the score of the m-th RR is:

$$s_m(\mathbb{Q}_i) = \boldsymbol{\beta}_m{}^\mathsf{T}\mathbf{x}_i, \text{ where } \mathbf{x}_i = \begin{bmatrix} 1 \\ \mathbf{q}_i^{RP} \end{bmatrix}. \tag{7.14}$$

Since $\mathbf{R}$ is a random matrix, the scores in Eq. 7.14 for each application of RP will be different. To construct a robust classifier, we fused the scores for several applications of RP and obtained an ensemble classifier, whose ensemble score of the m-th SVM for the i-th query protein is given as follows:

$$s_m^{en}(\mathbb{Q}_i) = \sum_{l=1}^{L} w_l \cdot s_m^{(l)}(\mathbb{Q}_i), \tag{7.15}$$

where $\sum_{l=1}^L w_l = 1$, $s_m^{(l)}(\mathbb{Q}_i)$ represents the score of the m-th RR for the i-th protein via the l-th application of RP, L is the total number of applications of RP, and $\{w_l\}_{l=1}^L$ are the weights. For simplicity, here we set $w_l = 1/L, l = 1, \ldots, L$. We refer L as 'ensemble size' in the sequel. Unless stated otherwise, the ensemble size was set to 10 in our experiments, i.e., $L = 10$. Note that instead of mapping the original data into an Ld-dim vector, the ensemble RP projects it into L d-dim vectors.

To predict the subcellular locations of datasets containing both single-label and multi-label proteins, a decision scheme for multi-label RR classifiers should be used. Unlike the single-label problem where each protein has one predicted label only, a multi-label protein should have more than one predicted labels. Here, we used the decision scheme described in mGOASVM (Section 5.3 in Chapter 5) . In this scheme, the predicted subcellular location(s) of the i-th query protein are given by:

$$\mathcal{M}^*(\mathbb{Q}_i) = \begin{cases} \bigcup_{m=1}^{M} \{m : s_m^{en}(\mathbb{Q}_i) > 0\}, \text{ where } \exists\ s_m^{en}(\mathbb{Q}_i) > 0 \ ; \\ \\ \arg\max_{m=1}^{M} s_m^{en}(\mathbb{Q}_i), \text{ otherwise.} \end{cases} \tag{7.16}$$

For ease of comparison, we refer to the proposed ensemble classifier with this multi-label decision scheme as R3P-Loc. The flowchart of R3P-Loc is shown in Fig. 7.3.

7.4 Summary

This chapter focuses on applying ensemble random projection to reduce dimensions of GO vectors and boost the performance of GO-based predictors. Two RP-based predictors are presented, namely RP-SVM and R3P-Loc, one on the multi-label SVM classifier and one on the multi-label RR classifier. Particularly, for R3P-Loc, to further reduce redundant information in the knowledge databases, two compact databases, namely ProSeq and ProSeq-GO are created for fast and efficient GO information extraction.

Chapter 8: Experimental Setup

The experimental setup focuses on how the experiment was designed. This chapter focuses on the setup of two models and is organised in the following way and is very effective partly because the writer includes the following:

Structure

(Introduction)		Not included
Model 1		Section 8.1
Data for model	⇩	
Performance of model		
Model 2	⇩	Section 8.2
Data for model		
Performance of model		
Statistical Methods	⇩	Section 8.3
Summary		Section 8.4

Content

- Gives a short introduction (e.g. Chapter 8, paragraph 1)
- States the theoretical basis for the methodology adopted (e.g. Chapter 8, paragraph 1, sentence 1)
- States the focus of the chapter (e.g. Chapter 8, paragraph 1, sentence 1)
- Gives a short introduction to subsection (e.g. Section 8.1)
- States limitations of the method (e.g. Section 8.1.1.1, paragraph 4, final sentence)
- Cites sources for methodology (e.g. Section 8.1.2)
- Develops paragraphs in a logical way, e.g. Section 8.2.2, paragraph 1:

States topic	sentence 1
Refers to relevant figures	sentence 2
Highlights main feature of figure	sentence 3
Explains main feature	sentence 4
Compares findings	sentence 5
Draws conclusion	sentence 6
Describes other features	sentence 7

- Highlights key figures from a chart (e.g. Section 8.2.1, paragraph 4, sentence 3)

- Develops sections in a logical way, e.g. Section 8.3:

Introduction outlines section and introduces three main points	paragraph 1
Describes first point	paragraph 2
Describes second point	paragraph 3
Describes third point	paragraph 4
Concludes	paragraph 5

Language

- Uses past simple tense and passive voice to describe methodology e.g. *was used* (e.g. Section 1.1.1.1, paragraph 1, sentence 1) *were created* (e.g. Section 1.1.1.1, paragraph 1, sentence 1)
- Refers to charts, figure and equations with a range of grammar e.g. *according to…* (e.g. Section 8.3, paragraph 5, sentence 1), *as can be seen from…* (e.g. Section 8.2.3, paragraph 2, sentence 5), *also shows* (e.g. Section 8.2.2, paragraph 1, sentence 7)

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

💡 Avoid vague language, e.g. *some criteria* (e.g. Section 8.1.1.1, paragraph 1, sentence 6). It is better to use *the following criteria*.

💡 Include a brief introduction for each section (e.g. Section 8.1.1 does not include any information and 8.1.1.1 follows).

💡 Use vertical lists when listing large numbers of items (e.g. Section 8.1.1.1, paragraph 2, sentence 1).

💡 Include only the key information when describing tables in text (e.g. Section 8.1.1.2, paragraph 1, sentence 3).

💡 Link the final paragraph to the next chapter.

Chapter 8

Experimental Setup

As stated in a comprehensive review [112], valid datasets construction for training and testing, and unbiased measurements for evaluating the performance of predictors are two indispensable steps to establish a statistical protein predictor. This chapter will focus on these two important parts for experimental setup. Datasets construction and performance metrics for predicting single-label proteins and multi-label proteins will be presented.

8.1 Prediction of Single-Label Proteins

This section will focus on constructing datasets and introducing performance metrics for single-location proteins, which are used in GOASVM and InterProSVM.

8.1.1 Datasets Construction

8.1.1.1 Datasets for GOASVM

Two benchmark datasets (EU16 [64] and HUM12 [85]) and a novel dataset were used to evaluate the performance of GOASVM. The EU16 dataset and HUM12 dataset were created from Swiss-Prot 48.2 in 2005 and Swiss-Prot 49.3 in 2006, respectively. The EU16 comprises 4150 eukaryotic proteins (2423 in the training set and 1727 in the independent test set) with 16 classes and the HUM12 has 2041 human proteins (919 in the training set and 1122 in the independent test set) with 12 classes. Both datasets were cut off at 25% sequence similarity by a culling program [216]. Here we use the EU16 dataset as an example to illustrate the details of dataset construction procedures. To obtain high-quality, well-defined working datasets, the data were screened strictly according to some criteria described below [64]:

1. Only protein sequences annotated with 'eukaryotic' were included, since the current study only focused on eukaryotic proteins;
2. Sequences annotated with ambiguous or uncertain terms, such as 'probably', 'maybe', 'probable', 'potential', or 'by similarity', were excluded;
3. Those protein sequences labelled with two or more subcellular locations were excluded because of the lack of uniqueness;
4. Sequences annotated with 'fragments' were excluded and also, sequences with less than 50 amino acid residues were removed since these proteins might just be fragments;
5. To avoid any homology bias, the sequence similarity in the same subcellular location among the obtained dataset was cut off at 25% operated by a culling program [216] to winnnow the redundant sequences;
6. Subcellular locations (subsets) containing less than 20 protein sequences were left out because of lacking statistical significance.

After strictly following the criteria mentioned above, only 4150 protein sequences were found, of which there are 25 cell wall, 21 centriole, 258 chloroplast, 97 cyanelle, 718 cytoplasm, 25 cytoskeleton, 113 endoplasmic reticulum, 806 extracellular, 85 Golgi apparatus, 46 lysosome, 228 mitochondrion, 1169 nucleus, 64 peroxisome, 413 plasma membrane, 38 plastid, and 44 vacuole. Then, this dataset was further divided into training dataset (2423 sequences) and testing dataset (1727 sequences). And the specific numbers of proteins within each compartment of the training and testing datasets are shown in Table 8.1. As can be seen, both the training and testing datasets are quite imbalanced. The number of proteins in different subcellular locations vary significantly (from 4 to 695). Further, the datasets are both in low sequence similarity and in 16 subcellular locations. Thus, the properties of the training and testing dataset are imbalanced, multi-class distributed and in low sequence similarity, which make conventional methods difficult to classify.

The proteins in HUM12 were screened according to the same criteria mentioned above except that instead of sequences annotated with 'eukaryotic', sequences annotated with 'human' in the ID (identification) field were collected. The specific breakdown of the HUM12 dataset of both training and testing are shown in Table 8.2.

These two datasets are good benchmarks for performance comparison, because none of the proteins in either dataset has more than 25% sequence identity to any other proteins in the same subcellular location. However, the training and testing sets of these two datasets were constructed at the same period of time. Therefore, the training and testing sets are likely to share similar GO information, causing over-estimation in the prediction accuracy.

To avoid over-estimating the prediction performance and to demonstrate the effectiveness of predictors, a eukaryotic dataset containing novel proteins was constructed by using the criteria specified above. To ensure that the proteins are really novel to predictors, the creation dates of these proteins should be significantly later than the training proteins

Table 8.1: Breakdown of the benchmark dataset for single-label eukaryotic proteins (EU16). EU16 was extracted from Swiss-Prot 48.2. The sequence identity is below 25%.

Label	Subcellular Location	No. of sequences	
		Training	Testing
1	Cell Wall	20	5
2	Centriole	17	4
3	Chloroplast	207	51
4	Cyanelle	78	19
5	Cytoplasm	384	334
6	Cytoskeleton	20	5
7	Endoplasmic reticulum	91	22
8	Extracellular	402	404
9	Golgi apparatus	68	17
10	Lysosome	37	9
11	Mitochondrion	183	45
12	Nucleus	474	695
13	Peroxisome	52	12
14	Plasma membrane	323	90
15	Plastid	31	7
16	Vacuole	36	8
Total		2423	1727

(from EU16) and also later than the GOA database. Because EU16 was created in 2005 and the GOA database used was released on 08-Mar-2011, we selected the proteins that were added to Swiss-Prot between 08-Mar-2011 and 18-Apr-2012. Moreover, only proteins with a single subcellular location that falls within the 16 classes of the EU16 dataset were selected. After limiting the sequence similarity to 25%, 608 eukaryotic proteins distributed in 14 subcellular locations (see Table 8.3) were selected.

8.1.1.2 Datasets for FusionSVM

For the fusion of InterProGOSVM and PairProSVM, the performance was evaluated on Huang and Li's dataset [217], which was created by selecting all eukaryotic proteins with

Table 8.2: Breakdown of the benchmark dataset for single-label human proteins (HUM12). EU16 and HUM12 were extracted from Swiss-Prot 49.3. The sequence identity is below 25%.

Label	Subcellular Location	No. of sequences	
		Training	Testing
1	Centriole	20	5
2	Cytoplasm	155	222
3	Cytoskeleton	12	2
4	Endoplasmic reticulum	28	7
5	Extracellular	140	161
6	Golgi apparatus	33	9
7	Lysosome	32	8
8	Microsome	7	1
9	Mitochondrion	125	103
10	Nucleus	196	384
11	Peroxisome	18	5
12	Plasma membrane	153	215
Total		919	1122

annotated subcellular locations from Swiss-Prot 41.0. The dataset comprises 3572 proteins with 11 classes. The breakdown of the dataset is shown in Table 8.4. Specifically, there are 622 cytoplasm, 1188 nuclear, 424 mitochondria, 915 extracellular, 26 Golgi apparatus, 225 chloroplast, 45 endoplasmic reticulum, 7 cytoskeleton, 29 vacuole, 47 peroxisome, and 44 lysosome. The sequence similarity is cut off at 50%.

Among the 3572 protein sequences, only 3120 sequences have valid GO vectors by using InterProScan (with at least one non-zero element in the GO vectors). For the remaining 452 sequences, InterProScan cannot find any GO terms. Therefore, we only used sequences with valid GO vectors in our experiments and reduced the dataset size to 3120 protein sequences.

Table 8.3: Breakdown of the novel single-label eukaryotic-protein dataset for GOASVM. The dataset contains proteins that were added to Swiss-Prot created between 08-Mar-2011 and 18-Apr-2012. The sequence identity of the dataset is below 25%. *: no new proteins were found in the corresponding subcellular location.

Label	Subcellular Location	No. of sequences
1	Cell Wall	2
2	Centriole	0*
3	Chloroplast	51
4	Cyanelle	0*
5	Cytoplasm	77
6	Cytoskeleton	4
7	Endoplasmic reticulum	28
8	Extracellular	103
9	Golgi apparatus	14
10	Lysosome	1
11	Mitochondrion	73
12	Nucleus	57
13	Peroxisome	6
14	Plasma membrane	169
15	Plastid	5
16	Vacuole	18
Total		608

8.1.2 Performance Metrics

Several performance measures were used, including the overall accuracy (ACC), overall Mathew's correlation coefficient (OMCC) [218] and weighted average Mathew's correlation (WAMCC) [218]. The latter two measures are based on Mathew's correlation coefficient (MCC) [219]. Specifically, denote $M \in R^{C \times C}$ as the confusion matrix of the prediction results, where C is the number of subcellular locations. Then $M_{i,j} (1 \leq i, j \leq C)$ represents the number of proteins that actually belong to class i but are predicted as class j. Then, we further denote:

$$p_c = M_{c,c}, \tag{8.1}$$

Table 8.4: Breakdown of the dataset used for FusionSVM. This dataset is extracted from Swiss-Prot 41.0 and the sequence similarity is cut off to 50%.

Label	Subcellular Location	No. of Sequence
1	Cytoplasm	622
2	Nuclear	1188
3	Mitochondria	424
4	Extracellular	915
5	Golgi apparatus	24
6	Chloroplast	225
7	Endoplasmic reticulum	45
8	Cytoskeleton	7
9	Vacuole	29
10	Peroxisome	47
11	Lysosome	44
Total		3572

$$q_c = \sum_{i=1, i\neq c}^{C} \sum_{j=1, j\neq c}^{C} M_{i,j}, \tag{8.2}$$

$$r_c = \sum_{i=1, i\neq c}^{C} M_{i,c}, \tag{8.3}$$

$$s_c = \sum_{j=1, j\neq c}^{C} M_{c,j}, \tag{8.4}$$

where $c(1 \leq c \leq C)$ is the index of a particular subcellular location. For class c, p_c is the number of true positives, q_c is the number of true negatives, r_c is the number of false positives, and s_c is the number of false negatives. Based on these notations, the ACC, MCC$_c$ for class c, OMCC and WAMCC are defined respectively as:

$$\text{ACC} = \frac{\sum_{c=1}^{C} M_{c,c}}{\sum_{i=1}^{C}\sum_{j=1}^{C} M_{i,j}}, \tag{8.5}$$

$$\text{MCC}_c = \frac{p_c q_c - r_c s_c}{\sqrt{(p_c + s_c)(p_c + r_c)(q_c + s_c)(q_c + r_c)}}, \tag{8.6}$$

$$\text{OMCC} = \frac{\hat{p}\hat{q} - \hat{r}\hat{s}}{\sqrt{(\hat{p}+\hat{r})(\hat{p}+\hat{s})(\hat{q}+\hat{r})(\hat{q}+\hat{s})}}, \tag{8.7}$$

$$\text{WAMCC} = \sum_{c=1}^{C} \frac{p+s}{N} \text{MCC}_c, \tag{8.8}$$

where $N = \sum_{c=1}^{C} p_c + s_c, \hat{p} = \sum_{c=1}^{C} p_c, \hat{q} = \sum_{c=1}^{C} q_c, \hat{r} = \sum_{c=1}^{C} r_c, \hat{s} = \sum_{c=1}^{C} s_c$.

MCC can overcome the shortcoming of accuracy on imbalanced data and have the advantage of avoiding the performance to be dominated by the majority classes. For example, a classifier which predicts all samples as positive cannot be regarded as a good classifier unless it can also predict negative samples accurately. In this case, the accuracy and MCC of the positive class are 100% and 0%, respectively. Therefore, MCC is a better measure for imbalanced classification.

8.2 Prediction of Multi-Label Proteins

This section will focus on constructing datasets and introducing performance metrics for multi-location proteins, which are used in mGOASVM, AD-SVM, mPLR-Loc, SS-Loc, HybridGO-Loc, RP-SVM and R3P-Loc.

8.2.1 Datasets Construction

For multi-location protein subcellular localization, datasets from three species are constructed: virus, plant and eukaryote.

The datasets that we used for evaluating the proposed multi-label predictors have also been used by other multi-label predictors, including Virus-mPLoc [154], KNN-SVM [156], Plant-mPLoc [96], Euk-mPLoc 2.0 [158], iLoc-Virus [93], iLoc-Plant [87] and iLoc-Euk [90].

The virus dataset was created from Swiss-Prot 57.9. It contains 207 viral proteins distributed in 6 locations (see Table 8.5). Of the 207 viral proteins, 165 belong to one

Table 8.5: Breakdown of the multi-label virus protein dataset. The sequence identity is cut off at 25%. The superscripts v stand for the virus dataset.

Label	Subcellular Location	No. of Locative Proteins
1	Viral capsid	8
2	Host cell membrane	33
3	Host endoplasmic reticulum	20
4	Host cytoplasm	87
5	Host nucleus	84
6	Secreted	20
Total number of locative proteins (N^v_{loc})		252
Total number of actual proteins (N^v_{act})		207

subcellular locations, 39 to two locations, 3 to three locations and none to four or more locations. This means that about 20% of proteins are located in more than one subcellular location. The sequence identity of this dataset was cut off at 25%.

The plant dataset was created from Swiss-Prot 55.3. It contains 978 plant proteins distributed in 12 locations (see Table 8.6). Of the 978 plant proteins, 904 belong to one subcellular locations, 71 to two locations, 3 to three locations and none to four or more locations. In other words, 8% of the plant proteins in this dataset are located in multiple locations. The sequence identity of this dataset was cut off at 25%.

The eukaryotic dataset was created from Swiss-Prot 55.3. It contains 7766 eukaryotic proteins distributed in 22 locations (see Table 8.7). Of the 7766 eukaryotic proteins, 6687 belong to one subcellular location, 1029 to two locations, 48 to three locations, 2 to four locations and none to five or more locations. In other words, about 14% of the eukaryotic proteins in this dataset are located in multiple locations. Similarly, the sequence identity of this dataset was cut off at 25%.

To further demonstrate the effectiveness of the proposed predictors, a plant dataset containing novel proteins was constructed by using the criteria specified in [96, 87]. Specif-

Table 8.6: Breakdown of the multi-label plant protein dataset. The sequence identity is cut off at 25%. The superscripts p stand for the plant dataset.

Label	Subcellular Location	No. of Locative Proteins
1	Cell membrane	56
2	Cell wall	32
3	Chloroplast	286
4	Cytoplasm	182
5	Endoplasmic reticulum	42
6	Extracellular	22
7	Golgi apparatus	21
8	Mitochondrion	150
9	Nucleus	152
10	Peroxisome	21
11	Plastid	39
12	Vacuole	52
Total number of locative proteins (N^p_{loc})		1055
Total number of actual proteins (N^p_{act})		978

ically, to ensure that the proteins are really novel to predictors, the creation dates of these proteins should be significantly later than the training proteins (from the plant dataset) and also later than the GOA database. Because the plant dataset was created in 2008 and the GOA database used was released on 08-Mar-2011, we selected the proteins that were added to Swiss-Prot between 08-Mar-2011 and 18-Apr-2012. Moreover, proteins with multiple subcellular locations that falls within the 12 classes specified in Table 8.6 were included. After limiting the sequence similarity to 25%, 175 plant proteins distributed in 12 subcellular locations (see Table 8.8) were selected. Of the 175 plant proteins, 147 belong to one subcellular location, 27 belong to two locations, 1 belong to three locations and none to four or more locations. In other words, 16% of the plant proteins in this novel dataset are located in multiple locations.

Here, we take the new plant dataset as an example to illustrate the details of the

Table 8.7: Breakdown of the multi-label eukaryotic protein dataset. The sequence identity is cut off at 25%. The superscripts e stand for the eukaryotic dataset.

Label	Subcellular Location	No. of Locative Proteins
1	Acrosome	14
2	Cell membrane	697
3	Cell wall	49
4	Centrosome	96
5	Chloroplast	385
6	Cyanelle	79
7	Cytoplasm	2186
8	Cytoskeleton	139
9	ER	457
10	Endosome	41
11	Extracellular	1048
12	Golgi apparatus	254
13	Hydrogenosome	10
14	Lysosome	57
15	Melanosome	47
16	Microsome	13
17	Mitochondrion	610
18	Nucleus	2320
19	Peroxisome	110
20	SPI	68
21	Synapse	47
22	Vacuole	170
Total number of locative proteins (N^e_{loc})		8897
Total number of actual proteins (N^e_{act})		7766

procedures, which are specified as follows:

1. Go to the UniProt/SwissProt official webpage (http://www.uniprot.org/);

2. Go to the 'Search' section and select 'Protein Knowledgebase (UniProtKB)' (default) in the 'Search in' option;

3. In the 'Query' option, select or type 'reviewered: yes';

4. Select 'AND' in the 'Advanced Search' option, and then select 'Taxonomy [OC]' and type in 'Viridiplantae';
5. Select 'AND' in the 'Advanced Search' option, and then select 'Fragment: no';
6. Select 'AND' in the 'Advanced Search' option, and then select 'Sequence length' and type in '50 - ' (no less than 50);
7. Select 'AND' in the 'Advanced Search' option, and then select 'Date entry integrated' and type in '20110308-20120418';
8. Select 'AND' in the 'Advanced Search' option, and then select "Subcellular location: XXX Confidence: Experimental"; (XXX means the specific subcellular locations. Here it includes 12 different locations: cell membrane; cell wall; chloroplast; endoplasmic reticulum; extracellular; golgi apparatus; mitochondrion; nucleus; peroxisome; plastid; vacuole.)
9. Further exclude those proteins which are not experimentally annotated (This is to recheck the proteins to guarantee they are all experimentally annotated).

After selecting the proteins, Blastclust[1] was applied to reduce the redundancy in the dataset so that none of the sequence pairs has sequence identity higher than 25%.

8.2.2 Datasets Analysis

Since the multi-label datasets are more complicated than single-label datasets, some analysis should be necessarily carried out. To better visualize the distributions of proteins in each subcellular locations in these three datasets, we have listed the breakdown of these three datasets in Figs. 8.1, 8.2 and 8.3. Fig. 8.1 shows that the majority (68%) of

[1]http://www.ncbi.nlm.nih.gov/Web/Newsltr/Spring04/blastlab.html

Table 8.8: Breakdown of the new multi-label plant dataset. The dataset was constructed from Swiss-Prot created between 08-Mar-2011 and 18-Apr-2012. The sequence identity of the dataset is below 25%.

Label	Subcellular Location	No. of Locative Proteins
1	Cell membrane	16
2	Cell wall	1
3	Chloroplast	54
4	Cytoplasm	38
5	Endoplasmic reticulum	9
6	Extracellular	3
7	Golgi apparatus	7
8	Mitochondrion	16
9	Nucleus	46
10	Peroxisome	6
11	Plastid	1
12	Vacuole	7
Total number of locative proteins		204
Total number of actual proteins		175

viral proteins in the virus dataset are located in host cytoplasm and host nucleus while proteins located in the rest of the subcellular locations totally account only around one third. This means that this multi-label dataset is imbalanced across the six subcellular locations. Similar conclusions can be drawn from Fig. 8.2, where most of the plant proteins exist in chloroplast, cytoplasm, nucleus and mitochondrion while proteins in other 8 subcellular locations totally account for less than 30%. This imbalanced property makes the prediction of these two multi-label datasets difficult. Fig. 8.3 also shows the imbalanced property of the multi-label dataset, where the majority (78%) of eukaryotic proteins are located in nucleus, cytoplasm, extracellular, cell membrane and mitochondrion while proteins in other 17 subcellular locations totally account for less than 22%.

More detailed statistical properties of these three datasets are listed in Table 8.9.

In Table 8.9, M and N denote the number of actual (or distinct) subcellular locations

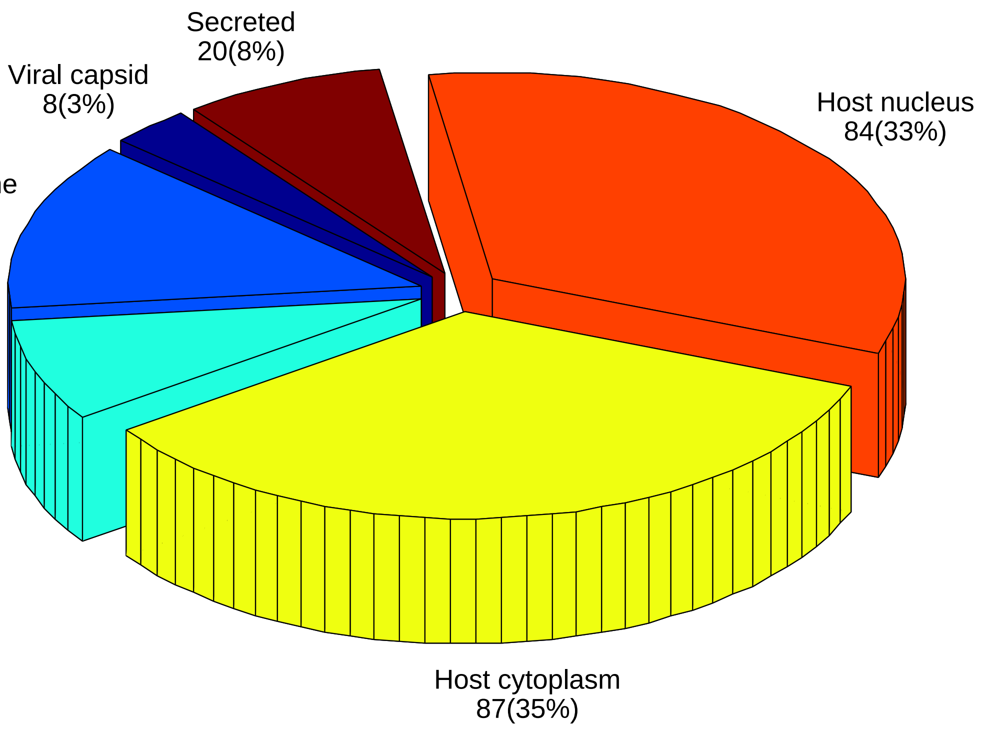


Figure 8.1: **Breakdown of the multi-label virus dataset**. The number of proteins shown in each subcellular location represents the number of 'locative proteins' [93, 108]. Here, 207 actual proteins have 252 locative proteins. The viral proteins are distributed in 6 subcellular locations, including viral capsid, host cell membrane, host endoplasmic reticulum, host cytoplasm, host nucleus and secreted.

and the number of actual (or distinct) proteins. Besides the commonly used properties for single-label classification, the following measurements [144] are used as well to explicitly quantify the multi-label properties of the datasets:

1. *Label Cardinality (LC)*. LC is the average number of labels per data instance, which is defined as: $LC = \frac{1}{N}\sum_{i=1}^{N}|\mathcal{L}(\mathbb{Q}_i)|$, where $\mathcal{L}(\mathbb{Q}_i)$ is the label set of the protein $\mathbb{Q}_i$ and $|\cdot|$ denotes the cardinality of a set;

2. *Label Density (LD)*. LD is LC normalized by the number of classes, which is defined as: $LD = \frac{LC}{M}$;

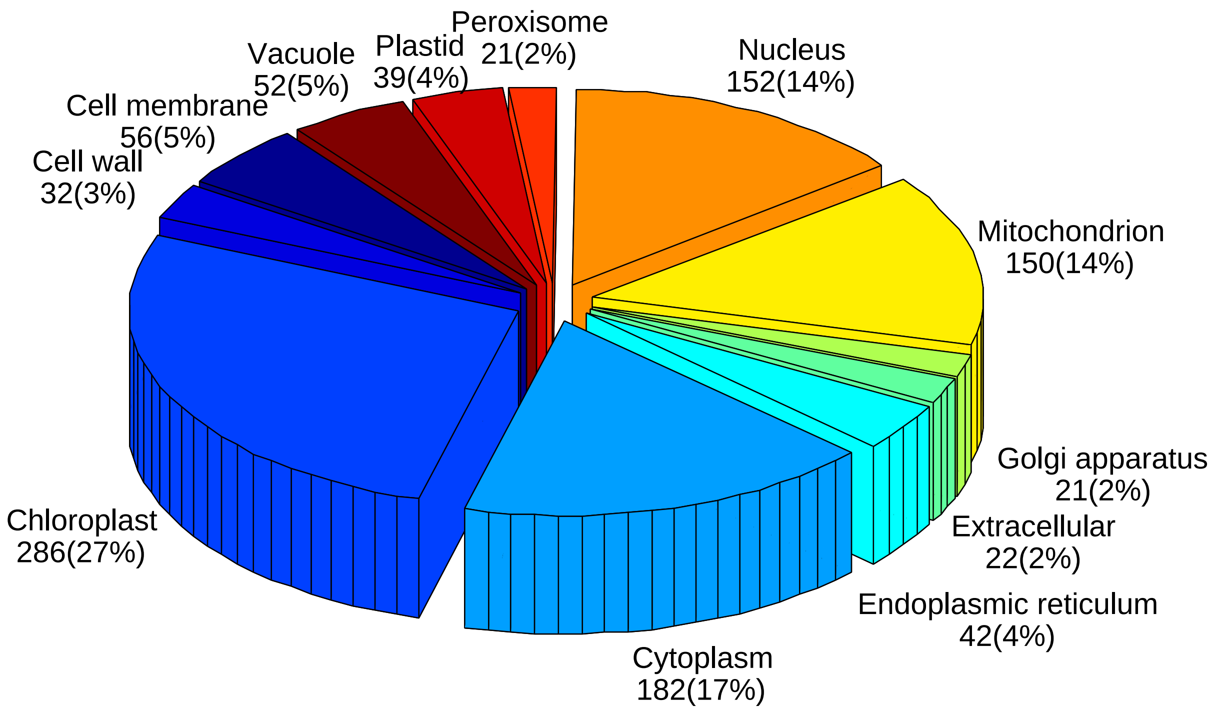


Figure 8.2: Breakdown of the multi-label plant dataset. The number of proteins shown in each subcellular location represents the number of 'locative proteins' [93, 108]. Here, 978 actual proteins have 1055 locative proteins. The plant proteins are distributed in 12 subcellular locations, including cell membrane, cell wall, chloroplast, cytoplasm, endoplasmic reticulum, extracellular, Golgi apparatus, mitochondrion, nucleus, peroxisome, plastid and vacuole.

3. *Distinct Label Set (DLS)*. DLS is the number of label combinations in the dataset;

4. *Proportion of Distinct Label Set (PDLS)*. $PDLS$ is DLS normalized by the number of actual data instances, which is defined as: $PDLS = \frac{DLS}{N}$;

5. *Total Locative Number (TLN)*. TLN is the total number of locative proteins. This concept is derived from locative proteins in [93], which will be further elaborated in Section 8.2.3.

Among these measurements, LC is used to measure the degree of multi-labels in a dataset. For a single-label dataset, $LC = 1$; for a multi-label dataset, $LC > 1$. And

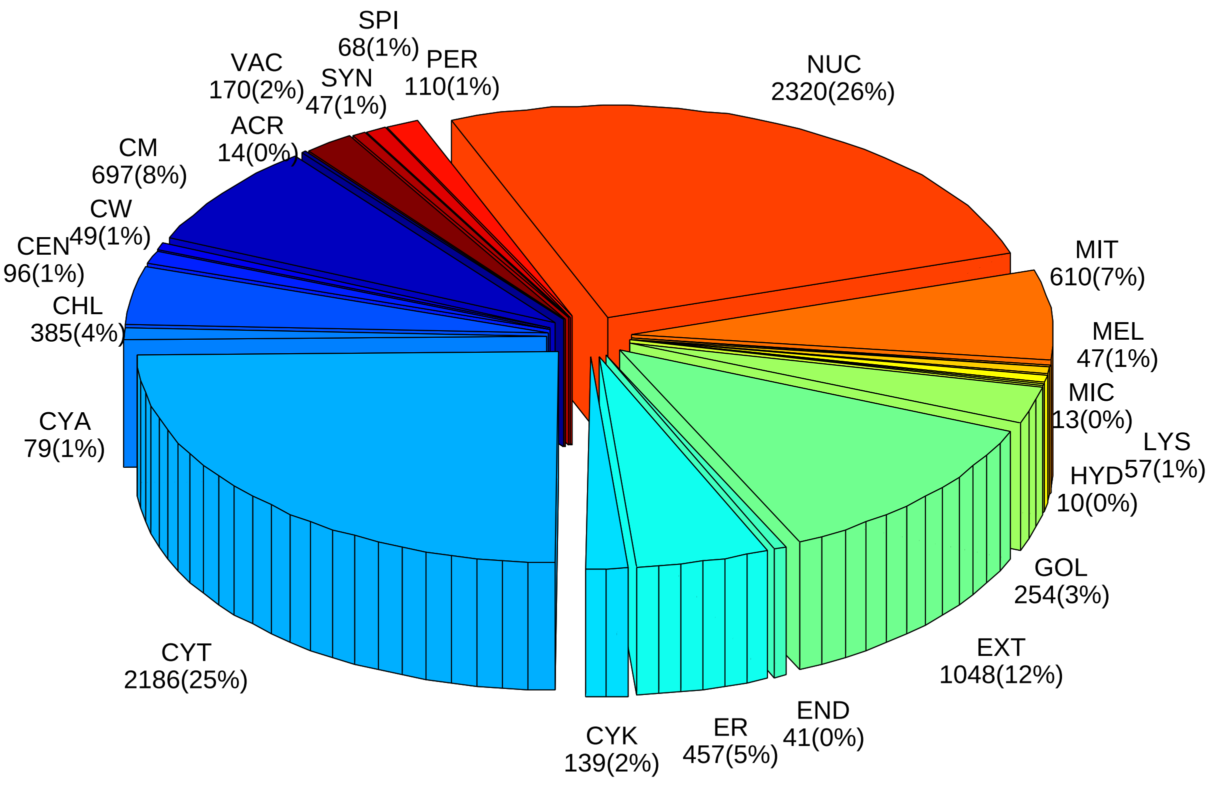


Figure 8.3: Breakdown of the multi-label eukaryotic dataset. The number of proteins shown in each subcellular location represents the number of 'locative proteins' [93, 108]. Here, 7766 actual proteins have 8897 locative proteins. The eukaryotic proteins are distributed in 22 subcellular locations, including acrosome (ACR), cell membrane (CM), cell wall (CW), centrosome (CEN), chloroplast (CHL), cyanelle (CYA), cytoplasm (CYT), cytoskeleton (CYK), endoplasmic reticulum (ER), endosome (END), extracellular (EXT), Golgi apparatus (GOL), hydrogenosome (HYD), lysosome (LYS), melanosome (MEL), microsome (MIC), mitochondrion (MIT), nucleus (NUC), peroxisome (PER), spindle pole body (SPI), synapse (SYN) and vacuole (VAC).

the larger the LC, the higher the degree of multi-labels. LD takes into consideration the number of classes in the classification problem. For two datasets with the same LC, the lower the LD, the more difficult the classification. DLS represents the number of possible label combinations in the dataset. The higher the DLS, the more complicated the composition. $PDLS$ represents the degree of distinct labels in a dataset. The larger the $PDLS$, the more probable the individual label-sets are different from each other.

Table 8.9: Statistical properties of the three multi-label benchmark datasets used in our experiments.

Dataset	M	N	TLN	LC	LD	DLS	$PDLS$
Virus	6	207	252	1.2174	0.2029	17	0.0821
Plant	12	978	1055	1.0787	0.0899	32	0.0327
Eukaryote	22	7766	8897	1.1456	0.0521	112	0.0144

M: number of subcellular locations.
N: number of actual proteins.
LC: label cardinality.
LD: label density.
DLS: distinct label set.
$PDLS$: proportion of distinct label set.
TLN: total locative number.

From Table 8.9, we notice that although the number of proteins in the virus dataset ($N = 207, TLN = 252$) is smaller than that of the plant dataset ($N = 978, TLN = 1055$) and eukaryotic dataset ($N = 7766, TLN = 8897$), the former ($LC = 1.2174, LD = 0.2029$) is a denser multi-label dataset than the latter two ($LC = 1.0787, LD = 0.0899$ and $LC = 1.1456, LD = 0.0521$).

8.2.3 Performance Metrics

Compared to traditional single-label classification, multi-label classification requires more complicated performance metrics to better reflect the multi-label capabilities of classifiers. Conventional single-label measures need to be modified to adapt to multi-label classification. These measures include *Accuracy, Precision, Recall, F1-score (F1)* and *Hamming Loss (HL)* [220, 221]. Specifically, denote $\mathcal{L}(\mathbb{Q}_i)$ and $\mathcal{M}(\mathbb{Q}_i)$ as the true label set and the predicted label set for the i-th protein $\mathbb{Q}_i$ ($i = 1, \ldots, N$), respectively.[2] Then the five

[2]Here, $N = 207$ for the virus dataset and $N = 978$ for the plant dataset.

measurements are defined as follows:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|}{|\mathcal{M}(\mathbb{Q}_i)\cup\mathcal{L}(\mathbb{Q}_i)|}\right) \tag{8.9}$$

$$Precision = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|}{|\mathcal{M}(\mathbb{Q}_i)|}\right) \tag{8.10}$$

$$Recall = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|}{|\mathcal{L}(\mathbb{Q}_i)|}\right) \tag{8.11}$$

$$F1 = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{2|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|}{|\mathcal{M}(\mathbb{Q}_i)|+|\mathcal{L}(\mathbb{Q}_i)|}\right) \tag{8.12}$$

$$HL = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\mathcal{M}(\mathbb{Q}_i)\cup\mathcal{L}(\mathbb{Q}_i)|-|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|}{M}\right) \tag{8.13}$$

where $|\cdot|$ means counting the number of elements in the set therein and $\cap$ represents the intersection of sets.

Accuracy, Precision, Recall and *F1* indicate the classification performance. The higher the measures, the better the prediction performance. Among them, *Accuracy* is the most commonly used criteria. *F1-score* is the harmonic mean of *Precision* and *Recall*, which allows us to compare the performance of classification systems by taking the trade-off between *Precision* and *Recall* into account. The *Hamming Loss (HL)* [220, 221] is different from other metrics. As can be seen from Eq. 8.13, when all of the proteins are correctly predicted, i.e., $|\mathcal{M}(\mathbb{Q}_i)\cup\mathcal{L}(\mathbb{Q}_i)| = |\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)|$ ($i = 1, \ldots, N$), then $HL = 0$; whereas, other metrics will be equal to 1. On the other hand, when the predictions of all proteins are completely wrong, i.e., $|\mathcal{M}(\mathbb{Q}_i)\cup\mathcal{L}(\mathbb{Q}_i)| = M$ and $|\mathcal{M}(\mathbb{Q}_i)\cap\mathcal{L}(\mathbb{Q}_i)| = 0$, then $HL = 1$; whereas, other metrics will be equal to 0. Therefore, the lower the *HL*, the better the prediction performance.

Two additional measurements [93, 108] are often used in multi-label subcellular localization prediction. They are overall locative accuracy (OLA) and overall actual accuracy (OAA). The former is given by:

$$OLA = \frac{1}{\sum_{i=1}^{N} |\mathcal{L}(\mathbb{Q}_i)|} \sum_{i=1}^{N} |\mathcal{M}(\mathbb{Q}_i) \cap \mathcal{L}(\mathbb{Q}_i)|, \tag{8.14}$$

and the overall actual accuracy (OLA) is:

$$OAA = \frac{1}{N} \sum_{i=1}^{N} \Delta[\mathcal{M}(\mathbb{Q}_i), \mathcal{L}(\mathbb{Q}_i)] \tag{8.15}$$

where

$$\Delta[\mathcal{M}(\mathbb{Q}_i), \mathcal{L}(\mathbb{Q}_i)] = \begin{cases} 1 & \text{, if } \mathcal{M}(\mathbb{Q}_i) = \mathcal{L}(\mathbb{Q}_i) \\ 0 & \text{, otherwise.} \end{cases} \tag{8.16}$$

According to Eq. 8.14, a locative protein is considered to be correctly predicted if any of the predicted labels matches any labels in the true label set. On the other hand, Eq. 8.15 suggests that an actual protein is considered to be correctly predicted only if *all* of the predicted labels match those in the true label set exactly. For example, for a protein coexist in, say three subcellular locations, if only two of the three are correctly predicted, or the predicted result contains a location not belonging to the three, the prediction is considered to be incorrect. In other words, when and only when all of the subcellular locations of a query protein are exactly predicted without any overprediction or underprediction, can the prediction be considered as correct. Therefore, OAA is a more stringent measure as compared to OLA. OAA is also more objective than OLA. This is because locative accuracy is liable to give biased performance measures when the predictor tends to over-predict, i.e., giving large $|\mathcal{M}(\mathbb{Q}_i)|$ for many $\mathbb{Q}_i$. In the extreme case, if every protein is predicted to have all of the M subcellular locations, according to Eq. 8.14, the OLA is 100%. But obviously, the predictions are wrong and meaningless. On the contrary, OAA is 0% in this extreme case, which definitely reflects the real performance.

Among all the metrics mentioned above, OAA is the most stringent and objective. This is because if only some (but not all) of the subcellular locations of a query protein are correctly predict, the numerators of the other 4 measures (Eqs. 8.9 to 8.14) are non-zero, whereas the numerator of OAA in Eq. 8.15 is 0 (thus contribute nothing to the frequency count).

8.3 Statistical Evaluation Methods

In statistical prediction, there are three methods that are often used for testing the generalization capabilities of predictors: independent tests, subsampling tests (or K-fold cross-validation) and leave-one-out cross validation (LOOCV).

In independent tests, the training set and the testing set were fixed, thus enabling us to obtain a fixed accuracy for the predictors. This kind of methods can directly demonstrate the capability of predictors. However, the selection of independent dataset often bears some sort of arbitrariness [222], which inevitably leads to non-bias-free accuracy for the predictors.

In subsampling tests, here we use five-fold cross validation as an example. The whole dataset was randomly divided into 5 disjoint parts with equal size [83]. The last part may have 1-4 more examples than the former 4 parts in order for each example to be evaluated on the model. Then one part of the dataset was used as the test set and the remained parts are jointly used as the training set. This procedure is repeated five times, and each time a different part was chosen as the test set. The number of the selections in dividing the benchmark dataset is obviously an astronomical figure even for a small-size dataset. This means that different selections lead to different results even for the same benchmark dataset, thus still being liable to statistical arbitrariness. Subsampling tests with a smaller K work definitely faster than that with a larger K. Thus, subsampling tests are faster than LOOCV, which can be regarded as N-fold cross-validation, where N is the number of samples in the dataset, and $N > K$. At the same time, it is also statistically acceptable and usually regarded as less biased than the independent tests.

In LOOCV, every protein in the benchmark dataset will be singled out one-by-one and is tested by the classifier trained by the remaining proteins. In each fold of LOOCV, a protein of the training dataset (suppose there are N proteins) was singled out as the test protein and the remaining $(N-1)$ proteins were used as the training data. This procedure was repeated N times, and in each fold a different protein was selected as the test protein. This ensures that every sequence in the dataset will be tested. In this case, the arbitrariness can be avoided because LOOCV will yield a unique outcome for the predictors. Therefore, LOOCV is considered to be the most rigorous and bias-free method [223]. Note that the jackknife cross validation in iLoc-Plant and its variants is the same as LOOCV, as mentioned in [87, 222]. Because the term jackknife also refers to the methods that estimate the bias and variance of an estimator [224], to avoid confusion, we only use the term LOOCV in the whole thesis.

For both single-label and multi-label datasets, LOOCV was used for benchmark datasets and independent tests was implemented for the novel datasets, with the benchmark datasets of the corresponding species as the training part.

8.4 Summary

This chapter mainly introduces experimental setups for both single-label protein subcellular localization and multi-label protein subcellular localization. datasets construction and performance metrics are presented for single-label and multi-label cases, respectively. For both cases, three benchmark datasets and one novel dataset were introduced, respectively. Different performance metrics were presented for prediction of single-label and multi-label proteins. Generally speaking, datasets and performance metrics for the multi-label case are much more sophisticated than the single-label one. Hence, a section was particularly specified for analysis of multi-label datasets. Finally, some statistical methods for evaluation were elaborated.

Chapter 9: Result and Analysis

The results chapter of a thesis is often simply a presentation of results, including tables, diagrams and a description of the findings. It is often done without any interpretation or discussion of the results, which often comes in a separate chapter. This chapter includes a discussion of both the results and the methodology chosen.

This chapter reports on the experimental results and analyses them. The chapter is organised in the following way and is very effective partly because the writer includes the following:

Structure

Introduction	Not included
Performance of each model described per section with a short analysis of the results.	Section 9.1-9.9.
Comparison of performances	Section 9.10
Summary	Section 9.11

Content

- Gives a short introductory paragraph outlining the content of the chapter at the start of the chapter (e.g. Chapter 9, paragraph 1)
- Compares results with other methodologies (e.g. Section 9.1.1, paragraph 3)
- Discusses limits of the method (e.g. Section 9.1.2, paragraph 1, sentence 4-6)
- Highlights implication of findings (e.g. Section 9.1.3, paragraph 1, final 2 sentences)
- Develops paragraphs clearly, e.g. Section 9.4, paragraph 1:

  Sentence 1 Overview of figure

  Sentence 2 Main findings

  Sentence 3-4 Interpretation of findings

  Sentence 5 Compares findings
- Refer the reader to discussions later in the chapter, e.g. *found in Section 9.10* (e.g. Section, 9.5.1, paragraph 2)
- Describes how to read charts (e.g. Section 9.9.2, paragraph 2, sentence 2)
- Provides a short summary section (e.g. Section 9.11)

Language

- Uses past tense and passive voice to describe the methodology, e.g. *was set* (Section 9.1.1, paragraph 1, sentence 1)
- Highlights main finding from the chart as a topic sentence (e.g. Section 9.1.1, paragraph 2, sentence 1)
- Interprets chart using uncertain language, e.g. *also suggests*, *this may be due to* sentence (Section 9.1.1, paragraph 2, sentence 2-3)
- Uses adverbs to highlight importance, e.g. *significantly* (e.g. Section 9.2.3, paragraph 2, sentence 3)
- Uses summary sentences at the end of paragraphs (e.g. Section 9.3.4, paragraph 1, final sentence)
- Uses a range of comparative grammar, e.g. *slightly smaller, higher than, better than, the better, also comparable, similar conclusions, compared to* (e.g. Section 9.4.2, section 1, paragraph 1-2)

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

Highlight unexpected results.

Provide a short introduction to each section. 9.1 does not include any information and 9.1.1 follows.

Use a subject with the verb note, e.g. *It is noted* (e.g. Section 9.1.2, paragraph 3, sentence 3).

Avoid confusing *compared with* and *comparing* e.g. Compared with modern understanding of genes, the 1950s had very limited understanding (e.g. Section 9.1.3).

Avoid using spoken language e.g. *quite a bit* and *a lot of* (e.g. Section 9.2.1 paragraph 1, sentence 4). It is better to use more formal terms e.g. *a considerable amount*.

Avoid overusing reporting verbs, e.g. *suggest* and *suggestion* are used throughout the chapter (e.g. Section 9.1.5 sentence 4 and 5). Researchers also use a variety of reporting verbs with similar meanings e.g. *indicate. implies, indicates.*

Avoid overly complex sentences. If the sentence has too many parts it becomes difficult to read (e.g. Section 9.3.1 paragraph 3, final sentence).

Include much of the analysis of findings in the discussion chapter.

Chapter 9

Results and Analysis

This chapter will elaborate the experimental results and related analysis for all the predictors introduced in previous chapters, including GOASVM and FusionSVM for single-location protein subcellular localization, and mGOASVM, AD-SVM, mPLR-Loc, SS-Loc, HybridGO-Loc, RP-SVM and REP-Loc for multi-location protein subcellular localization.

9.1 Performance of GOASVM

9.1.1 Comparing GO Vector Construction Methods

Table 9.1 shows the performance of different GO-vector construction methods on the EU16, HUM12 and the novel eukaryote (NE16) datasets, which are detailed in Tables 8.1, 8.2 and 8.3, respectively. Linear SVMs were used for all cases, and the penalty factor was set to 0.1. For the EU16 and HUM12 datasets, leave-one-out cross-validation (LOOCV) was used to evaluate the performance of GOASVM; for the NE16 dataset, the EU16 training dataset was used for training the classifier, which was subsequently used to classify proteins in the NE16 dataset. Four different GO-vector construction methods were tried, including 1-0 value, term-frequency (TF), inverse sequence-frequency (ISF) and term-frequency inverse sequence-frequency (TF-ISF).

Table 9.1: Performance of different GO-vector construction methods for GOASVM on the EU16, HUM12 and the novel eukaryotic datasets, respectively. *NE16*: the novel eukaryotic dataset whose proteins are distributed in 16 subcellular locations (See Table 8.3 in Chapter 8); *TF*: term-frequency; *ISF*: inverse sequence-frequency; *TF-ISF*: term-frequency inverse sequence frequency. *OMCC*: Overall MCC; *WAMCC*: Weighted average MCC; *ACC*: Overall accuracy. Refer to Eqs. 8.5, Eqs. 8.7 and Eqs. 8.8 for the definition of ACC, OMCC and WAMCC. The higher these three evaluation measures, the better the performance.

Dataset	GO Vector Construction Method	OMCC	WAMCC	ACC
EU16	1-0 value	0.9252	0.9189	92.98%
	TF	**0.9419**	**0.9379**	**94.55%**
	ISF	0.9243	0.9191	92.90%
	TF-ISF	0.9384	0.9339	94.22%
HUM12	1-0 value	0.8896	0.8817	89.88%
	TF	**0.9074**	**0.9021**	**91.51%**
	ISF	0.8659	0.8583	87.70%
	TF-ISF	0.8991	0.8935	90.75%
NE16	1-0 value	0.6877	0.6791	70.72%
	TF	**0.7035**	**0.6926**	**72.20%**
	ISF	0.6386	0.6256	66.12%
	TF-ISF	0.6772	0.6626	69.74%

Evidently, for all the three datasets, term-frequency (TF) performs the best among these four methods, which demonstrates that the frequencies of occurrences of GO terms provide additional information for subcellular localization. The results also suggest that inverse sequence-frequency (ISF) is detrimental to classification performance, despite its proven effectiveness in document retrieval. This may be due to the differences between the frequency of occurrences of common GO terms in our datasets and the frequency of occurrences of common words in document retrieval. In document retrieval, almost all documents contain the common words; as a result, the inverse document frequency is effective in suppressing the influence of these words in the retrieval. However, the common GO terms do not appear in all of the proteins in our datasets. In fact, for example, even the most commonly occurred GO term appears only in one-third of the proteins in EU16. We conjecture that this low-frequency of occurrences of common GO terms makes ISF not effective for subcellular localization.

Many existing GO-based methods use the 1-0 value approach to constructing GO vectors, including ProLoc-GO [72], Euk-OET-PLoc [64], and Hum-PLoc [85]. Table 9.1 shows that term-frequency (TF) performs almost 2% better than 1-0 value (72.20% vs 70.72%). Similar conclusions can be also drawn from the performance of GOASVM based on leave-one-out cross validation on the EU16 training set and the HUM12 training set. The results are biologically relevant because proteins of the same subcellular localization are expected to have a similar number of occurrences of the same GO term. In this regard, the 1-0 value approach is inferior because it quantizes the number of occurrences of a GO term to 0 or 1. Recently, we found that an approach similar to the TF approach had also been used in iLoc-Euk [90], iLoc-Hum [88], iLoc-Plant [87], iLoc-Gpos [89], iLoc-Gneg [91], and iLoc-Virus [93].

9.1.2 Performance of Successive-Search Strategy

Because the novel proteins were recently added to Swiss-Prot, many of them have not been annotated in the GOA database. As a results, if we used the accession numbers of these proteins to search against the GOA database, the corresponding GO vectors will contain all zeros. This suggests that we should use the ACs of their homologs as the searching keys, i.e., the procedure shown in Fig. 4.3 should be adopted. However, we observed that for some novel proteins, even the top homologs do not have any GO terms annotated to them. In particular, in the new dataset, there are 169 protein sequences whose top homologs do not have any GO terms (2-nd row of Table 9.2), causing GOASVM unable to make any predictions. As can be seen from Table 9.2, by using only the first

Table 9.2: Performance of GOASVM using successive-search strategy on the novel eukaryotic (NE16) dataset denoted in Table 8.3. The 2nd column represents the upper bound of k in $\mathbf{q}_{i,k}$ shown in Fig 4.1 of Section 4.1.2 in Chapter 4. For example, when $k_{\max} = 2$, only the AC of the 1st- or 2nd homolog will be used for retrieving the GO terms. *No. of sequences without GO terms* means the number of protein sequences for which no GO terms can be retrieved. *OMCC*: Overall MCC; *WAMCC*: Weighted average MCC; *ACC*: Overall accuracy. See Supplementary Materials for the definition of these performance measures. Note for fair comparison, the *Baseline* shown here is the performance of Euk-OET-PLoc, which we implemented and also adopts the same procedure as GOASVM to obtain GO terms from homologs. *: Since the web-server of Euk-OET-PLoc is not available now, we implemented it according to [64].

Method	$k_{\max}$	No. without GO terms	OMCC	WAMCC	ACC
GOASVM	1	169	0.5421	0.5642	57.07%
	2	112	0.5947	0.6006	62.01%
	3	12	0.6930	0.6834	71.22%
	4	7	0.6980	0.6881	71.71%
	5	3	0.7018	0.6911	72.04%
	6	3	0.7018	0.6911	72.04%
	7	0	**0.7035**	**0.6926**	**72.20%**
Baseline*	7	0	0.5246	0.5330	55.43%

homolog, the overall prediction accuracy of GOASVM is only 57.07% (347/608). To overcome this limitation, the following strategy was adopted. For the 169 proteins (2-nd row of Table 9.2) whose top homologs do not have any GO terms in the GOA database, we used the second-top homolog to find the GO terms; similarly, for the 112 proteins (3-rd row of Table 9.2) whose top and 2-nd homologs do not have any GO terms, the third-top homolog was used; and so on until all the query proteins can correspond to at least one GO term. In the case where BLAST fails to find any homologs (although this case rarely happens) the default E-value threshold (the -e option) can be relaxed. Detailed descriptions of this strategy can be found in Section 4.1.2 in Chapter 4.

Table 9.2 shows the prediction performance of GOASVM on the NE16 dataset (608 novel proteins). As explained earlier, to ensure that these proteins are novel to GOASVM, 2423 proteins extracted from the training set of EU16 were used for training the classifier. For fair comparison, Euk-OET-PLoc [64] also uses the same version of the GOA database (08-Mar-2011) to retrieve GO terms and adopts the same procedure as GOASVM to obtain GO terms from homologs. In such case, for Euk-OET-PLoc, it is unnecessary to use the PseAA[29] as a backup method because a valid GO vector can be found for every protein in this novel dataset. Also, according to Euk-OET-PLoc [64], several parameters are optimized and only the best performance is shown here (See the last row of Table 9.2). As can be seen, GOASVM performs significantly better than Euk-OET-PLoc (72.20% vs 55.43%), demonstrating that GOASVM is more capable of predicting novel proteins than Euk-OET-PLoc. Moreover, results clearly suggest that when more distant homologs are allowed to be used for searching GO terms in the GOA database, we have a higher chance of finding at least one GO terms for each of these novel proteins, thus improving the overall performance. In particular, when the most distant homolog has a rank of 7 ($k_{\text{max}} = 7$), GOASVM is able to find GO terms for all of the novel proteins and the accuracy is also the highest, which is almost 15% (absolute) higher than that using only the top homolog. Given the novelty of these proteins and the low sequence similarity (below 25%), an accuracy of 72.2% is fairly high, suggesting that the homologs of novel proteins can provide useful GO information for protein subcellular localization.

Note that the gene association file that we downloaded from the GOA database does not provide any subcellular localization labels. This file only allows us to create a hash table storing the association between the accession numbers and their corresponding GO terms. This hash table covers all of the accession numbers in the GOA database released on 08-Mar-2011, meaning that it will cover the EU16 (dated in 2005) but not the accession numbers in the novel eukaryotic dataset. It is important to emphasize that given a query protein, having a match in this hash table does not mean that a subcellular-localization assignment can be obtained. In fact, having a match only means that a non-null GO vector can be obtained. After that, the SVMs play an important role in classifying the non-null GO vector.

9.1.3 Comparing with Methods Based on Other Features

Table 9.3: Performance of different features and different SVM classifiers on the EU16 training dataset. Features include amino acid composition (AA) [20], amino-acid pair composition (PairAA) [20], AA composition with gap (length = 48) (GapAA) [21], pseudo AA composition (PseAA) [29], and profile alignment scores [218].

Classifier	Feature	Post-Processing	OMCC	WAMCC	ACC
RBF SVM	AA	Vector Norm	0.3846	0.3124	42.30%
RBF SVM	AA+PairAA	Vector Norm	0.4119	0.3342	44.86%
Linear SVM	AA+PairAA+GapAA(48)	Vector Norm	0.4524	0.3797	48.66%
RBF SVM	PseAA	Vector Norm	0.4185	0.3467	45.48%
Linear SVM	Profile vectors	Geometric Mean	0.5149	0.4656	54.52%
Linear SVM	GO vectors (GOASVM)	None	**0.9419**	**0.9379**	**94.55%**

Table 9.3 shows the performance of GOASVM using different features and different SVM classifiers on the EU16 dataset. The penalty factor for training the SVMs was set to 0.1 for both linear SVMs and RBF-SVMs. For RBF-SVMs, the kernel parameter was set to 1. For the first four methods, Vector Norm was adopted for better classification performances. GapAA [21] takes the maximum gap length 48 (the minimum length of all the sequences is 50). As AA, PairAA and PseAA produce low-dimensional feature vectors, the performance achieved by RBF-SVMs is better than that achieved by linear SVMs. So we just present the performance of RBF-SVMs. As can be seen, amino-acid composition and its variant are not good features for subcellular localization. The highest accuracy is only 48.66%. Moreover, although homology-based method can achieves better accuracy

(54.52%) than amino-acid composition based methods, the performance is still very poor, probably because of the low sequence similarity in this dataset. On the other hand, GOASVM can achieve a significantly better performance (94.55%), almost 40% (absolute) better than homology-based method. This suggests that gene-ontology based method can provide significantly richer information pertaining to protein subcellular localization than the other methods. The high OMCC and WAMCC also suggest that GOASVM is capable of handling imbalanced classification problems.

9.1.4 Comparing with State-of-the-Art GO Methods

To further demonstrate the superiority of GOASVM over other state-of-the-art GO methods, we also did experiments on the EU16 dataset and the HUM12 dataset, respectively. Table 9.4 compares the performance of GOASVM against three state-of-the-art GO-based methods on the EU16 dataset and the HUM12 dataset, respectively. As Euk-OET-PLoc and Hum-PLoc could not produce valid GO vectors for some proteins in EU16 and HUM12, both methods use PseAA as a backup. ProLoc-GO uses either the ACs of proteins as searching keys or uses the ACs of homologs returned from BLAST as searching keys. GOASVM also uses BLAST to find homologs, but unlike ProLoc-GO, GOASVM uses more than the top-ranked homologs.

Table 9.4 shows that for ProLoc-GO, using ACs as input performs better than using sequences (ACs of homologs) as input. However, the results for GOASVM are not conclusive in this regard because under LOOCV, using ACs as input performs better than using sequences, but the situation is opposite under independent tests. Table 9.4 also shows that no matter using ACs as input or sequences as input, GOASVM performs better than Euk-OET-PLoc and ProLoc-GO, for both the EU16 and HUM12 datasets.

To show that the high performance of GOASVM is not purely attribute to the ho-

Table 9.4: Comparing GOASVM with state-of-the-art GO-based methods on (a) the EU16 dataset and (b) the HUM12 dataset, respectively. *S*: Sequences; *AC*: accession number; LOOCV: leave-one-out cross-validation. $m(n)$: m means the accuracy; n means the WAMCC. See Supplementary Materials for the definition of WAMCC. (–) means the corresponding references do not provide the WAMCC.

Method	Input Data	Feature	Accuracy (WAMCC)	
			LOOCV	Independent Test
ProLoc-GO [72]	S	GO (using BLAST)	86.6% (0.7999)	83.3% (0.706)
ProLoc-GO [72]	AC	GO (No BLAST)	89.0% (–)	85.7% (0.710)
Euk-OET-PLoc [64]	S + AC	GO + PseAA	81.6% (–)	83.7% (–)
GOASVM	S	GO (usig BLAST)	**94.68% (0.9388)**	93.86% (0.9252)
GOASVM	AC	GO (No BLAST)	94.55% (0.9379)	**94.61% (0.9348)**
BLAST [60]	S	–	56.75% (–)	60.39% (–)

(a) Performance on the EU16 dataset

Method	Input Data	Feature	Accuracy (WAMCC)	
			LOOCV	Independent Test
ProLoc-GO [72]	S	GO (using BLAST)	90.0% (0.822)	88.1% (0.661)
ProLoc-GO [72]	AC	GO (No BLAST)	91.1% (–)	90.6% (0.724)
Hum-PLoc [85]	S + AC	GO + PseAA	81.1% (–)	85.0% (–)
GOASVM	S	GO (usig BLAST)	**91.73% (0.9033)**	94.21% (0.9346)
GOASVM	AC	GO (No BLAST)	91.51% (0.9021)	**94.39% (0.9367)**
BLAST [60]	S	–	68.55% (–)	65.69% (–)

(b) Performance on the HUM12 dataset

mologous information obtained from BLAST, we used BLAST directly as a subcellular localization predictor. Specifically, the subcellular location of a query protein is determined by the subcellular location of its closest homolog as determined by BLAST using Swiss-Prot 2012_04 as the protein database. The subcellular location of the homologs were obtained from their CC field in Swiss-Prot. Results in Table 9.4 show that the performance of this approach is significantly poorer than that of other machine learning approaches, suggesting that homologous information alone is not sufficient for subcellular localization prediction. [109] also used BLAST to find the subcellular locations of

Table 9.5: Performance of GOASVM based on different versions of the GOA database on the EU16 training dataset. The 2nd column specifies the publication year of the GOA database being used for constructing the GO vectors. For proteins without a GO term in the GOA database, pseudo amino-acid composition (PseAA) was used as the backup feature. When the latest GOA database is used (last row), only one protein in the dataset does not have a GO term. Therefore, we assigned '0' to all of the elements in the GO vector of this protein instead of using PseAA. *LOOCV*: leave-one-out cross validation. Note for fair comparison, GOASVM here only uses the ACs as input and thus the backup method is needed.

Method	Feature		Accuracy	
	Main	Backup	LOOCV	Independent Test
Euk-OET-PLoc [64]	GO (GOA2005)	PseAA	81.6%	83.7%
GOASVM	GO (GOA2005)	PseAA	86.42%	89.11%
GOASVM	GO (GOA2011)	–	**94.55%**	**94.61%**

proteins. Their results also suggest that using BLAST alone is not sufficient for reliable prediction.

Although all the datasets mentioned in this paper were cut off at 25% sequence similarity, the performance of GOASVM increased from 72.20% (Table 9.2) on the novel dataset to more than 90% (Table 9.4) on both the EU16 dataset and the HUM12 dataset. This is mainly because in Table 9.4, the training and testing sets were constructed at the same time, whereas there are 6 years apart between the creation of the training set and the testing set in Table 9.2, which causes the latter to have less similarity in GO information between the training set and test sets than the former. This in turn implies that the performance of GOASVM on our novel dataset (Table 9.2) can more objectively reflect the classification capabilities of the predictors.

9.1.5 GOASVM Using Old GOA Databases

The newer the version of GOA database, the more annotation information it contains. To investigate how the updated information affects the performance of GOASVM, we performed experiments using an earlier version (published in Oct. 2005) of the GOA database and compared the results with Euk-OET-PLoc on the EU16 dataset. Comparison between the last and second last rows of Table 9.5 reveals that using newer versions of the GOA database can achieve better performance than using older versions. This suggests that annotation information is very important to the prediction. The results also show that GOASVM significantly outperforms Euk-OET-PLoc, suggesting that the GO vector construction method and classifier (term-frequency and SVM) in GOASVM are superior to the those used in Euk-OET-PLoc (1-0 value and K-NN).

9.2 Performance of FusionSVM

9.2.1 Comparing GO Vector Construction Methods and Normalization Methods

Table 9.6 shows the performance of 12 InterProGOSVM methods using the FusionSVM dataset (3572 proteins). For ease of reference, we label these methods as GO_1, GO_2, . . ., GO_12. Linear SVMs were used in all cases and the penalty factor was also set to 1. When using vector norm or geometric mean to post-process the GO vectors, the inverse sequence-frequency can produce more discriminated GO vectors, as evident in the higher accuracy, OMCC and WAMCC corresponding to GO_6 and GO_10. As there may be quite a few redundant GO terms existing in a lot of protein sequences, using ISF can remove or weaken their impact on final prediction of subcellular locations. Except for ISF, using the raw GO vectors as the SVM input achieves the best performance, as evident in the higher accuracy,

Table 9.6: Performance of InterProGOSVM methods using different approaches to constructing the raw GO vectors and different post-processing approaches to normalizing the raw GO vectors. 'None' in Post-processing means that the raw GO vectors $\mathbf{q}_i$ are used as input to the SVMs. *ISF*: inverse sequence-frequency; *TF*: term-frequency; *TF-ISF*: term-frequency inverse sequence frequency.

Method ID	GO Vectors Construction	Post-processing	ACC	OMCC	WAMCC
GO_1	1-0 value	None	**72.21%**	**0.6943**	**0.6467**
GO_2	ISF	None	71.89%	0.6908	0.6438
GO_3	TF	None	71.99%	0.6919	0.6451
GO_4	TF-ISF	None	71.15%	0.6827	0.6325
GO_5	1-0 value	Vector Norm	71.25%	0.6837	0.6335
GO_6	ISF	Vector Norm	72.02%	0.6922	0.6427
GO_7	TF	Vector Norm	70.96%	0.6806	0.6293
GO_8	TF-ISF	Vector Norm	71.73%	0.6890	0.6386
GO_9	1-0 value	Geometric Mean	70.51%	0.6756	0.6344
GO_10	ISF	Geometric Mean	72.08%	0.6929	0.6468
GO_11	TF	Geometric Mean	70.64%	0.6771	0.6290
GO_12	TF-ISF	Geometric Mean	71.03%	0.6813	0.6391

OMCC and WAMCC corresponding to GO_1, GO_3, and GO_4. This suggests that post-processing could remove some of the subcellular localization information pertaining to the raw GO vectors. GO_1 achieves the best performance, suggesting that post-processing is not necessary.

9.2.2 Performance of PairProSVM

Table 9.7 shows the performance of different SVMs using various features extracted from the protein sequences. The features include amino acid composition (AA) [20], amino-acid pair composition (PairAA) [20], AA composition with the maximum gap length equal to 59 (the minimum length of all of the 3120 sequences is 61) [21], pseudo AA composition [29], profile alignment scores and GO vectors. The penalty factor for training the SVMs was set to 1 for both linear SVM and RBF-SVM. For RBF-SVMs the kernel parameter was set to 1. As AA and PairAA produce low-dimensional feature vectors, the performance achieved by RBF-SVM is better than that of the linear SVM. So, we just present the performance of RBF-SVM.

Table 9.7 shows that amino-acid composition and its variant are not good features for subcellular localization. AA method only explores the amino acid composition information, so it performs the worst. PairAA, GapAA and the extended PseAA extract the sequence-order information, so their combinations achieves a slightly better prediction performance. Among the amino acid based methods, the highest accuracy is only 61.44%. On the other hand, the homology-based method that exploits the homologous sequences in protein databases (via PSI-BLAST) achieves a significant better performance. This suggests that the information pertaining to the amino acid sequences is limited. On the contrary, homology-based method PairProSVM can extract much more valuable information about protein subcellular localization than amino acid based methods. The higher OMCC and WAMCC also suggest that PairProSVM can handle imbalanced problems better. The results also suggest that InterProGOSVM outperforms the amino-acid-composition methods and InterProGOSVM is also comparable, although a bit inferior, to PairProSVM.

9.2.3 Performance of FusionSVM

Table 9.8 shows the performance of fusing the InterProGOSVM and PairProSVM. The performance was obtained by optimizing the fusion weights w^{GO} (based on the test dataset). The results show that the combination of PairProSVM and GO_10 (ISF with geometric mean) achieves the highest accuracy—79.04%, which is significant better than PairProSVM (77.05%) and the InterProGOSVM method (72.21%) alone. The results also

Table 9.7: Comparing different features and different SVM classifiers on the FusionSVM dataset (3572 proteins). Performance obtained by using amino acid composition (AA) [20], amino-acid pair composition (PairAA) [20], AA composition with gap (length = 59) (GapAA) [21], pseudo AA composition (PseAA) [29], and profile alignment scores as feature vectors and different SVMs as classifiers. The last two rows correspond to the PairProSVM proposed in [218] and InterProSVM.

Classifier	Feature	Post-processing	ACC	OMCC	WAMCC
RBF-SVM	AA	Vector Norm	54.29%	0.4972	0.3788
RBF-SVM	AA+PairAA	Vector Norm	56.47%	0.5212	0.4089
Linear SVM	AA+PairAA+GapAA(59)	Vector Norm	61.44%	0.5759	0.4783
RBF-SVM	AA+PseAA	Vector Norm	57.98%	0.5378	0.4297
Linear SVM	Profile Alignment	Geometric Mean	**77.05%**	**0.7476**	**0.7048**
Linear SVM	GO vectors (InterProScan)	None	72.21%	0.6943	0.6467

suggest that fusion of PairProSVM and any configuration of InterProGOSVM can outperform the individual methods. This is mainly because the information obtained from homology search and from functional domain databases has different perspectives and is therefore complementary to each other.

Surprisingly, fusing the best performing InterProGOSVM and profile-alignment method does not give the best performance. And for different fusion methods, the best performance is achieved at different optimal w^{GO}. Since the performance of PairProSVM seems to be a bit better than that of InterProGOSVM, it is reasonable to give less weight to InterProGOSVM and more to PairProSVM.

9.2.4 Correlation between the Weighting Factor and FusionSVM

As mentioned above, the w^{GO} will significantly influence the final performance of each fusion method. It is necessary to discover how the parameter impacts the accuracy of fusion methods. Here, we chose the fusion method with the best performance—GO_10. Fig. 9.1 shows the performance of fusing GO_10 and PairProSVM by varying w^{GO} from

Table 9.8: Performance of the fusion of InterProGOSVM and PairProSVM.

Method I	Optimal w^{GO}	ACC	OMCC	WAMCC
GO_1	0.4490	78.91%	0.7680	0.7322
GO_2	0.2643	78.56%	0.7641	0.7260
GO_3	0.3970	78.75%	0.7662	0.7291
GO_4	0.3693	78.72%	0.7659	0.7285
GO_5	0.3711	78.78%	0.7666	0.7293
GO_6	0.3428	78.78%	0.7666	0.7294
GO_7	0.4263	78.81%	0.7670	0.7289
GO_8	0.2947	78.40%	0.7624	0.7234
GO_9	0.4186	78.97%	0.7687	0.7318
GO_10	0.4515	**79.04%**	**0.7694**	**0.7335**
GO_11	0.3993	78.37%	0.7620	0.7222
GO_12	0.3670	78.62%	0.7648	0.7263

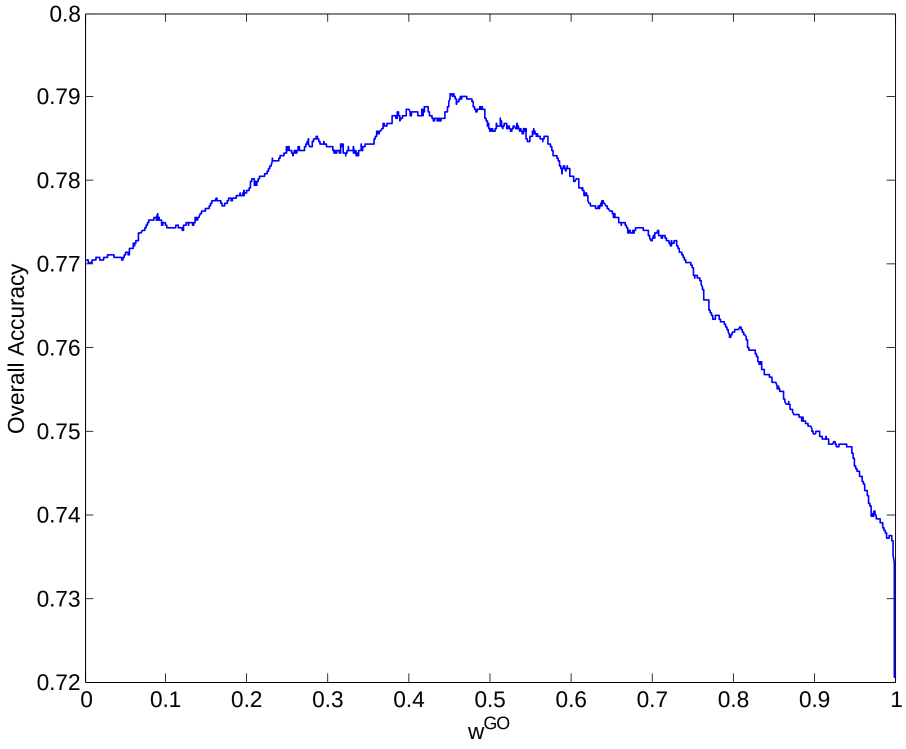


Figure 9.1: Performance of fusing of GO_10 and PairProSVM using different fusion weight w^{GO}.

0 to 1. As can be seen, the performance changes steadily with the change of w^{GO}. It suggests that w^{GO} would not impact the final performance of the fusion method abruptly and the improvement of the fusion method over PairProSVM exists for a wide range of w^{GO}. Further, to show that the improvement of the fusion methods over each individual method is statistically significant, we also performed the McNemar's test [225] on their SVM scores to compare their performance [226]. The p-value between the accuracy of the fusion system (GO_10 and PairProSVM) and the PairProSVM system is 0.0055 ($\ll 0.05$), which suggests that the performance of the fusion predictor is significantly better than that of the PairProSVM predictor.

9.3 Performance of mGOASVM

9.3.1 Comparing with State-of-the-Art Predictors

Table 9.9(a) compares the performance of mGOASVM against three state-of-the-art virus-specialized multi-label predictors on the virus dataset. Both Virus-mPLoc [154] and iLoc-Virus [93] use the accession numbers of homologs returned from BLAST [60] as searching keys to retrieve GO terms from the GOA database. The KNN-SVM ensemble classifier [156] uses the true accession number of proteins directly as input. For a fair comparison with these two predictors, the performance of mGOASVM shown in Table 9.9(a) was obtained by using the accession numbers of homologous proteins as the searching keys. Like Virus-mPLoc and iLoc-Virus, mGOASVM uses BLAST [60] to find the homologs and then uses the accession numbers of the homologs as the searching keys. Here, mGOASVM selects the top homolog for each protein. If BLAST cannot find a homolog for a protein, we assign zeros to all entries of the corresponding GO vectors. In the virus dataset, a homolog can always be found for every protein.

Table 9.9: Comparing mGOASVM with state-of-the-art multi-label predictors based on leave-one-out cross validation (LOOCV) using (a) the virus dataset and (b) the plant dataset, respectively. "–" means the corresponding references do not provide the overall actual accuracy. *KNN-SVM*: the KNN-SVM ensemble classifier proposed in [156]. *SCL*: subcellular locations in the virus dataset, including viral capsid (VC), host cell membrane (HCM), host endoplasmic reticulum (HER), host cytoplasm (HCYT), host nucleus (H-NUC) and secreted (SEC); *OAA*: overall actual accuracy; *OLA*: overall locative accuracy.

Label	SCL	LOOCV Locative Accuracy			
		Virus-mPLoc [154]	KNN-SVM [156]	iLoc-Virus [93]	mGOASVM
1	VC	8/8 = 100.0%	8/8 = 100.0%	8/8 = 100.0%	8/8 = 100.0%
2	HCM	19/33 = 57.6%	27/33 = 81.8%	25/33 = 75.8%	32/33 = 97.0%
3	HER	13/20 = 65.0%	15/20 = 75.0%	15/20 = 75.0%	17/20 = 85.0%
4	HCYT	52/87 = 59.8%	86/87 = 98.8%	64/87 = 73.6%	85/87 = 97.7%
5	HNUC	51/84 = 60.7%	54/84 = 65.1%	70/84 = 83.3%	82/84 = 97.6%
6	SEC	9/20 = 45.0%	13/20 = 65.0%	15/20 = 75.0%	20/20 = 100.0%
OAA		–	–	155/207 =74.8%	184/207 = **88.9%**
OLA		152/252 = 60.3%	203/252 = 80.7%	197/252 = 78.2%	244/252 = **96.8%**

(a) Performance on the viral protein dataset

Label	Subcellular Location	LOOCV Locative Accuracy		
		Plant-mPLoc [96]	iLoc-Plant [87]	mGOASVM
1	Cell membrane	24/56 = 42.9%	39/56 = 69.6%	53/56 = 94.6%
2	Cell wall	8/32 = 25.0%	19/32 = 59.4%	27/32 = 84.4%
3	Chloroplast	248/286 = 86.7%	252/286 = 88.1%	272/286 = 95.1%
4	Cytoplasm	72/182 = 39.6%	114/182 = 62.6%	174/182 = 95.6%
5	Endoplasmic reticulum	17/42 = 40.5%	21/42 = 50.0%	38/42 = 90.5%
6	Extracellular	3/22 = 13.6%	2/22 = 9.1%	22/22 = 100.0%
7	Golgi apparatus	6/21 = 28.6%	16/21 = 76.2%	19/21 = 90.5%
8	Mitochondrion	114/150 = 76.0%	112/150 = 74.7%	150/150 = 100.0%
9	Nucleus	136/152 = 89.5%	140/152 = 92.1%	151/152 = 99.3%
10	Peroxisome	14/21 = 66.7%	6/21 = 28.6%	21/21 = 100.0%
11	Plastid	4/39 = 10.3%	7/39 = 17.9%	39/39 = 100.0%
12	Vacuole	26/52 = 50.0%	28/52 = 53.8%	49/52 = 94.2%
OAA		–	666/978 = 68.1%	855/978 = **87.4%**
OLA		672/1055 = 63.7%	756/1055 = 71.7%	1015/1055 =**96.2%**

(b) Performance on the plant protein dataset

Table 9.9(b) compares the performance of mGOASVM against two state-of-the-art plant-specialized multi-label predictors on the plant dataset. Plant-mPLoc [96] uses similar methods as Virus-mPLoc, and iLoc-Plant [87] uses similar methods as iLoc-Virus.

Here mGOASVM also selects the top homolog for each protein.

As shown in Table 9.9, for the virus dataset, mGOASVM performs significantly better than Virus-mPLoc and iLoc-Virus; for the plant dataset, mGOASVM also performs remarkably better than Plant-mPLoc and iLoc-Plant. In the virus dataset, both the overall locative accuracy and overall actual accuracy of mGOASVM are more than 14% (absolute) higher than iLoc-Virus (96.8% vs 78.2% and 88.9% vs 74.8%, respectively); and in the plant dataset, the corresponding two measures are more than 19% (absolute) higher than iLoc-Plant (96.2% vs 71.7% and 87.4% vs 68.1%, respectively). mGOASVM also performs significantly better than KNN-SVM ensemble classifier in terms of overall locative accuracy (96.8% vs 80.7%); except for the host cytoplasm, mGOASVM is more accurate than KNN-SVM in predicting all subcellular locations. The results on both datasets demonstrate that mGOASVM is more capable of handling multi-label problems than Virus-mPLoc, iLoc-Virus, KNN-SVM ensemble classifier, Plant-mPLoc and iLoc-Plant. As for the individual locative accuracy, in the virus dataset, except for the "viral capsid" for which all of mGOASVM, Virus-mPLoc and iLoc-Virus reach 100%, the locative accuracies of mGOASVM are remarkably higher than those of Virus-mPLoc and iLoc-Virus; while in the plant dataset, the individual locative accuracies of mGOASVM for all of the 12 locations are impressively higher than those of Plant-mPLoc and iLoc-Plant.

9.3.2 Kernel Selection and Optimization

A support vector machine (SVM) can use linear, RBF or polynomial function as its kernel. Some works [227, 21] have demonstrated that RBF kernels achieve better results than linear and polynomial kernels. However, our results show that linear SVMs perform better in our case. Table 9.10 shows the performance of mGOASVM using different types of kernel functions with different parameters based on leave-one-out cross validation

Table 9.10: Performance of mGOASVM using different kernels with different parameters based on leave-one-out cross validation (LOOCV) using the virus dataset. The penalty parameter (C) was set to 0.1 for all cases. σ is the kernel parameter for the RBF SVM; d is the polynomial degree in the Polynomial SVM.

Kernel	Parameter	Locative Accuracy	Actual Accuracy
Linear SVM	–	244/252 = **96.8%**	184/207 = **88.9%**
RBF SVM	$\sigma = 2^{-2}$	182/252 = 72.2%	53/207 = 25.6%
RBF SVM	$\sigma = 2^{-1}$	118/252 = 46.8%	87/207 = 42.0%
RBF SVM	$\sigma = 1$	148/252 = 58.7%	116/207 = 56.0%
RBF SVM	$\sigma = 2^{1}$	189/252 = 75.0%	142/207 = 68.6%
RBF SVM	$\sigma = 2^{2}$	223/252 = 88.5%	154/207 = **74.4%**
RBF SVM	$\sigma = 2^{3}$	231/252 = 91.7%	150/207 = 72.5%
RBF SVM	$\sigma = 2^{4}$	233/252 = **92.5%**	115/207 = 55.6%
RBF SVM	$\sigma = 2^{5}$	136/252 = 54.0%	5/207 = 2.4%
Polynomial SVM	$d = 2$	231/252 = **91.7%**	180/207 = **87.0%**
Polynomial SVM	$d = 3$	230/252 = 91.3%	178/207 = 86.0%

(LOOCV) using the virus dataset. For RBF SVM, the kernel parameter σ was selected from the set $\{2^{-2}, 2^{-1}, \ldots, 2^{5}\}$. For polynomial SVM, the degree of polynomial was set to either 2 or 3. The penalty parameter (C) was set to 0.1 for all cases. Table 9.10 shows that SVMs that use the linear kernel perform better than that with RBF and polynomial kernels. This is plausible because the dimension of GO vectors is larger than the number of training vectors, aggravating the curse of dimensionality problem in non-linear SVMs [228]. The over-fitting problem becomes more severe when the degree of non-linearity is high (small σ), leading to degradation in performance, as demonstrated in Table 4. In other words, highly nonlinear SVMs become vulnerable to overfitting due to the high-dimensionality of the GO vectors.

Table 9.11: Performance of different GO-vector construction methods based on leave-one-out cross validation (LOOCV) for (a) the virus dataset and (b) the plant dataset, respectively.

GO Vector Construction Methods	Locative Accuracy	Actual Accuracy
1-0 value	244/252 = **96.8%**	179/207 = 86.5%
Term-frequency (TF)	244/252 = **96.8%**	184/207 = **88.9%**

(a) Performance on the viral protein dataset

GO Vector Construction Methods	Locative Accuracy	Actual Accuracy
1-0 value	1014/1055 = 96.1%	788/978 = 80.6%
Term-frequency (TF)	1015/1055 = **96.2%**	855/978 = **87.4%**

(b) Performance on the plant protein dataset

9.3.3 Term-Frequency for mGOASVM

Table 9.11 shows the performance of the GO-vector construction methods. Linear SVMs were used in both cases, and the penalty factor was set to 0.1. The results show that term-frequency (TF) achieves a bit better performance than 1-0 value in the locative accuracy, but performs almost 2% and 7% better than 1-0 value in the actual accuracy for the virus dataset and the plant dataset, respectively, which demonstrates that the frequencies of occurrences of GO terms could also provide information for subcellular locations. The results are biologically relevant because proteins of the same subcellular localization are expected to have a similar number of occurrences of the same GO term. In this regard, the 1-0 value approach is inferior because it quantizes the number of occurrences of a GO term to 0 or 1. Moreover, the more remarkable improvement achieved for the plant dataset than that for the virus dataset also suggests that the term-frequency (TF) construction method can boost the performance more impressively for datasets with larger size and more multi-label proteins.

Table 9.12: Distribution of the number of labels predicted by mGOASVM for proteins in the virus and plant datasets, respectively. $|\mathcal{M}(\mathbf{p}_i)|$: Number of predicted labels for the i-th ($i = 1, \ldots, N_{\text{act}}$) protein; $|\mathcal{L}(\mathbf{p}_i)|$: Number of the true labels for the i-th protein; *Over-prediction*: the number of predicted labels is larger than that of the true labels; *Equal-prediction*: the number of predicted labels is equal to that of the true labels; *Under-prediction*: the number of predicted labels is smaller than that of the true labels; n_k^o, n_k^e or n_k^u: the number of proteins that are over-, equal-, or under-predicted by k ($k = 0, \ldots, 5$ for the virus dataset and $k = 0, \ldots, 11$ for the plant dataset) labels, respectively; N^o, N^e or N^u: the total number of proteins that are over-, equal-, or under-predicted, respectively.

Dataset	Condition	Case	n_k^o, n_k^e or n_k^u				$(N^o, N^e \text{ or } N^u)/N_{\text{act}}$
			$k=0$	$k=1$	$k=2$	$k>2$	
Virus	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert > \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Over-prediction	0	18	0	0	18/207 = 8.7%
	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert = \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Equal-prediction	187	0	0	0	187/207 = 90.3%
	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert < \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Under-prediction	0	2	0	0	2/207 = 1.0%
Plant	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert > \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Over-prediction	0	83	2	0	85/978 = 8.7%
	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert = \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Equal-prediction	879	0	0	0	879/978 = 89.9%
	$\lvert\mathcal{M}(\mathbf{p}_i)\rvert < \lvert\mathcal{L}(\mathbf{p}_i)\rvert$	Under-prediction	0	14	0	0	14/978 = 1.4%

9.3.4 Multi-label Properties for mGOASVM

To reveal that the high locative accuracies of mGOASVM are due to the capability of mGOASVM rather than due to over-prediction, we have investigated the distributions of the number of predicted labels in both virus and plant datasets. We consider $|\mathcal{M}(\mathbf{p}_i)|$ and $|\mathcal{L}(\mathbf{p}_i)|$ ($i = 1, \ldots, N_{\text{act}}$) in Eq. 8.14 as the number of predicted labels and the number of true labels for the i-th protein, respectively. The distributions of the number of labels predicted by mGOASVM are shown in Table 9.12. Denote n_k^o, n_k^e or n_k^u as the number of proteins that are over-, equal-, and under-predicted by k ($k = 0, \ldots, 5$ for the virus dataset and $k = 0, \ldots, 11$ for the plant dataset) labels. Also denote N^o, N^e or N^u as the total number of proteins that are over-, equal-, and under-predicted, respectively. Here, over-prediction, equal-prediction and under-prediction are respectively defined as the number of predicted labels that is larger than, equal to, and smaller than the number

Table 9.13: Performance of mGOASVM for multi-location proteins using different inputs on (a) the virus dataset and (b) the plant dataset, respectively. *S*: Sequence; *AC*: Accession Number; #*homo*: Number of homologs used in the experiments; l ($l = 1, \ldots, 3$): Number of co-locations. #*homo*=0 means only the true accession number is used.

Input Data	#*homo*	Actual Accuracy of Protein Groups			Overall Actual Accuracy
		$l = 1$	$l = 2$	$l = 3$	
AC	0	154/165 = 93.3%	34/39 = 87.2%	3/3 = 100%	191/207 = **92.3%**
S	1	148/165 = 89.7%	33/39 = 84.6%	3/3 = 100%	184/207 = 88.9%
S + AC	1	151/165 = 91.5%	34/39 = 87.2%	3/3 = 100%	188/207 = 90.8%

(a) Performance on the viral protein dataset

Input Data	#*homo*	Actual Accuracy of Protein Groups			Overall Actual Accuracy
		$l = 1$	$l = 2$	$l = 3$	
AC	0	813/904 = 89.9%	49/71 = 69.0%	1/3 = 33.3%	863/978 = **88.2%**
S	1	802/904 = 88.7%	52/71 = 73.2%	1/3 = 33.3%	855/978 = 87.4%
S + AC	1	811/904 = 89.7%	47/71 = 66.2%	1/3 = 33.3%	859/978 = 87.8%

(b) Performance on the plant protein dataset

of true labels. Table 9.12 shows that proteins that are over- or under-predicted account for a small percentage of the datasets only (8.7% and 1.0% over- and under-predicted in the virus dataset, 8.7% and 1.4% over- and under-predicted in the plant dataset). Even among the proteins that are over-predicted, most of them are over-predicted by one location only. These include all of the 18 proteins in the virus dataset, and 83 out of 85 in the plant dataset. None of the proteins in the virus dataset are over-predicted by more than 1 location. Only 2 out of 85 proteins in the plant dataset are over-predicted by 2 locations, and none are over-predicted by more than 2 locations. As for under-prediction, all of the under-predicted proteins are only under-predicted by 1 location in both datasets. These results demonstrate that the over-prediction and under-prediction percentages are small, which suggests that mGOASVM can effectively determine the number of subcellular locations of a query protein.

Table 9.13 shows the performance of mGOASVM for multi-location proteins using different inputs. Denote l ($l = 1, \ldots, \mathcal{L}$) as the number of co-locations. As the maximum number of co-locations in both datasets is 3, the individual actual accuracies for l ($l = 1, \ldots, 3$) are shown in Table 9.13. Note that high actual accuracies for $l > 1$ are more difficult to achieve than that for $l = 1$, since not only the number of subcellular locations for a protein should be predicted correctly, but also the subcellular locations should be predicted precisely. As can be seen, mGOASVM achieves high performance not only for single-label proteins (the column corresponding to $l = 1$), but also for multi-label proteins (the columns corresponding to $l = 2$ and $l = 3$). The results demonstrate that mGOASVM can tackle multi-label problems well.

9.3.5 Further Analysis for mGOASVM

Table 9.14 shows the performance of mGOASVM with different inputs and different numbers of homologous proteins for the virus and plant datasets. The input data can be of three possible types: (1) accession number only, (2) sequence only and (3) both accession number and sequence. mGOASVM can extract information from these inputs by producing multiple GO vectors for each protein. Denote $\#homo$ as the number of homologous proteins, where $\#homo \in \{0, 1, 2, 4, 8\}$ for the virus dataset and $\#homo \in \{0, 1, 2\}$ for the plant dataset. For different combinations of inputs and numbers of homologs, the number of distinct GO terms can be different. Typically, the number of distinct GO terms increases with the number of homologs.

Table 9.14 shows that the number of homologs can affect the performance of mGOASVM. The results are biologically relevant because the homologs can provide information about the subcellular locations. However, more homologs may bring redundant or even noisy information, which are detrimental to the prediction accuracy. For example, in the plant dataset, the performance of using one homolog is better than that of using two (87.4%

Table 9.14: Performance of mGOASVM with different inputs and different numbers of homologous proteins for (a) the virus dataset and (b) the plant dataset, respectively. *S*: Sequence; *AC*: Accession Number; *#homo*: Number of homologs used in the experiments; $N_d(GO)$: Number of Distinct GO Terms. *#homo*=0 means only the true accession number is used.

Input Data Type	*#homo*	$N_d(GO)$	Locative Accuracy	Actual Accuracy
AC	0	331	244/252 = **96.8%**	191/207 = **92.3%**
S	1	310	244/252 = **96.8%**	184/207 = **88.9%**
S	2	455	235/252 = 93.3%	178/207 = 86.0%
S	4	664	221/252 = 87.7%	160/207 = 77.3%
S	8	1134	202/252 = 80.2%	130/207 = 62.8%
S + AC	1	334	242/252 = **96.0%**	188/207 = **90.8%**
S + AC	2	460	238/252 = 94.4%	179/207 = 86.5%
S + AC	4	664	230/252 = 91.3%	169/207 = 81.6%
S + AC	8	1134	216/252 = 85.7%	145/207 = 70.1%

(a) Performance on the viral protein dataset

Input Data	*#homo*	$N_d(GO)$	Locative Accuracy	Actual Accuracy
AC	0	1532	1023/1055 = **97.0%**	863/978 = **88.2%**
S	1	1541	1015/1055 = **96.2%**	855/978 = **87.4%**
S	2	1906	907/1055 = 85.8%	617/978 = 63.1%
S + AC	1	1541	1010/1055 = **95.7%**	859/978 = **87.8%**
S + AC	2	1906	949/1055 = 90.0%	684/978 = 70.0%

(b) Performance on the plant protein dataset

vs 63.1%), which in turn suggests that we should limit the number of homologs to avoid bringing irrelevant information. Moreover, as can be seen from Table 9.14, the performance achieved by mGOASVM using sequences with the top homolog are comparable to that of mGOASVM using the accession number only.

Table 9.14 shows that mGOASVM using both sequences and accession numbers performs better than using sequences only, but worse than using accession numbers.

9.3.6 Prediction of Novel Proteins

To further demonstrate the effectiveness of mGOASVM, a plant dataset (See Table 8.8 in Chapter 8) containing novel proteins was constructed. Because the novel proteins were recently added to Swiss-Prot, many of them have not been annotated in the GOA database. As a result, if we used the accession numbers of these proteins to search against the GOA database, the corresponding GO vectors will contain all zeros. This suggests that we should use the ACs of their homologs as the searching keys, i.e., the procedure shown in Fig. 4.3 using sequences as input should be adopted. 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 successive-search strategy procedure (also specified in Section 4.1.2 in Chapter 4) was adopted. For the proteins whose top homologs do not have any GO terms in the GOA database, we used the second-top homolog to find the GO terms; similarly, for the proteins whose top and 2-nd homologs do not have any GO terms, the third-top homolog was used; and so on until all the query proteins can correspond to at least one GO term. In the case where BLAST fails to find any homologs, we used the method PseAA [29] as a back-up. In this dataset, among 175 proteins, 5 of them require to use the backup method.

Because BLAST searches were used in the above procedure, the prediction performance will depend on the closeness (degree of homology) between the training proteins and test proteins. To determine the number of test proteins that are close homologs of the training proteins, we performed a BLAST search for each of the test proteins. The E-value threshold was set to 10 so that none of the proteins in the lists returned from BLAST have E-value larger than 10. Then, we identified the training proteins in the lists based on their accession numbers, and recorded their corresponding E-values.

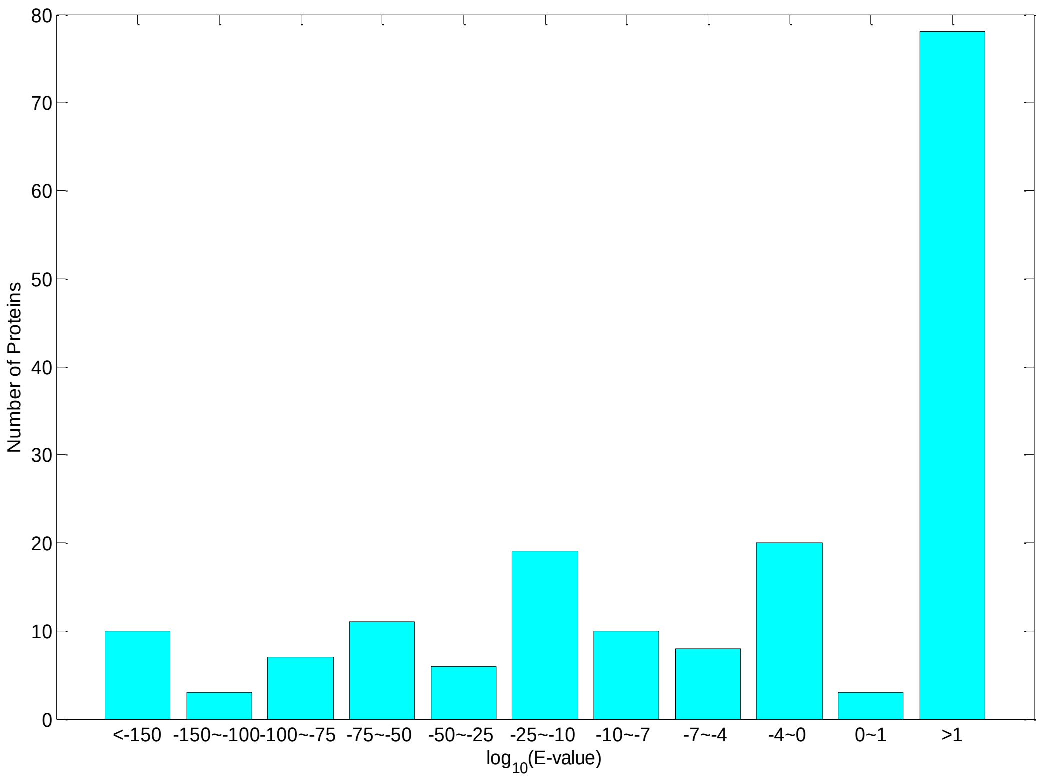


Figure 9.2: **Distribution of the closeness between the new testing proteins and the training proteins.** The *closeness* is defined as the BLAST E-values of the training proteins using the test proteins as the query proteins in the BLAST searches. *Number of Proteins*: The number of testing proteins whose E-values fall into the interval specified under the bar. Small E-values suggest that the corresponding new proteins are close homologs of the training proteins.

Fig. 9.2 shows the distribution of the E-values, which quantify the closeness between the training proteins and test proteins. If we use a common criteria that homologous proteins should have E-value less than 10^{-4}, then 74 out of 175 test proteins are homologs of training proteins, which account for 42% of the test set. Note that this homologous relationship does not mean that using BLAST's homology transfers can predict all of the test proteins correctly. In fact, BLAST's homology transfers (based on the CC field of the homologous proteins) can only achieve a prediction accuracy of 26.9% (47/175). As the

Table 9.15: Comparing mGOASVM with a state-of-the-art multi-label plant predictor based on independent tests using the novel plant dataset.

Label	Subcellular Location	Independent Test Locative Accuracy	
		Plant-mPLoc [96]	mGOASVM
1	Cell membrane	8/16 = 50.0%	7/16 = 43.8%
2	Cell wall	0/1 = 0%	0/1 = 0%
3	Chloroplast	27/54 = 50.0%	39/54 = 72.2%
4	Cytoplasm	5/38 = 13.2%	19/38 = 50.0%
5	Endoplasmic reticulum	1/9 = 11.1%	3/9 = 33.3%
6	Extracellular	0/3 = 0%	1/3 = 33.3%
7	Golgi apparatus	3/7 = 42.9%	3/7 = 42.9%
8	Mitochondrion	6/16 = 37.5%	11/16 = 68.8%
9	Nucleus	31/46 = 67.4%	33/46 = 71.7%
10	Peroxisome	4/6 = 66.7%	3/6 = 50.0%
11	Plastid	0/1 = 0%	0/1 = 0%
12	Vacuole	2/7 = 28.6%	4/7 = 57.1%
Overall Locative Accuracy		87/204 = 42.7%	123/204 =**60.3%**
Overall Actual Accuracy		60/175 = 34.3%	97/175 = **55.4%**

prediction accuracy of mGOASVM on this test set (see Table 9.15) is significantly higher than this percentage, the extra information available from the GOA database plays a very important role in the prediction.

Table 9.15 shows the prediction performance of mGOASVM on this novel protein dataset. As explained earlier, to ensure that these proteins are novel to mGOASVM, 978 proteins of the plant dataset (See Table 8.6 in Chapter 8) were used for training the classifier. We compared mGOASVM with the Plant-mPLoc [96] web-server. The iLoc-Plant [87] web-server is not working properly during testing; so we only reported the performance of the Plant-mPLoc [96] web-server. As shown in Table 9.15, mGOASVM performs significantly better than Plant-mPLoc. The overall locative accuracy and the overall actual accuracy of mGOASVM are more than 17%, 21% higher than those of Plant-

Table 9.16: Prediction results of 10 novel proteins by mGOASVM. *AC*: UniProtKB accession number; *Ground-truth location(s)*: the experimentally-validated actual subcellular location(s) where a protein resides.

AC	Ground-truth Location(s)	Prediction Results	
		Plant-mPLoc [96]	mGOASVM
Q8VYI3	Peroxisome	Chloroplast	Peroxisome
F4JV80	Chloroplast, Mitochondrion	Nucleus	Chloroplast, Mitochondrion
Q93YP7	Mitochondrion	Cell membrane, Chloroplast, Golgi apparatus	Mitochondrion
Q9LK40	Nucleus	Chloroplast	Nucleus
Q6NPS8	Cytoplasm, Nucleus	Endoplasmic reticulum	Cytoplasm, Nucleus
Q3ED65	Chloroplast	Chloroplast, Cytoplasm	Chloroplast
Q9LQC8	Golgi apparatus	Endoplasmic reticulum, Golgi apparatus	Golgi apparatus
Q8VYI1	Endoplasmic reticulum	Endoplasmic reticulum, Vacuole	Endoplasmic reticulum
Q9FNY2	Cytoplasm, Nucleus	Nucleus	Cytoplasm, Nucleus
Q9FJL3	Cytoplasm, Nucleus	Nucleus, Vacuole	Cytoplasm, Nucleus

mPLoc, respectively (locative accuracy 60.3% vs 42.7%, and actual accuracy 55.4% vs 34.3%). For most of 12 individual subcellular locations, mGOASVM outperforms Plant-mPLoc, except in cell membrane and peroxisome. Given the novelty and multi-label properties of these proteins and the low sequence similarity (below 25%), the locative accuracy of 60.3% and the actual accuracy of 55.4% achieved by mGOASVM are fairly high. On the other hand, due to the scarcity of data, mGOASVM does not perform well in some subcellular locations, such as cell wall and plastid. But the situation will be improved when more and more proteins are available for training our SVM classifiers.

To more specifically demonstrate the superiority of mGOASVM over state-of-the-art predictors, prediction results of 10 typical novel proteins by mGOASVM and Plant-mPLoc are shown in Table 9.16. As can be seen, for the single-location protein ‘Q8VY13’, mGOASVM can correctly predict it to be located in 'Peroxisome', while Plant-mPLoc gives a wrong prediction ('Chloroplast'); for multi-location protein 'F4JV80', mGOASVM can correctly predict it to be located in both 'Chloroplast' and 'Mitochondrion', while Plant-mPLoc predicts it to be a single-location protein located in 'Nucleus'. Similarly, for 'Q93YP7', 'Q9LK40' and 'Q6NPS8', mGOASVM can predict them all correctly, while Plant-mPLoc gives all wrong predictions for them. For single-location proteins 'Q3ED65', 'Q9LQC8' and 'Q8VYI1', Plant-mPLoc predicts them partially correctly, but wrongly considers them to be multi-label proteins. On the other hand, mGOASVM correctly predicts them as single-location proteins, located in 'Chloroplast', 'Golgi apparatus' and 'Endoplasmic reticulum', respectively. For 'Q9FNY2' and 'Q9FJL3', Plant-mPLoc also predicts them partially correctly, while mGOASVM can exactly locate both of them in the right subcellular location(s).

9.4 Performance of AD-SVM

9.4.1 Effect of Adaptive Decisions on AD-SVM

Fig. 9.3 shows the performance of AD-SVM on the virus dataset and the plant dataset with respect to the adaptive-decision parameter θ (Eq. 5.11 in Section 5.4.2 of Chapter 5) based on leave-one-out cross-validation. As can be seen, for the virus dataset, as θ increases from 0.0 to 1.0, the overall actual accuracy increases first, reaches the peak at $\theta = 0.3$ (with an actual accuracy of 93.2%), and then decreases. An analysis of the predicted labels $\{\mathcal{L}(\mathbf{P}_i); i = 1, \ldots, N\}$ suggests that the increases in actual accuracy is due to the reduction in the number of over-prediction, i.e., the number of cases where $|\mathcal{M}(\mathbf{P}_i)| > |\mathcal{L}(\mathbf{P}_i)|$ has been reduced. When $\theta > 0.3$, the benefit of reducing the over-prediction diminishes because the criterion in Eq. 5.9 becomes so stringent that some of the proteins were

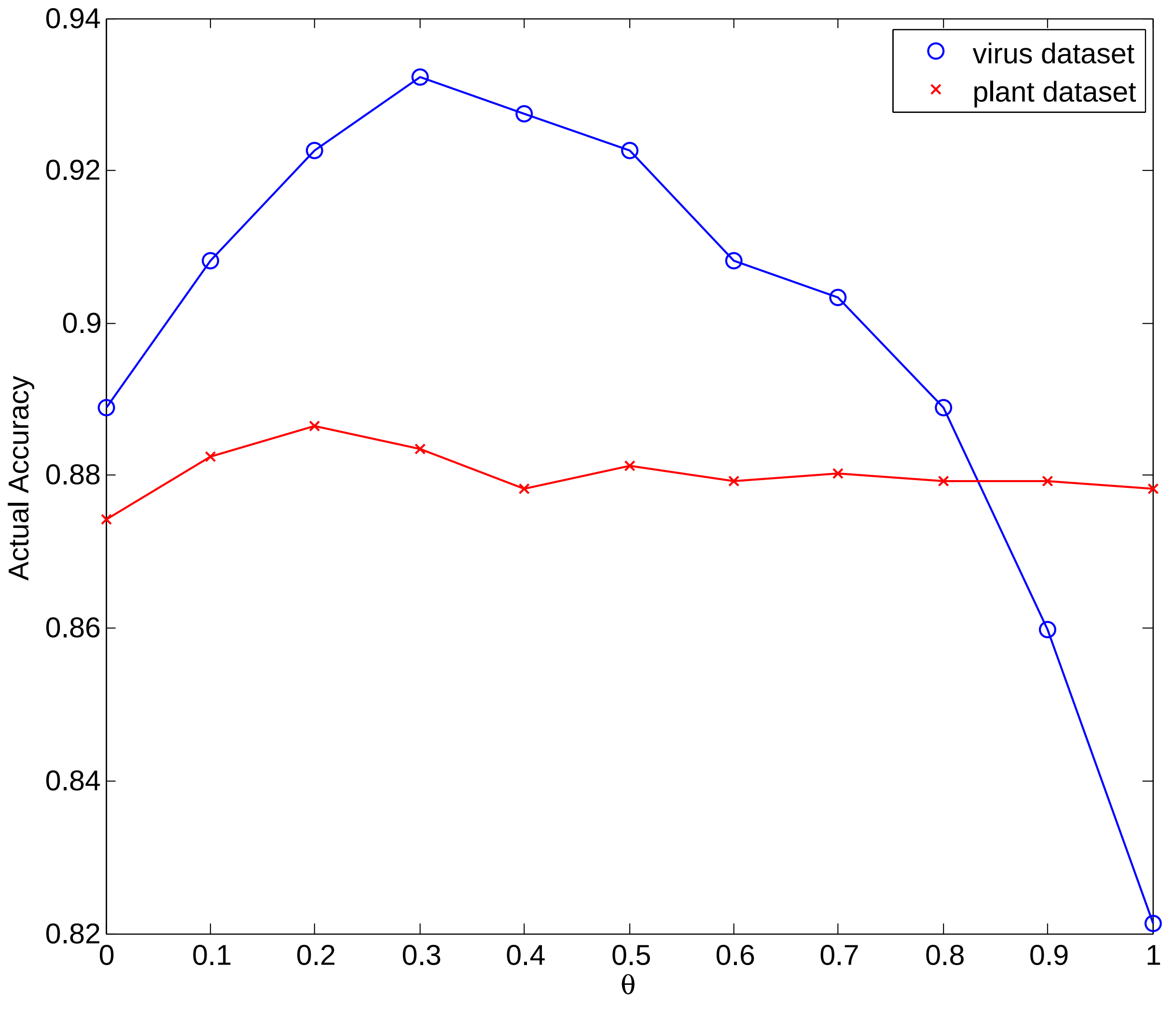


Figure 9.3: Performance of AD-SVM based on leave-one-out cross-validation (LOOCV) varying with θ using the virus and plant datasets, respectively. $\theta = 0$ represents the performance of mGOASVM.

under-predicted, i.e., the number of cases where $|\mathcal{M}(\mathbf{P}_i)| < |\mathcal{L}(\mathbf{P}_i)|$ increases. Note that the performance at $\theta = 0.0$ is equivalent to the performance of mGOASVM, and that the best actual accuracy (93.2% when $\theta = 0.3$) obtained by the proposed decision scheme is more than 4% (absolute) higher than mGOASVM (88.9%).

For the plant dataset, when θ increases from 0.0 to 1.0, the overall actual accuracy

Table 9.17: Comparing AD-SVM with mGOASVM based on leave-one-out cross validation (LOOCV) using the virus dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM	AD-SVM
1	Viral capsid	8/8 = 1.000	8/8 = 1.000
2	Host cell membrane	32/33 = 0.970	32/33 = 0.970
3	Host ER	17/20 = 0.850	17/20 = 0.850
4	Host cytoplasm	85/87 = 0.977	83/87 = 0.954
5	Host nucleus	82/84 = 0.976	82/84 = 0.976
6	Secreted	20/20 = 1.000	20/20 = 1.000
Overall Locative Accuracy (*OLA*)		244/252 = **0.968**	242/252 = 0.960
Overall Actual Accuracy (*OAA*)		184/207 = 0.889	193/207 = **0.932**
Accuracy		0.935	**0.953**
Precision		0.939	**0.960**
Recall		**0.973**	0.966
F1		0.950	**0.960**
HL		0.026	**0.019**

increases from 87.4%, and then fluctuates around 88%. If we take the same θ as that for the virus dataset, i.e., $\theta = 0.3$, the performance of AD-SVM is 88.3%, which is still better than that of mGOASVM at $\theta = 0.0$.

9.4.2 Comparing AD-SVM with mGOASVM

Table 9.17 and Table 9.18 compare the performance of AD-SVM against mGOASVM on the virus and plant dataset. Both of the predictors use the information of GO terms as features. mGOASVM uses a multi-label SVM classifier; and AD-SVM uses a multi-label SVM classifier incorporated with the proposed adaptive decision scheme.

As shown in Table 9.17, although the *OLA* of AD-SVM is slightly smaller than that of mGOASVM, the *OAA* of AD-SVM is more than 4% (absolute) higher than that of mGOASVM. In terms of *Accuracy, Precision, F1* and *HL*, AD-SVM performs better than

Table 9.18: Comparing AD-SVM with mGOASVM based on leave-one-out cross validation (LOOCV) using the plant dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM	AD-SVM
1	Cell membrane	53/56 = 0.946	52/56 = 0.929
2	Cell wall	27/32 = 0.844	27/32 = 0.844
3	Chloroplast	272/286 = 0.951	271/286 = 0.948
4	Cytoplasm	174/182 = 0.956	167/182 = 0.917
5	Endoplasmic reticulum	38/42 = 0.905	38/42 = 0.905
6	Extracellular	22/22 = 1.000	22/22 = 1.000
7	Golgi apparatus	19/21 = 0.905	19/21 = 0.905
8	Mitochondrion	150/150 = 1.000	149/150 = 0.993
9	Nucleus	151/152 = 0.993	148/152 = 0.974
10	Peroxisome	21/21 = 1.000	21/21 = 1.000
11	Plastid	39/39 = 1.000	36/39 = 0.923
12	Vacuole	49/52 = 0.942	48/52 = 0.923
Overall Locative Accuracy (*OLA*)		1015/1055 =**0.962**	998/1055 = 0.946
Overall Actual Accuracy (*OAA*)		855/978 = 0.874	867/978 = **0.887**
	Accuracy	0.926	**0.928**
	Precision	0.933	**0.941**
	Recall	**0.968**	0.956
	F1	**0.942**	**0.942**
	HL	**0.013**	**0.013**

mGOASVM. In terms of *Recall*, mGOASVM performs the better. This is understandable because according to the analysis in the Section 5.4.2, the *Recall* decreases when θ increases. The results suggest that the multi-label SVM classifiers using the proposed adaptive decision scheme perform better than the state-of-the-art classifiers. The individual locative accuracies of AD-SVM are also comparable to mGOASVM.

Similar conclusions can be drawn from Table 9.18, where the superiority of AD-SVM over mGOASVM seems to be not so obvious compared to that in Table 9.17.

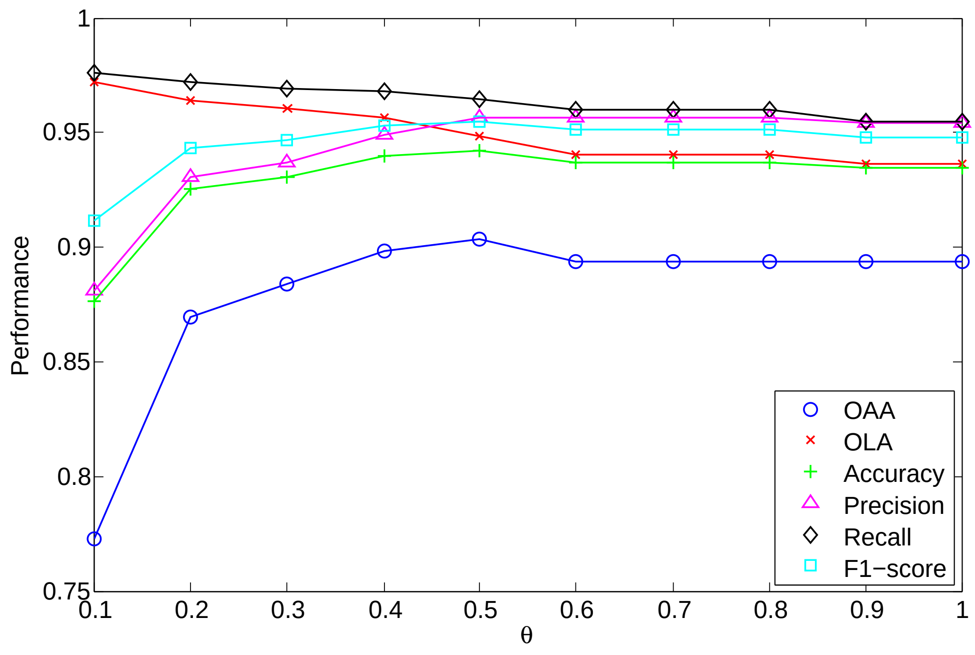
Comprehensive comparisons of all related multi-label predictors can be found in Section 9.10.

9.5 Performance of mPLR-Loc

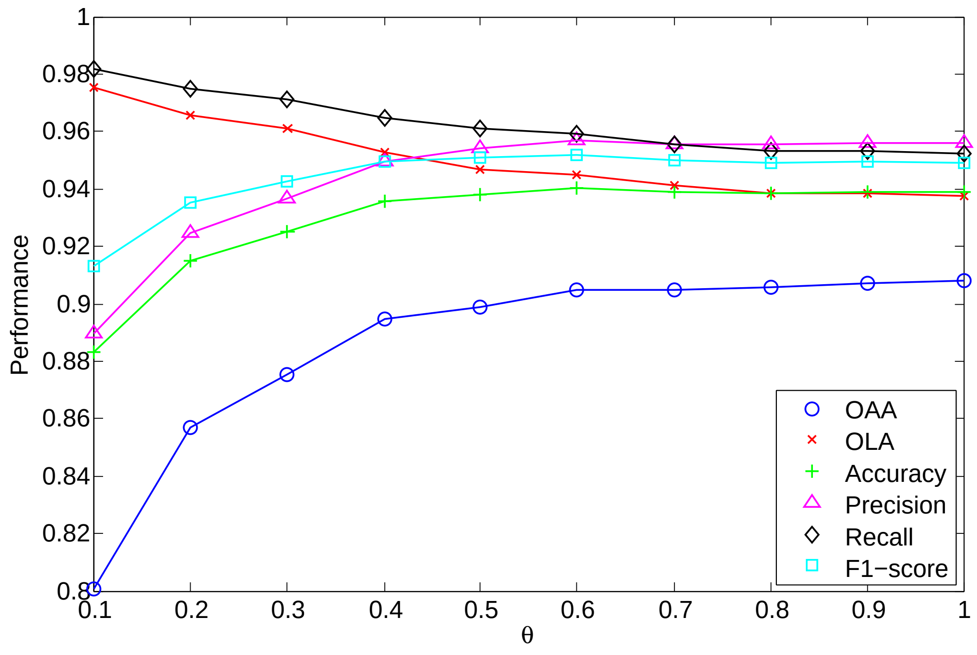
9.5.1 Effect of Adaptive Decisions on mPLR-Loc

Fig. 9.4(a) shows the performance of mPLR-Loc on the virus dataset for different values of θ (Eq. 7.9) based on leave-one-out cross-validation. In all cases, the penalty parameter ρ of logistic regression was set to 1.0. The performance of mPLR-Loc at $\theta = 0.0$ is not provided because according to Eq. 5.20 and Eq. 7.9, all of the query proteins will be predicted as having all of the M subcellular locations, which defeats the purpose of prediction. As evident from Fig. 9.4(a), when θ increases from 0.1 to 1.0, the *OAA* of mPLR-Loc increases first, reaches the peak at $\theta = 0.5$, with $OAA = 0.903$, which is almost 2% (absolute) higher than mGOASVM (0.889). The *Precision* achieved by mPLR-Loc increases until $\theta = 0.5$ and then remains almost unchanged when $\theta \geq 0.5$. On the contrary, *OLA* and *Recall* peak at $\theta = 0.1$, and these measures drop with θ until $\theta = 1.0$. Among these metrics, no matter how θ changes, *OAA* is no higher than other five measurements.

An analysis of the predicted labels $\{\mathcal{L}(\mathbb{Q}_i); i = 1, \ldots, 207\}$ suggests that the increase in *OAA* is due to the reduction in the number of over-prediction, i.e., the number of cases where $|\mathcal{M}(\mathbb{Q}_i)| > |\mathcal{L}(\mathbb{Q}_i)|$. When $\theta > 0.5$, the benefit of reducing the over-prediction diminishes because the criterion in Eq. 5.20 becomes so stringent that some of the proteins were under-predicted, i.e., the number of cases where $|\mathcal{M}(\mathbb{Q}_i)| < |\mathcal{L}(\mathbb{Q}_i)|$. When θ increases from 0.1 to 0.5, the number of cases where $|\mathcal{M}(\mathbb{Q}_i)| > |\mathcal{L}(\mathbb{Q}_i)|$ decreases while at the same time $|\mathcal{M}(\mathbb{Q}_i) \cap \mathcal{L}(\mathbb{Q}_i)|$ remains almost unchanged. In other words, the denominators of *Accuracy* and *F1-score* decrease while the numerators for both metrics remain almost unchanged, leading to better performance for both metrics. When $\theta > 0.5$, for the similar



(a) on the virus dataset



(b) on the plant dataset

Figure 9.4: Performance of mPLR-Loc with respect to θ based on leave-one-out cross-validation on (a) the virus dataset and (b) the plant dataset, respectively. See Eqs. 8.9–8.15 for the definitions of the performance measures in the legend.

reason mentioned above, the increase in under-prediction outweighs the benefit of the reduction in over-prediction, causing performance loss. For *Precision*, when $\theta > 0.5$, the loss due to the stringent criterion is counteracted by the gain due to the reduction in $|\mathcal{M}(\mathbb{Q}_i)|$, the denominator of Eq. 8.10. Thus, the *Precision* increases monotonically when θ increases from 0.1 to 1. However, *OLA* and *Recall* decrease monotonically with respect to θ because the denominator of these measures (see Eqs. 8.14 and 8.11) is independent of $|\mathcal{M}(\mathbb{Q}_i)|$ and the number of correctly predicted labels in the numerator decreases when the decision criterion is getting stricter.

Fig. 9.4(b) show the performance of mPLR-Loc (with $\rho = 1$) on the plant dataset. Fig. 9.4(b) shows that the trends of *OLA*, *Accuracy*, *Precision*, *Recall* and *F1-score* are similar to those of mPLR-Loc in the virus dataset. The figure also shows that the *OAA* achieved by mPLR-Loc is monotonically increasing with respect to θ and reaches the optimum at $\theta = 1.0$, which is in contrast to the results in the virus dataset where the *OAA* is almost unchanged when $\theta \geq 0.5$.

9.5.2 Effect of Regularization on mPLR-Loc

Fig. 9.5 shows the performance of mPLR-Loc with respect to the parameter ρ (Eq. 5.18) on the virus dataset. In all cases, the adaptive thresholding parameter θ was set to 0.8. As can be seen, the variations of *OAA*, *Accuracy*, *Precision* and *F1-score* with respect to ρ are very similar. More importantly, all of these four metrics show that there is a wide range of ρ for which the performance is optimal. This suggests that introducing the penalty term in Eq. 5.13 not only helps to avoid numerical difficulty, but also improves performance.

Fig. 9.5 shows that the *OLA* and *Recall* are largely unaffected by the change in ρ. This is understandable because the parameter ρ is to overcome numerical difficulty when

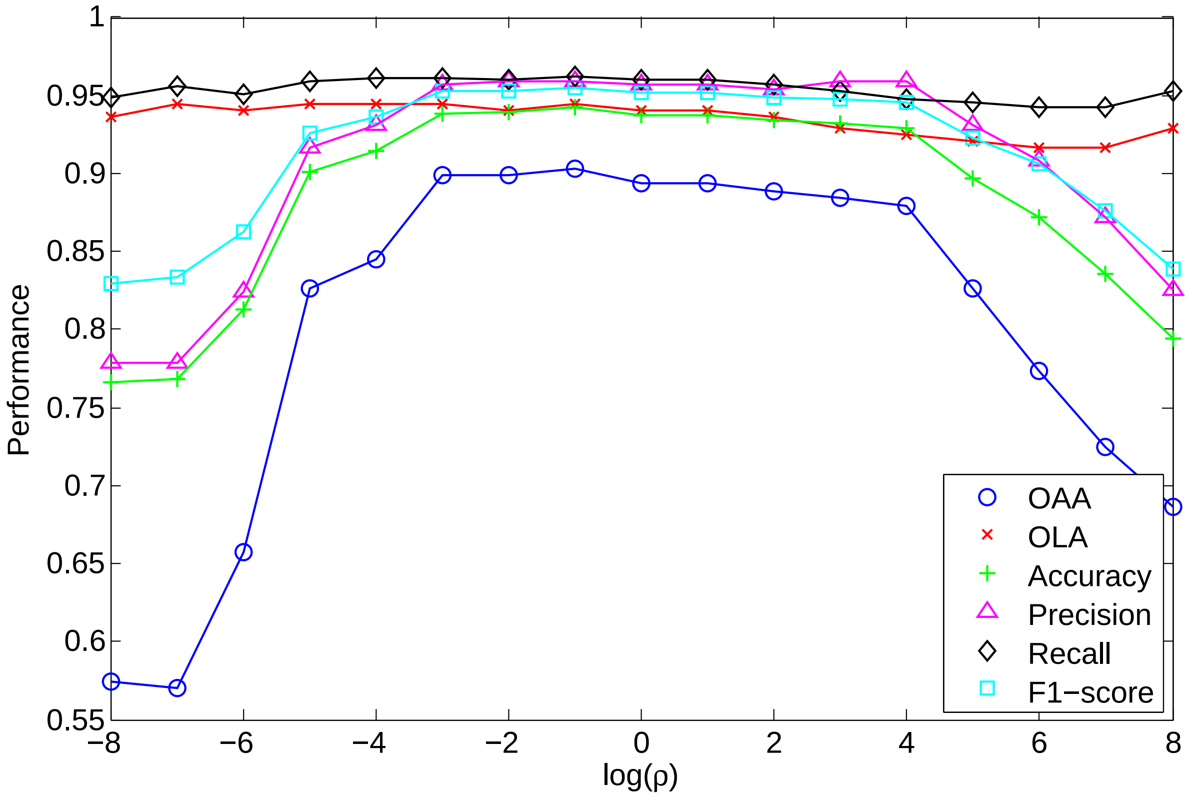


Figure 9.5: Performance of mPLR-Loc with respect to ρ in Eq. 5.18 based on leave-one-out cross-validation on the virus dataset. See Eqs. 8.9–8.15 for the definitions of the performance measures in the legend.

estimating the LR parameters $\boldsymbol{\beta}$. More specifically, when ρ is small (say $\log(\rho) < -5$), the value of ρ is insufficient to avoid matrix singularity in Eq. 5.17, which leads to extremely poor performance. When ρ is too large (say $\log(\rho) > 5$), the matrix in Eq. 5.16 will be dominated by the value of ρ, which also causes poor performance. The *OAA* of mPLR-Loc reaches its maximum 0.903 at $\log(\rho) = -1$.

9.5.3 Comparing mPLR-Loc with mGOASVM

Table 9.19 and Table 9.20 compare the performance of mPLR-Loc against mGOASVM on the virus and plant dataset. Both of these predictors derive the feature vectors from

Table 9.19: Comparing mPLR-Loc with mGOASVM based on leave-one-out cross validation using the virus dataset. "–" means the corresponding references do not provide the related metrics. *Host ER*: Host endoplasmic reticulum. See Eqs. 8.9–8.15 for the definitions of the performance measures.

Label	Subcellular Location	LOOCV Locative Accuracy	
		mGOASVM	mPLR-Loc
1	Viral capsid	8/8 = 1.000	8/8 = 1.000
2	Host cell membrane	32/33 = 0.970	30/33 = 0.909
3	Host ER	17/20 = 0.850	17/20 = 0.850
4	Host cytoplasm	85/87 = 0.977	86/87 = 0.989
5	Host nucleus	82/84 = 0.976	81/84 = 0.964
6	Secreted	20/20 = 1.000	17/20 = 0.850
Overall Actual Accuracy (*OAA*)		184/207 = 0.889	187/207 = **0.903**
Overall Locative Accuracy (*OLA*)		244/252 = **0.968**	239/252 = 0.948
	Accuracy	0.935	**0.942**
	Precision	0.939	**0.957**
	Recall	**0.973**	0.965
	F1	0.950	**0.955**
	HL	0.026	**0.023**

GO terms. mGOASVM uses a multi-label SVM classifier; and the mPLR-Loc uses a multi-label penalized logistic regression classifier incorporated with the proposed adaptive decision scheme.

As shown in Table 9.19, although the *OLA* of mPLR-Loc is slightly smaller than that of mGOASVM, the *OAA* of mPLR-Loc is 2% (absolute) higher than that of mGOASVM. In terms of *Accuracy, Precision, F1* and *HL*, mPLR-Loc performs better than mGOASVM. In terms of *Recall*, mGOASVM performs the best among all the predictors. This is understandable because according to the analysis in the Section 9.5.1, the *Recall* decreases when θ increases. The results suggest that the mPLR-Loc performs better than the state-of-the-art classifiers. The individual locative accuracies of mPLR-Loc are also comparable to mGOASVM.

Table 9.20: Comparing mPLR-Loc with mGOASVM based on leave-one-out cross validation using the plant dataset. "–" means the corresponding references do not provide the related metrics. See Eqs. 8.9–8.15 for the definitions of the performance measures.

Label	Subcellular Location	LOOCV Locative Accuracy	
		mGOASVM	mPLR-Loc
1	Cell membrane	53/56 = 0.946	50/56 = 0.893
2	Cell wall	27/32 = 0.844	25/32 = 0.781
3	Chloroplast	272/286 = 0.951	281/286 = 0.983
4	Cytoplasm	174/182 = 0.956	164/182 = 0.901
5	Endoplasmic reticulum	38/42 = 0.905	35/42 = 0.833
6	Extracellular	22/22 = 1.000	19/22 = 0.864
7	Golgi apparatus	19/21 = 0.905	18/21 = 0.857
8	Mitochondrion	150/150 = 1.000	149/150 = 0.993
9	Nucleus	151/152 = 0.993	146/152 = 0.961
10	Peroxisome	21/21 = 1.000	21/21 = 1.000
11	Plastid	39/39 = 1.000	36/39 = 0.923
12	Vacuole	49/52 = 0.942	45/52 = 0.942
Overall Actual Accuracy (*OAA*)		855/978 = 0.874	888/978 = **0.908**
Overall Locative Accuracy (*OLA*)		1015/1055 =**0.962**	989/1055 = 0.937
	Accuracy	0.926	**0.939**
	Precision	0.933	**0.956**
	Recall	**0.968**	0.952
	F1	0.942	**0.949**
	HL	0.013	**0.010**

Similar conclusions can be drawn from Table 9.20, where the superiority of mPLR-Loc over mGOASVM is more evident compared to that in Table 9.19.

Comprehensive comparisons of related multi-label predictors can be found in Section 9.10.

9.6 Performance of SS-Loc

Table 9.21 compares the performance of SS-Loc against mGOASVM on the plant dataset based on leave-one-out cross validation (LOOCV). For a fair comparison, the performance

Table 9.21: Comparing SS-Loc with mGOASVM based on leave-one-out cross validation (LOOCV).

Label	Subcellular Location	LOOCV Locative Accuracy	
		mGOASVM	SS-Loc
1	Cell membrane	53/56 = 94.6%	55/56 = 98.2%
2	Cell wall	27/32 = 84.4%	28/32 = 87.5%
3	Chloroplast	272/286 = 95.1%	285/286 = 99.7%
4	Cytoplasm	174/182 = 95.6%	175/182 = 96.2%
5	Endoplasmic reticulum	38/42 = 90.5%	40/42 = 95.2%
6	Extracellular	22/22 = 100.0%	22/22 = 100.0%
7	Golgi apparatus	19/21 = 90.5%	18/21 = 85.7%
8	Mitochondrion	150/150 = 100.0%	150/150 = 100.0%
9	Nucleus	151/152 = 99.3%	150/152 = 98.7%
10	Peroxisome	21/21 = 100.0%	21/21 = 100.0%
11	Plastid	39/39 = 100.0%	39/39 = 100.0%
12	Vacuole	49/52 = 94.2%	50/52 = 96.2%
Overall Locative Accuracy		1015/1055 =96.2%	1033/1055 =**97.9%**
Overall Actual Accuracy		855/978 = 87.4%	876/978 = **89.6%**

of both predictors shown in Table 9.9 were obtained by using the accession numbers of homologous proteins as the searching keys. Specifically, the ACs of the homologous proteins, as returned from BLAST search, will be successively used to search against the GOA database until a match is found (See Fig 4.1 for details).

As shown in Table 9.21, SS-Loc performs better than mGOASVM in terms of both the overall actual accuracy (89.6% vs 97.4%) and the overall locative accuracy (97.9% vs 96.2%). As for the individual locative accuracy, the individual locative accuracies of our proposed predictor for all of the 12 locations are impressively higher than those of mGOASVM.

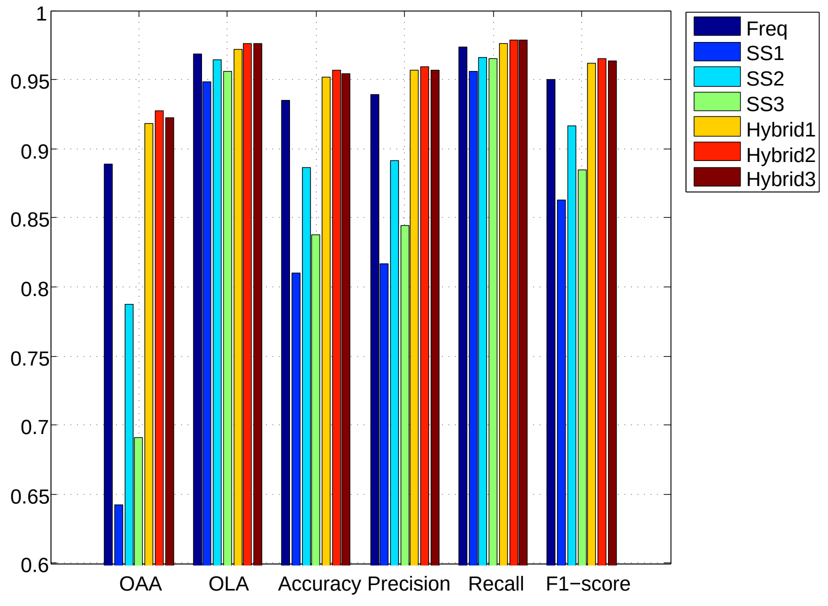
In terms of GO information extraction, mGOASVM only exploits the occurrences of GO terms, whereas SS-Loc discovers the semantic relationships between GO terms, based on which the semantic similarity between proteins (from the GO annotation perspective) can be obtained. The superior performance of SS-Loc clearly suggests that the semantic similarity over Gene Ontology is conducive to the prediction of multi-label protein subcellular localization.

9.7 Performance of HybridGO-Loc

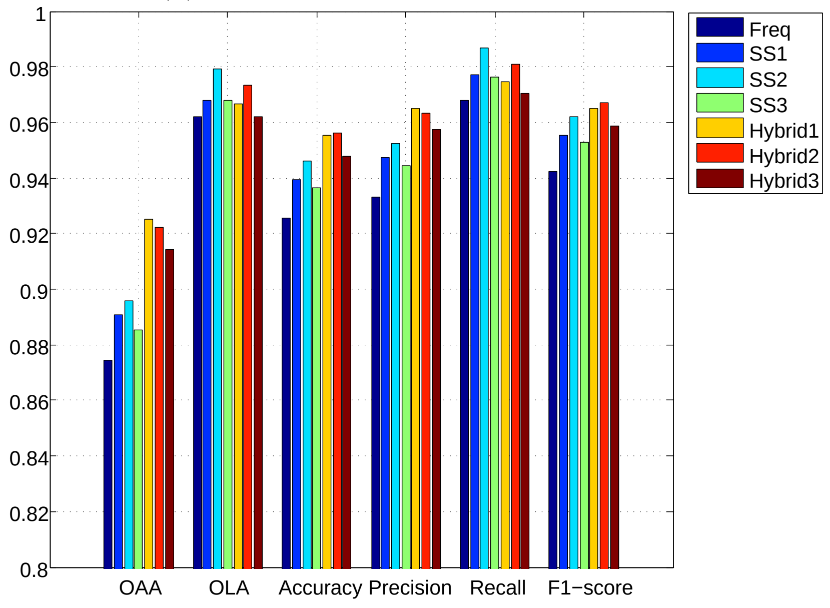
9.7.1 Comparing Different Features

Fig. 9.6(a) shows the performance of individual and hybridized GO features on the virus dataset based on leave-one-out cross validation (LOOCV). In the figure, *SS1, SS2 and SS3* represent Lin's, Jiang's and RS similarity measures, respectively. *Hybrid1*, *Hybrid2* and *Hybrid3* represent the hybridized features obtained from these measures. As can be seen, in terms of all the six performance metrics, the performance of the hybrid features is remarkably better than the performance of individual features, regardless of which of the GO frequency features or the three GO SS features were used. Specifically, the OAAs (the most stringent and objective metric) of all of the three hybrid features are at least 3% (absolute) higher than that of the individual features, which suggests that hybridizing the two features can significantly boost the prediction performance. Moreover, among the hybridized features, the performance of *Hybrid2*, namely combining GO frequency features and GO SS features by Jiang's measure, outperforms *Hybrid1* and *Hybrid3*. Another interesting thing is that although all of the individual GO SS features perform much worse than the GO frequency features, the performance of the three hybridized features is still better that of any of the individual features. This suggests that the GO frequency features and SS features are complementary to each other.

Similar conclusions can be drawn from the plant dataset shown in Fig. 9.6(b). Howev-



(a) Performance on the virus dataset



(b) Performance on the plant dataset

Figure 9.6: **Performance of the hybrid features and individual features on the (a) virus and (b) plant datasets, respectively**. *Freq*: GO frequency features; *SS1, SS2* and *SS3*: GO semantic similarity features by using Lin's measure [174], Jiang's measure [192] and RS measure [175], respectively; *Hybrid1, Hybrid2* and *Hybrid3*: GO hybrid features by combining GO frequency features with GO semantic similarity features based on *SS1, SS2* and *SS3*, respectively.

er, comparison between Fig. 9.6(a) and Fig. 9.6(b) reveals that for the plant dataset, the performance of hybridized features outperforms all of the individual features in terms of all metrics except *OLA* and *Recall*, while for the virus dataset, the former is superior to the latter in terms of all metrics. However, the losses in these two metrics do not outweigh the significant improvement on other metrics, especially on *OAA*, which has around 3% (absolute) improvement in terms of hybridized features as opposed to using individual features. Among the hybridizing features, *Hybrid2* also outperforms *Hybrid1* and *Hybrid3* in terms of *OLA, Accuracy, Recall* and *F1-score*, whereas *Hybrid1* performs better than others in terms of *OAA* and *Precision*. These results demonstrate that the GO SS features obtained by Lin's measure and Jiang's measure are better candidates than the RS measure for combining with the GO frequency features; however, there is no evidence suggesting which measure is better. It is also interesting to see that the performance of the three individual GO SS features is better than that of GO frequency features, in contrary to the results shown in Fig 9.6(a).

9.7.2 Comparing HybridGO-Loc with mGOASVM

Table 9.22 and Table 9.23 compare the performance of HybridGO-Loc against mGOASVM on the virus and plant dataset based on leave-one-out cross validation. Note that we used the best performing hybridizing features with the adaptive decision strategy. Specifically, for both the virus and plant datasets, the best performance was achieved when *Hybrid2* and the adaptive decision strategy with $\theta = 0.3$ were used. θ was determined by cross-validation as stated previously. Unless stated otherwise, we used *Hybrid2* to represent HybridGO-Loc in subsequent experiments. Our proposed predictor use the GO frequency features and GO semantic similarity features, whereas other predictors use only the GO frequency of occurrences as features. From the classification perspective, mGOASVM

Table 9.22: Comparing HybridGO-Loc with mGOASVM based on leave-one-out cross validation (LOOCV) using the virus dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM	HybridGO-Loc
1	Viral capsid	8/8 = 1.000	8/8 = 1.000
2	Host cell membrane	32/33 = 0.970	32/33 = 0.970
3	Host endoplasmic reticulum	17/20 = 0.850	18/20 = 0.900
4	Host cytoplasm	85/87 = 0.977	85/87 = 0.966
5	Host nucleus	82/84 = 0.976	82/84 = 0.988
6	Secreted	20/20 = 1.000	20/20 = 1.000
Overall Locative Accuracy (*OLA*)		244/252 = 0.968	245/252 = **0.972**
Overall Actual Accuracy (*OAA*)		184/207 = 0.889	194/207 = **0.937**
Accuracy		0.935	**0.961**
Precision		0.939	**0.965**
Recall		0.973	**0.976**
F1		0.950	**0.968**
HL		0.026	**0.016**

[108] uses a multi-label SVM classifier; and HybridGO-Loc uses a multi-label SVM classifier incorporated with the adaptive decision scheme.

As shown in Table 9.22, HybridGO-Loc performs remarkably better than mGOASVM in all of the performance metrics, especially for the *OAA* (0.937 vs 0.889). These results demonstrate that hybridizing the GO frequency features and GO SS features can significantly boost prediction performance, which also suggests that these two kinds of information are proved to be complementary to each other in terms of predicting subcellular localization. Similar conclusions can be drawn for the plant dataset from Table 9.23 except that the *OLA* of the proposed predictor is slightly worse than that of mGOASVM, and the *Recall* is equivalent to that of mGOASVM. Nevertheless, the small losses do not outweigh the impressive improvement in the other metrics, especially in the *OAA* (0.936 vs 0.874).

Table 9.23: Comparing HybridGO-Loc with mGOASVM based on leave-one-out cross validation (LOOCV) using the plant dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM [108]	HybridGO-Loc
1	Cell membrane	53/56 = 0.946	51/56 = 0.911
2	Cell wall	27/32 = 0.844	28/32 = 0.875
3	Chloroplast	272/286 = 0.951	278/286 = 0.972
4	Cytoplasm	174/182 = 0.956	168/182 = 0.923
5	Endoplasmic reticulum	38/42 = 0.905	38/42 = 0.905
6	Extracellular	22/22 = 1.000	21/22 = 0.955
7	Golgi apparatus	19/21 = 0.905	19/21 = 0.905
8	Mitochondrion	150/150 = 1.000	149/150 = 0.993
9	Nucleus	151/152 = 0.993	150/152 = 0.987
10	Peroxisome	21/21 = 1.000	21/21 = 1.000
11	Plastid	39/39 = 1.000	38/39 = 0.974
12	Vacuole	49/52 = 0.942	48/52 = 0.923
Overall Locative Accuracy (*OLA*)		1015/1055 =**0.962**	1009/1055 = 0.956
Overall Actual Accuracy (*OAA*)		855/978 = 0.874	915/978 = **0.936**
	Accuracy	0.926	**0.959**
	Precision	0.933	**0.972**
	Recall	**0.968**	**0.968**
	F1	0.942	**0.966**
	HL	0.013	**0.007**

Comprehensive comparisons of all related multi-label predictors can be found in Section 9.10.

9.7.3 Prediction of Novel Proteins

To further demonstrate the effectiveness of HybridGO-Loc, the novel plant dataset (See Table 8.8 in Chapter 8) was used to compare with state-of-the-art multi-label predictors using independent tests. Specifically, this new plant dataset contains 175 plant proteins, of which 147 belong to one subcellular location, 27 belong to two locations, 1 belong to three locations and none to four or more locations. These plant proteins were added to

Table 9.24: Comparing HybridGO-Loc with state-of-the-art multi-label plant predictors based on independent tests using the new plant dataset. *SCL*: subcellular locations, including cell membrane (CM), cell wall (CW), chloroplast (CHL), cytoplasm (CYT), endoplasmic reticulum (ER), extracellular (EXT), Golgi apparatus (GOL), mitochondrion (MIT), nucleus (NUC), peroxisome (PER), plastid (PLA) and vacuole (VAC).

Label	SCL	Independent Test Locative Accuracy			
		Plant-mPLoc [96]	iLoc-Plant [87]	mGOASVM	HybridGO-Loc
1	CM	8/16 = 0.500	1/16 = 0.063	7/16 = 0.438	16/16 = 1.000
2	CW	0/1 = 0	0/1 = 0	0/1 = 0%	1/1 = 1.000
3	CHL	27/54 = 0.500	45/54 = 0.833	39/54 = 0.722	30/54 = 0.556
4	CYT	5/38 = 0.132	15/38 = 0.395	19/38 = 0.500	31/38 = 0.816
5	ER	1/9 = 0.111	1/9 = 0.111	3/9 = 0.333	4/9 = 0.444
6	EXT	0/3 = 0	0/3 = 0	1/3 = 0.333	0/3 = 0
7	GOL	3/7 = 0.429	1/7 = 0.143	3/7 = 0.429	7/7 = 1.000
8	MIT	6/16 = 0.375	3/16 = 0.188	11/16 = 0.688	16/16 = 1.000
9	NUC	31/46 = 0.674	43/46 = 0.935	33/46 = 0.717	44/46 = 0.957
10	PER	4/6 = 0.667	0/6 = 0	3/6 = 0.500	4/6 = 0.667
11	PLA	0/1 = 0	0/1 = 0	0/1 = 0	0/1 = 0
12	VAC	2/7 = 0.286	4/7 = 0.571	4/7 = 0.571	7/7 = 1.000
OLA		87/204 = 0.427	113/204 = 0.554	123/204 = 0.603	160/204 = **0.784**
OAA		60/175 = 0.343	91/175 = 0.520	97/175 = 0.554	127/175 = **0.726**
Accuracy		0.417	0.574	0.594	**0.784**
Precision		0.444	0.626	0.630	**0.826**
Recall		0.474	0.577	0.609	**0.798**
F1		0.444	0.592	0.611	**0.803**
HL		0.116	0.076	0.075	**0.037**

Swiss-Prot between 08-Mar-2011 and 18-Apr-2012. Because the plant dataset used for training the predictors was created on 29-Apr-2008, there is an almost 3-year time gap between the training data and test data in our experiments.

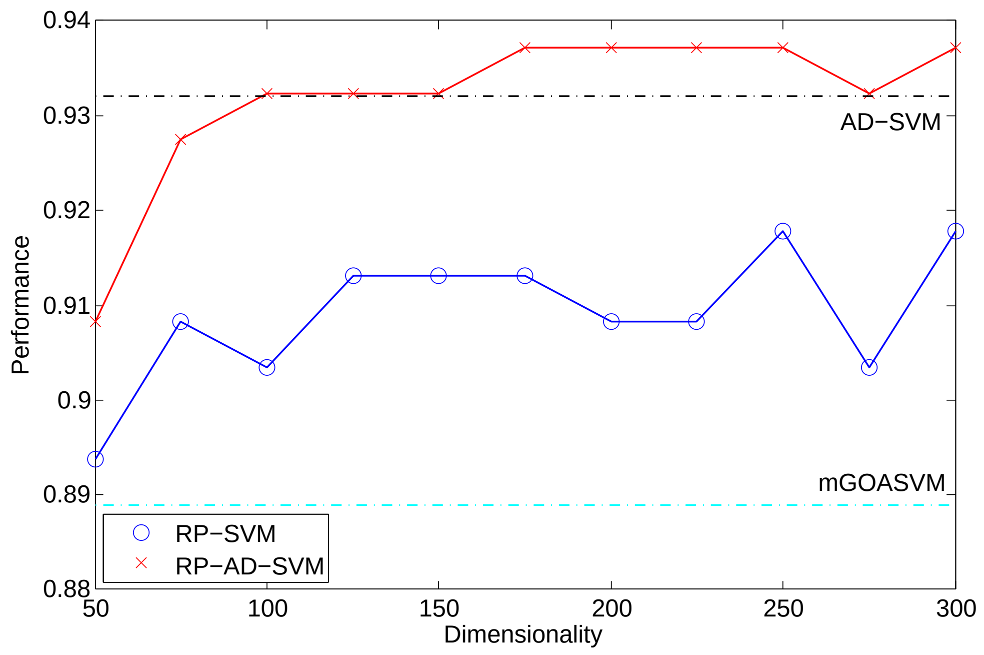
Table 9.24 compare the performance of HybridGO-Loc against several state-of-the-art multi-label plant predictors on the new plant dataset. All the predictors use the 978 proteins of the plant dataset (See Table 8.6) for training the classifier and make independent test on the new 175 proteins. As can be seen, HybridGO-Loc performs significantly better than all the other predictors in terms of all of the performance metrics. Similar conclusions can also be drawn from the performance in individual subcellular locations.

9.8 Performance of RP-SVM

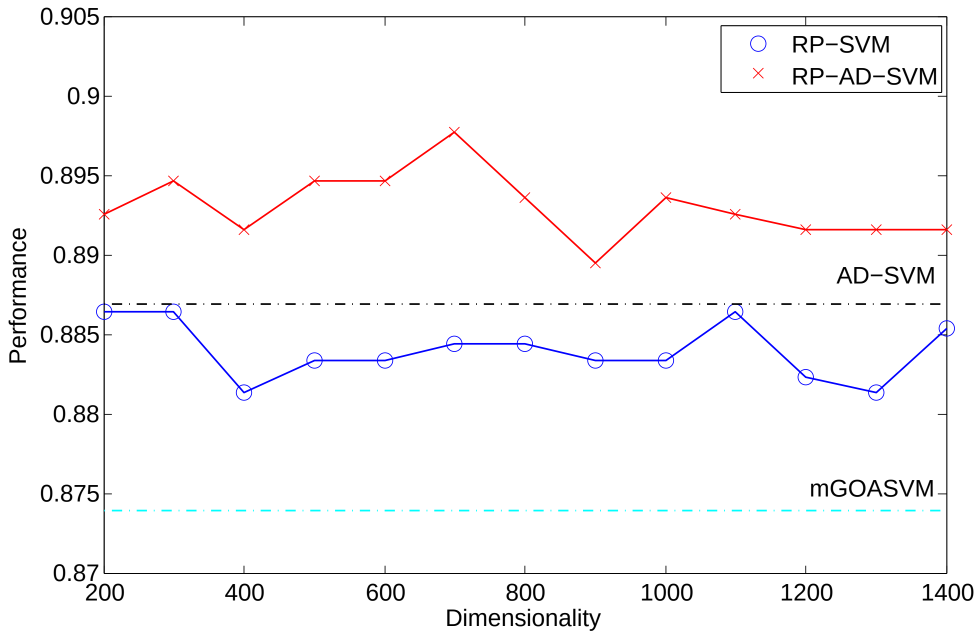
9.8.1 Performance of Ensemble Random Projection

Fig. 9.7(a) shows the performances of RP-SVM and RP-AD-SVM for different feature dimensions based on leave-one-out cross-validation on the virus dataset. The cyan dotted lines and black dotted lines represent the performance of mGOASVM [108] and AD-SVM [229], respectively. In other words, these two horizontal lines represent the original performance without dimension reduction for the two decision schemes. The dimensionality of the original feature vectors is 331. As can be seen, for dimensions between 50 and 300, the performance of RP-SVM is better than that of mGOASVM, which demonstrates that RP can boost the classification performance even the dimension is only one-sixth (50/331) of that of the original one. This suggests that the original feature vectors really have irrelevant or redundant information. Fig. 9.7(a) also shows that the performance of RP-AD-SVM is equivalent to, or better than that of AD-SVM when the dimensionality is larger than 100. This result demonstrates that random projection is complementary to the adaptive decision scheme. Similar conclusions can be drawn from the plant dataset shown in Fig. 9.7(b). Comparing Fig. 9.7(a) and Fig. 9.7(b) reveals that for the plant dataset, RP-AD-SVM outperforms AD-SVM for a wild range of feature dimensions (200 to 1541)[1], whereas for the virus dataset, the former outperforms the latter at a much narrower range (100 to 300). This suggests that RP-AD-SVM is more robust in classifying

[1]The dimensionality of the original feature vectors for the plant dataset is 1541.



(a) Performance on the virus dataset



(b) Performance on the plant dataset

Figure 9.7: Performance of RP-SVM and RP-AD-SVM at different feature dimensions based on leave-one-out cross-validation (LOOCV) on (a) the virus dataset and (b) the plant dataset, respectively. The cyan dotted lines and black dotted lines in both figures represent the performance of mGOASVM [108] and AD-SVM [229] on the two datasets, respectively.

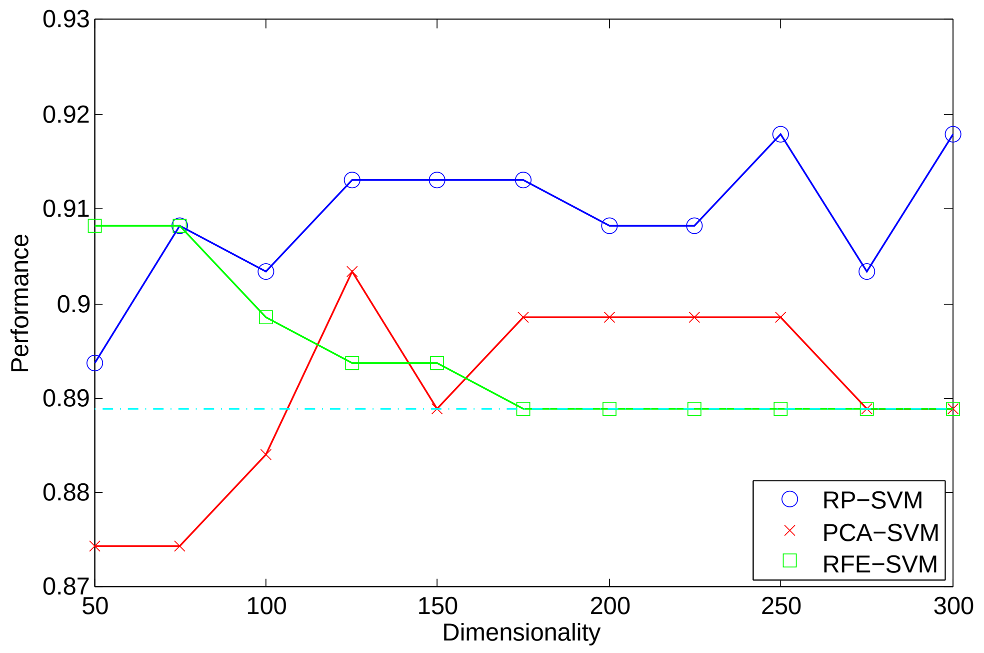
plant proteins than in classifying virus proteins.

9.8.2 Comparing with Other Dimension-Reduction Methods

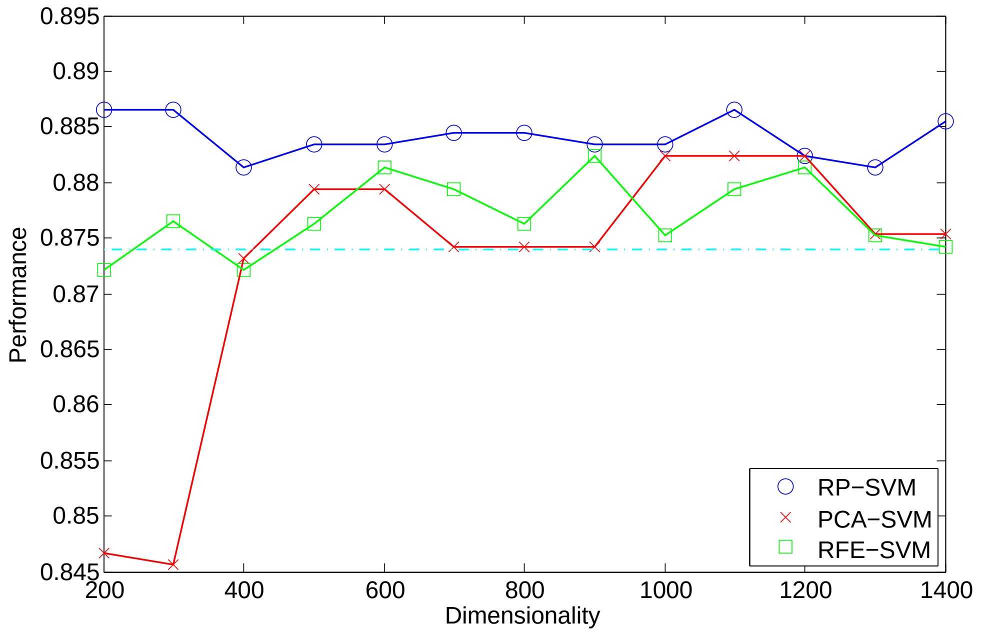
Fig. 9.8(a) and Fig. 9.8(b) compare RP-SVM with other dimension-reduction methods on the virus dataset and the plant dataset, respectively. Here, PCA-SVM and RFE-SVM mean replacing RP with principal component analysis (PCA) and recursive feature elimination (RFE) [230]. As can be seen, for the virus dataset, both RP-SVM and RFE-SVM perform better than mGOASVM when the dimensionality is larger than 50, while PCA-SVM performs better than mGOASVM only when the dimensionality is larger than 100. This suggests that the former two methods are more robust than PCA-SVM. When the dimension is higher than 75, RP-SVM outperforms both RFE-SVM and PCA-SVM, although RFE-SVM performs the best when the dimension is 50. For the plant dataset, only RP-SVM performs the best for a wide range of dimensionality, while RFE-SVM and PCA-SVM perform poorly when the dimension is reduced to 200 (out of 1541).

9.8.3 Performance of Single Random-Projection

Fig. 9.9(a) and Fig. 9.9(b) show the performance statistics of RP-SVM on the virus and the plant datasets, respectively, when the ensemble size (L in Eq. 7.5) is fixed to 1, which we refer to as 1-RP-SVM for simplicity. We created ten 1-RP-SVM classifiers, each with a different RP matrix. The *min OAA, max OAA, mean OAA* and *median OAA* represent the minimum, maximum, mean and median *OAA* of these 10 classifiers. As can be seen, for both datasets, even the *max OAA* is not always higher than that of mGOASVM, let alone the *minimum, mean or median OAA*. This demonstrates that a single RP cannot guarantee that the original performance can be kept when the dimension is reduced. On the contrary, combining the effect of several RPs, as evidenced by Fig. 9.7, can boost the



(a) Performance on the virus dataset



(b) Performance on the plant dataset

Figure 9.8: Comparing ensemble random projection with other dimension-reduction methods at different feature dimensions based on leave-one-out cross-validation (LOOCV) on (a) the virus dataset and (b) the plant dataset, respectively. The cyan dotted lines in both figures represent the performance of mGOASVM for the two datasets.

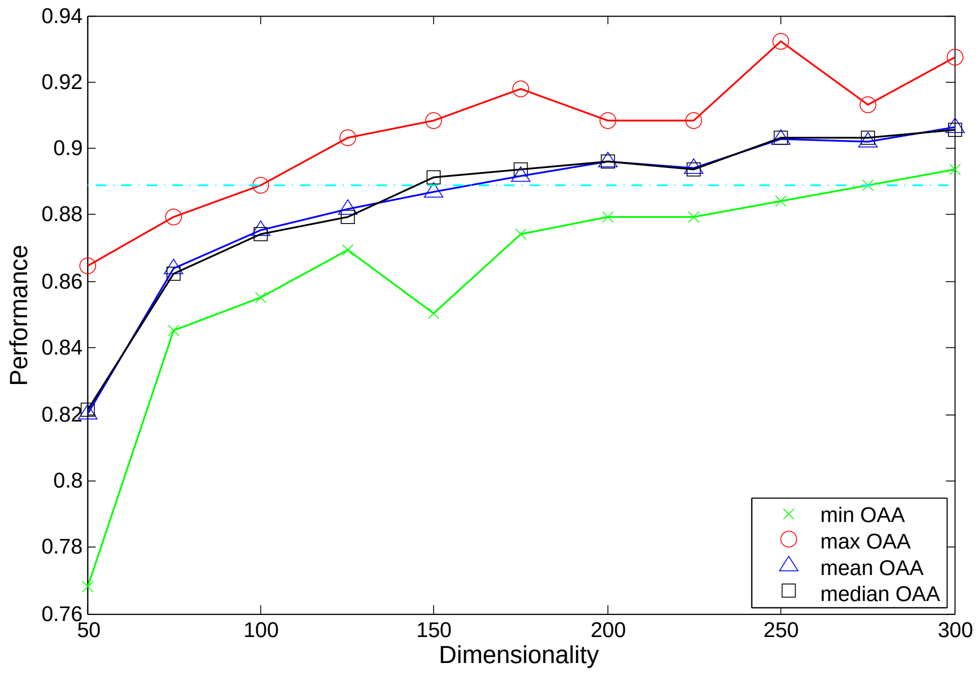
performance to a level higher than any of the individual RPs.

9.8.4 Effect of Dimensions and Ensemble Size

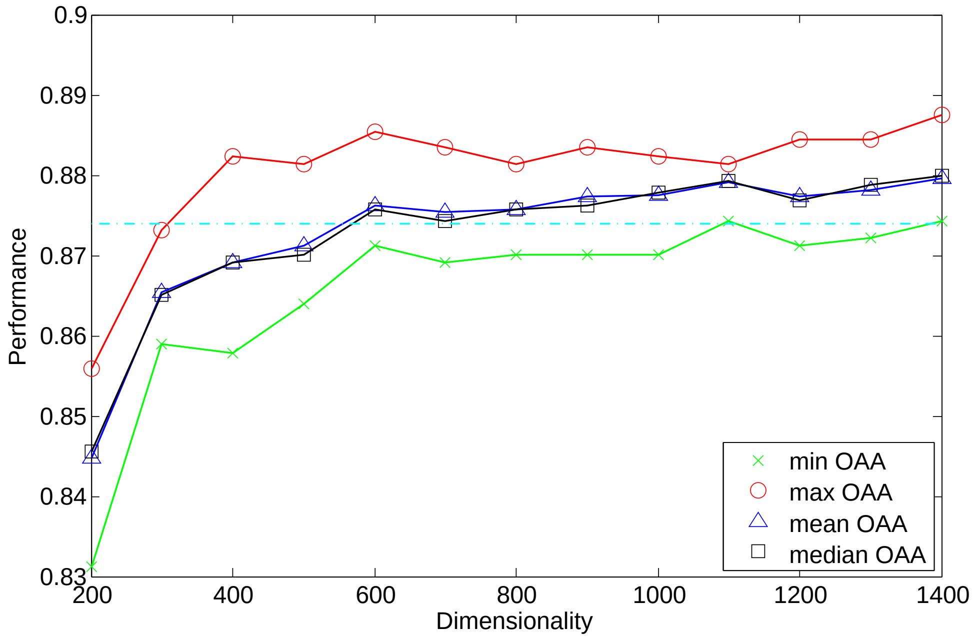
As individual RP cannot guarantee good performance, it is reasonable to ask: at least how many applications of RP can guarantee that the performance of the ensemble classifier is equivalent to, or even better than that of the one without RP (i.e., mGOASVM)? Fig. 9.10(a) and Fig. 9.10(b) show the performance of RP-SVM for different dimensions and different ensemble sizes of RPs on the virus and plant datasets, respectively. The blue/red areas represent the condition under which RP-SVM performs better/worse than mGOASVM. The yellow dotted planes in both figures represent the performance of mGOASVM on the two datasets. As can be seen, in the virus dataset, for dimensionality between 75 and 300, the performance of RP-SVM with at least 3 times of RP is better than that of mGOASVM; for dimensionality 50, we need at least 8 applications of RP to guarantee that the performance will not deteriorate. In the plant dataset, for dimensionality from 300 to 1400, RP-SVM with at least 4 applications of RP can outperform mGOASVM; for dimensionality 200, we need at least 5 applications of RP to obtain a performance better than mGOASVM. These results suggest that the proposed RP-SVM is very robust because only 3 or 4 applications of RP will be sufficient to achieve good performance.

9.8.5 Comparing RP-SVM with mGOASVM

Table 9.25 and Table 9.26 compare the performance of RP-SVM against mGOASVM on the virus and plant dataset. Both of the predictors use the information of GO terms as features. From the classification perspective, mGOASVM uses a multi-label SVM classifier; and the proposed RP-SVM uses ensemble RP to perform dimension reduction.

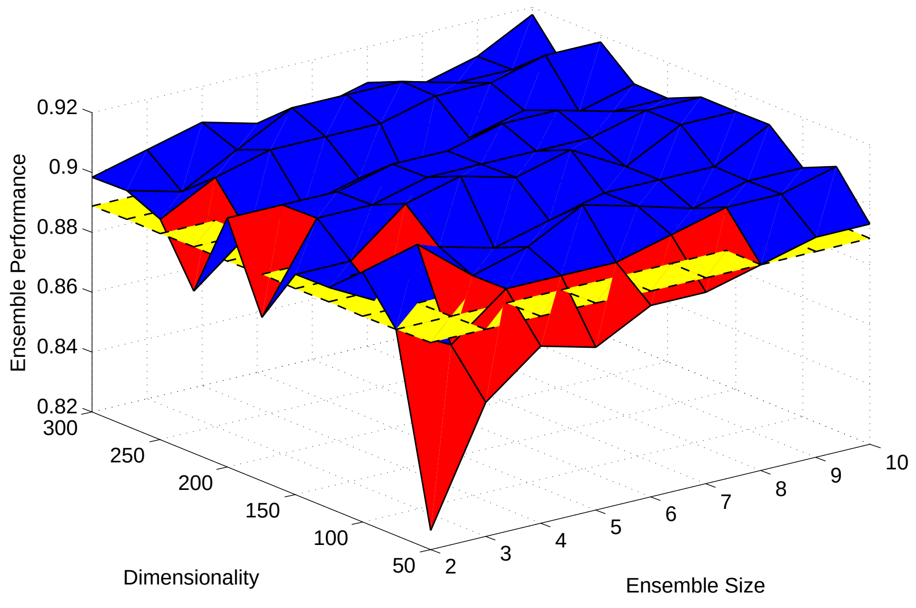


(a) Performance on the virus dataset

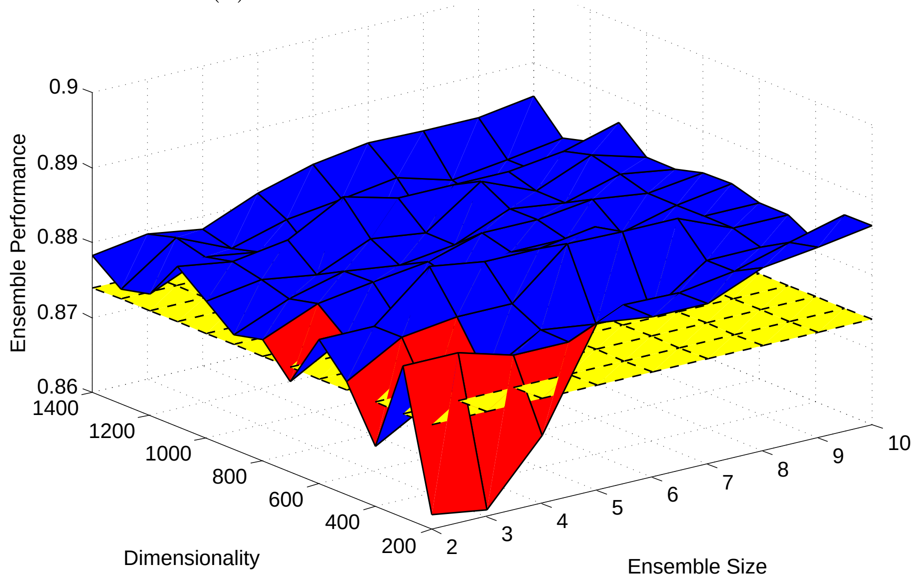


(b) Performance on the plant dataset

Figure 9.9: Performance of 1-RP-SVM at different feature dimensions based on leave-one-out cross-validation (LOOCV) on (a) the virus dataset and (b) the plant dataset, respectively. The cyan dotted lines in both figures represent the performance of mGOASVM on the two datasets. *1-RP-SVM*: RP-SVM with an ensemble size of 1.



(a) Performance on the virus dataset



(b) Performance on the plant dataset

Figure 9.10: Performance of RP-SVM at different feature dimensions and different ensemble sizes of random projection based on leave-one-out cross-validation (LOOCV) on (a) the virus dataset and (b) the plant dataset. The blue (red) areas represent the conditions for which RP-SVM outperforms (is inferior to) mGOASVM. The yellow dotted planes in both figures represent the performance of mGOASVM on the two datasets. *Ensemble Size*: Number of applications of random projections for constructing the ensemble classifier.

Table 9.25: Comparing RP-SVM with mGOASVM based on leave-one-out cross validation (LOOCV) using the virus dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM [108]	RP-SVM
1	Viral capsid	$8/8 = 1.000$	$8/8 = 1.000$
2	Host cell membrane	$32/33 = 0.970$	$31/33 = 0.939$
3	Host endoplasmic reticulum	$17/20 = 0.850$	$17/20 = 0.850$
4	Host cytoplasm	$85/87 = 0.977$	$86/87 = 0.989$
5	Host nucleus	$82/84 = 0.976$	$82/84 = 0.976$
6	Secreted	$20/20 = 1.000$	$20/20 = 1.000$
Overall Locative Accuracy (*OLA*)		$244/252 =$ **0.968**	$244/252 =$ **0.968**
Overall Actual Accuracy (*OAA*)		$184/207 = 0.889$	$190/207 =$ **0.918**
Accuracy		0.935	**0.950**
Precision		0.939	**0.957**
Recall		0.973	**0.976**
F1		0.950	**0.961**
HL		0.026	**0.020**

As shown in Table 9.25, the *OAA* of RP-SVM is more than 2% (absolute) higher than that of mGOASVM, although with the same *OLA*. In terms of *Accuracy, Precision, Recall, F1* and *HL*, RP-SVM perform better than mGOASVM. The results suggest that the proposed RP-SVM performs better than mGOASVM. The individual locative accuracies of RP-SVM are also comparable to mGOASVM.

Similar conclusions can be drawn for the plant dataset. As can be seen from Table 9.26, RP-SVM performs better than mGOASVM in terms of all metrics.

Comprehensive comparisons of all related multi-label predictors can be found in Section 9.10.

Table 9.26: Comparing RP-SVM with state-of-the-art multi-label predictors based on leave-one-out cross validation (LOOCV) using the plant dataset. "–" means the corresponding references do not provide the related metrics.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM	RP-SVM
1	Cell membrane	53/56 = 0.946	54/56 = 0.964
2	Cell wall	27/32 = 0.844	29/32 = 0.906
3	Chloroplast	272/286 = 0.951	284/286 = 0.993
4	Cytoplasm	174/182 = 0.956	172/182 = 0.945
5	Endoplasmic reticulum	38/42 = 0.905	39/42 = 0.929
6	Extracellular	22/22 = 1.000	21/22 = 0.955
7	Golgi apparatus	19/21 = 0.905	19/21 = 0.905
8	Mitochondrion	150/150 = 1.000	150/150 = 1.000
9	Nucleus	151/152 = 0.993	148/152 = 0.974
10	Peroxisome	21/21 = 1.000	21/21 = 1.000
11	Plastid	39/39 = 1.000	37/39 = 0.949
12	Vacuole	49/52 = 0.942	50/52 = 0.962
Overall Locative Accuracy (*OLA*)		1015/1055 = 0.962	1024/1055 = **0.971**
Overall Actual Accuracy (*OAA*)		855/978 = 0.874	867/978 = **0.887**
	Accuracy	0.926	**0.938**
	Precision	0.933	**0.946**
	Recall	0.968	**0.979**
	F1	0.942	**0.954**
	HL	0.013	**0.011**

9.9 Performance of R3P-Loc

9.9.1 Performance on the Compact Databases

Table 9.27 compares the performance of R3P-Loc on the proposed compact databases (ProSeq and ProSeq-GO) with that on the traditional databases (Swiss-Prot and GOA). The latter adopts the successive-search strategy used in [102] to enable R3P-Loc to be applicable to all the proteins based on the Swiss-Prot and GOA databases, while using ProSeq and ProSeq databases can avoid this time-consuming and laborious pro-

Table 9.27: Performance of R3P-Loc on the proposed compact databases based on leave-one-out cross validation (LOOCV) using the multi-label eukaryotic dataset. *ER*: endoplasmic reticulum; *OAA*: overall actual accuracy; *OLA*: overall locative accuracy; *F1*: F1-score; *HL*: Hamming loss.

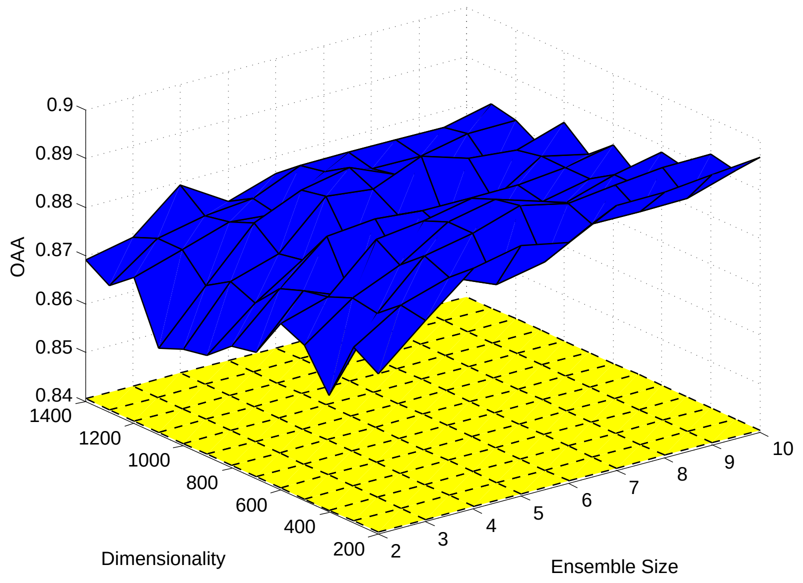
Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		Swiss-Prot + GOA	ProSeq + ProSeq-GO
1	Acrosome	2/14 = 0.143	2/14 = 0.143
2	Cell membrane	523/697 = 0.750	525/697 = 0.753
3	Cell wall	46/49 = 0.939	45/49 = 0.918
4	Centrosome	65/96 = 0.677	65/96 = 0.677
5	Chloroplast	375/385 = 0.974	375/385 = 0.974
6	Cyanelle	79/79 = 1.000	79/79 = 1.000
7	Cytoplasm	1964/2186 = 0.898	1960/2186 = 0.897
8	Cytoskeleton	50/139 = 0.360	53/139 = 0.381
9	ER	424/457 = 0.928	426/457 = 0.932
10	Endosome	12/41 = 0.293	12/41 = 0.293
11	Extracellular	968/1048 = 0.924	969/1048 = 0.925
12	Golgi apparatus	209/254 = 0.823	208/254 = 0.819
13	Hydrogenosome	10/10 = 1.000	10/10 = 1.000
14	Lysosome	47/57 = 0.825	47/57 = 0.825
15	Melanosome	9/47 = 0.192	10/47 = 0.213
16	Microsome	1/13 = 0.077	1/13 = 0.077
17	Mitochondrion	575/610 = 0.943	576/610 = 0.944
18	Nucleus	2169/2320 = 0.935	2157/2320 = 0.930
19	Peroxisome	103/110 = 0.936	104/110 = 0.946
20	Spindle pole body	47/68 = 0.691	42/68 = 0.618
21	Synapse	26/47 = 0.553	26/47 = 0.553
22	Vacuole	157/170 = 0.924	156/170 = 0.918
	OAA	6191/7766 = 0.797	6201/7766 = **0.799**
	OLA	7861/8897 = **0.884**	7848/8897 = 0.882
	Accuracy	**0.859**	**0.859**
	Precision	**0.882**	**0.882**
	Recall	**0.899**	0.898
	F1	**0.880**	**0.880**
	HL	**0.013**	**0.013**

cedure. The numbers of distinct GO terms found for the eukaryotic dataset by using ProSeq+ProSeq-GO and Swiss-Prot+GOA are 10775 and 10808, respectively.

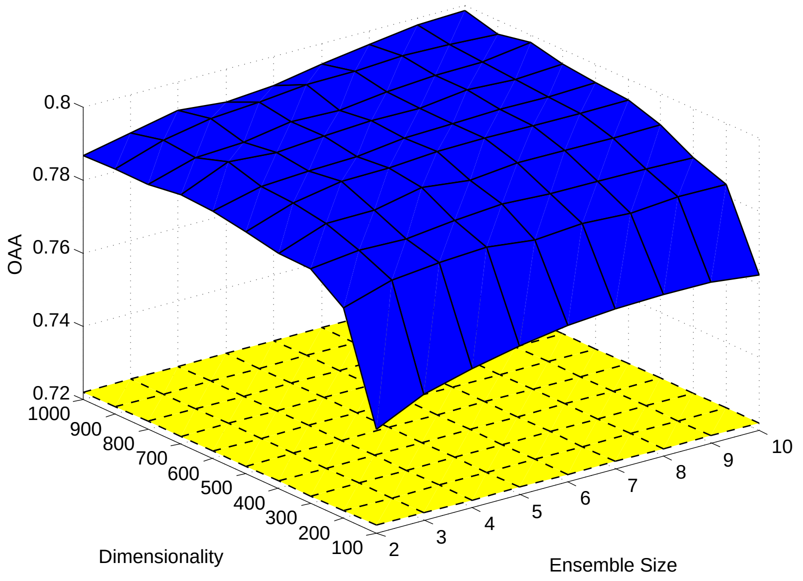
As can be seen, the performance using the combination of ProSeq and ProSeq-GO is almost equivalent to that using Swiss-Prot and GOA. The OAA of the former is even a bit better than that of the latter (0.799 vs 0.797), while the OLA and Recall is a bit worse. The rest measures and the locative accuracies in each location are almost the same. The experimental results suggest that despite of retrieving a bit fewer number of GO terms, using the proposed ProSeq and ProSeq-GO performs almost the same as that using Swiss-Prot and GOA combining with successive-search strategy, which demonstrates the effectiveness of the proposed compact databases.

9.9.2 Effect of Dimensions and Ensemble Size

Fig. 9.11 (a) shows the performance of R3P-Loc at different projected dimensions and ensemble sizes of random projection on the plant dataset. The dimensionality of the original feature vectors is 1541. The yellow dotted plane represents the performance using only multi-label ridge regression classifiers, namely the performance without random projection. For ease of comparison, we refer it to as RR-Loc. The mesh with blue (red) surfaces represent the projected dimensions and ensemble sizes at which the R3P-Loc performs better (poorer) than RR-Loc. As can be seen, there are no red areas across all dimensions (200 to 1200) and all ensemble sizes (2 to 10), which means the ensemble R3P-Loc always performs better than RR-Loc. The results suggest that using ensemble random projection can always boost the performance of RR-Loc. Similar conclusions can be drawn from Fig. 9.11 (b), which shows the performance of R3P-Loc at different projected dimensions and ensemble sizes of random projection on the eukaryotic dataset. The difference is that the original dimension of the feature vectors is 10,775, which means

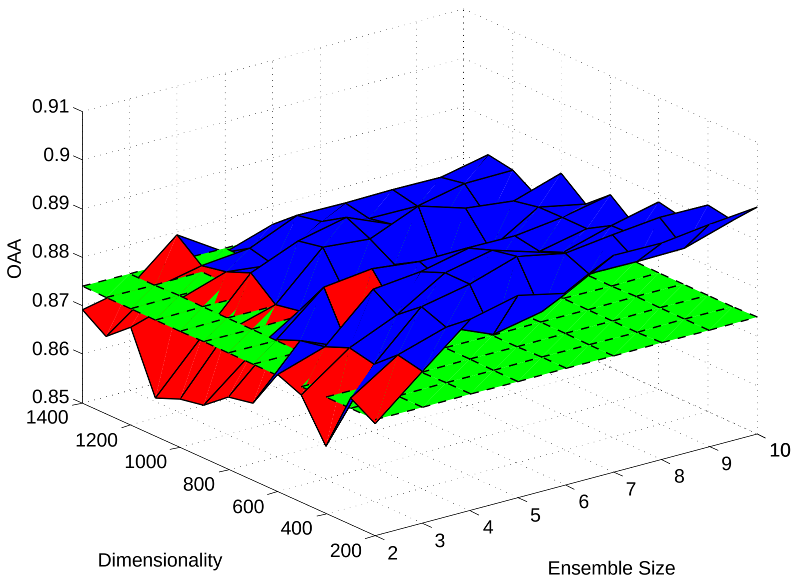


(a) Performance on the plant dataset

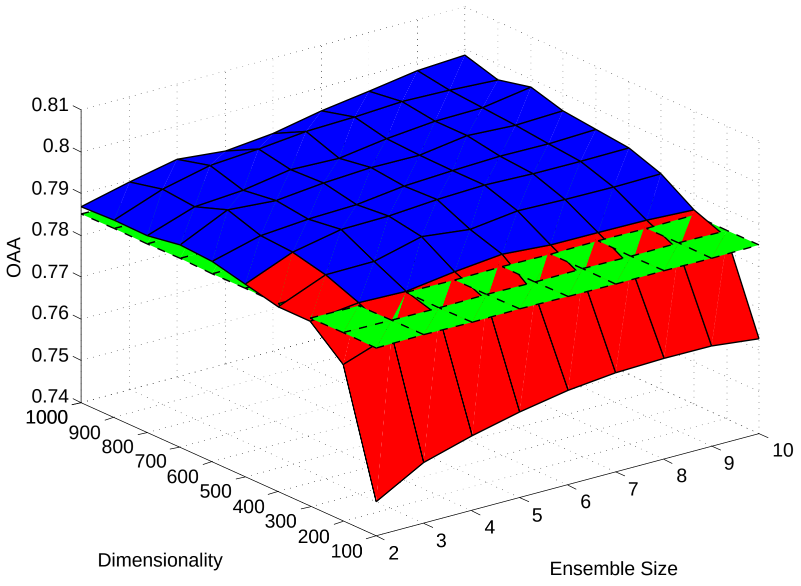


(b) Performance on the eukaryotic dataset

Figure 9.11: Performance of R3P-Loc at different projected dimensions and ensemble sizes of random projection on (a) the plant dataset and (b) the eukaryotic dataset, respectively. The yellow dotted plane represents the performance using only multi-label ridge regression classifiers (short for RR-Loc), namely the performance without random projection. The mesh with blue surfaces represent the projected dimensions and ensemble sizes at which the R3P-Loc performs better than RR-Loc. The original dimensions of the feature vectors for the plant and eukaryotic datasets are 1541 and 10775, respectively. *Ensemble Size*: Number of times of random projection for ensemble.



(a) Performance on the plant dataset



(b) Performance on the eukaryotic dataset

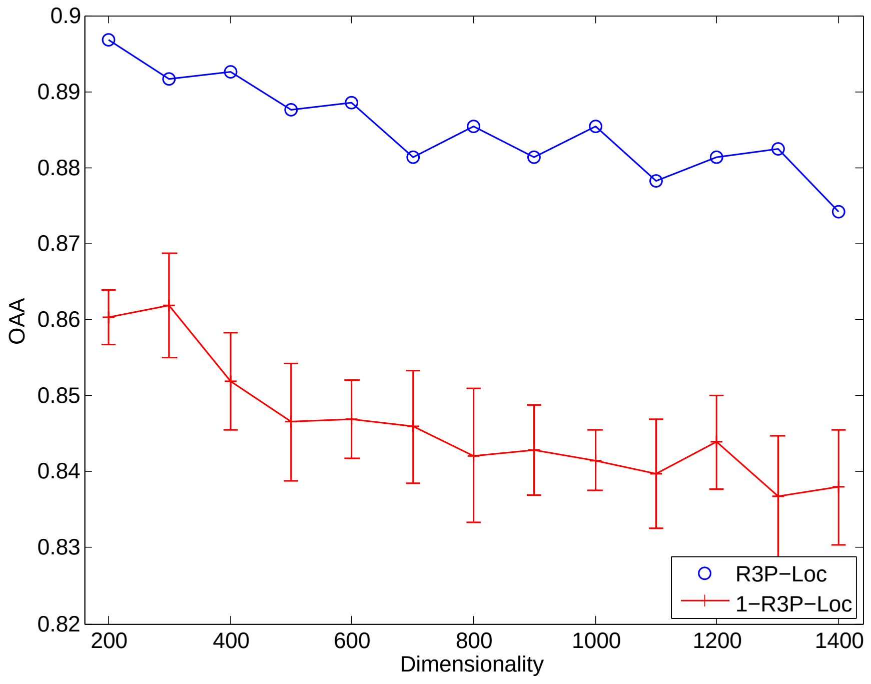
Figure 9.12: Comparing R3P-Loc with mGOASVM at different projected dimensions and ensemble sizes of random projection on (a) the plant dataset and (b) the eukaryotic dataset, respectively. The green dotted plane represents the accuracy of mGOASVM [108], which is a constant for all projected dimensions and ensemble size. The mesh with blue (red) surfaces represent the projected dimensions and ensemble sizes at which the ensemble R3P-Loc performs better (poorer) than mGOASVM. The original dimensions of the feature vectors for the plant and eukaryotic datasets are 1541 and 10775, respectively. *Ensemble Size*: Number of times of random projection for ensemble.

that R3P-Loc performs better than RR-Loc even when the feature dimension is reduced by almost 10 100 times.

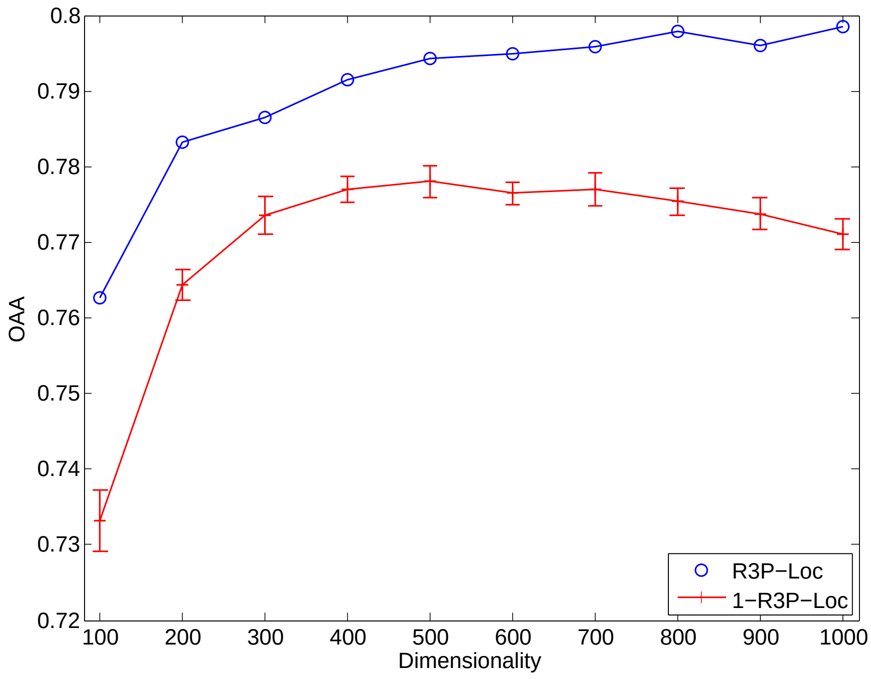
Fig. 9.12 (a) compares the performance of R3P-Loc with mGOASVM [108] at different projected dimensions and ensemble sizes of random projection on the plant dataset. The green dotted plane represents the accuracy of mGOASVM, which is a constant for all projected dimensions and ensemble size. The mesh with blue (red) surfaces represent the projected dimensions and ensemble sizes at which the ensemble R3P-Loc performs better (poorer) than mGOASVM. As can be seen, R3P-Loc performs better than mGOASVM throughout all dimensions (200 to 1400) when the ensemble size is more than 4. On the other hand, when the ensemble size is less than 2, the performance of R3P-Loc is worse than mGOASVM for almost all the dimensions. These results suggest that a large enough ensemble size of random projection is important for boosting the performance of R3P-Loc. Fig. 9.12 (b) compares the performance of R3P-Loc with mGOASVM on the eukaryotic dataset. As can be seen, R3P-Loc performs better than mGOASVM when the dimension is larger than 300 and the ensemble size is no less than 3 or the dimension is larger than 500 and the ensemble size is no less than 2. These experimental results suggest that a large enough projected dimension is also necessary for improving the performance of R3P-Loc.

9.9.3 Performance of Ensemble Random Projection

Fig. 9.13(a) shows the performance statistics of R3P-Loc based on LOOCV at different feature dimensions, when the ensemble size (L in Eq. 7.15) is fixed to 1, which we refer to as 1-R3P-Loc. We created ten 1-R3P-Loc classifiers, each with a different RP matrix. The result shows that even the highest accuracy of the ten 1-R3P-Loc is lower than that of R3P-Loc for all dimensions (200 to 1400). This suggests that the ensemble random



(a) Performance on the plant dataset



(b) Performance on the eukaryotic dataset

Figure 9.13: Performance of R3P-Loc at different feature dimensions on (a) the plant dataset and (b) the eukaryotic dataset, respectively. The original dimensions of the feature vectors for the plant and eukaryotic datasets are 1541 and 10775, respectively. *1-R3P-Loc*: RP-Loc with an ensemble size of 1.

Table 9.28: Comparing R3P-Loc with mGOASVM using the plant dataset.

Label	Subcellular Location	LOOCV Locative Accuracy (LA)	
		mGOASVM	R3P-Loc
1	Cell membrane	53/56 = 0.946	5/56 = 0.893
2	Cell wall	27/32 = 0.844	28/32 = 0.875
3	Chloroplast	272/286 = 0.951	279/286 = 0.976
4	Cytoplasm	174/182 = 0.956	172/182 = 0.945
5	Endoplasmic reticulum	38/42 = 0.905	36/42 = 0.857
6	Extracellular	22/22 = 1.000	17/22 = 0.773
7	Golgi apparatus	19/21 = 0.905	19/21 = 0.905
8	Mitochondrion	150/150 = 1.000	142/150 = 0.947
9	Nucleus	151/152 = 0.993	147/152 = 0.967
10	Peroxisome	21/21 = 1.000	21/21 = 1.000
11	Plastid	39/39 = 1.000	36/39 = 0.923
12	Vacuole	49/52 = 0.942	48/52 = 0.923
Overall Actual Accuracy (*OAA*)		855/978 = 0.874	877/978 = **0.897**
Overall Locative Accuracy (*OLA*)		1015/1055 = **0.962**	995/1055 = 0.943
	Accuracy	0.926	**0.934**
	Precision	0.933	**0.950**
	Recall	**0.968**	0.956
	F1	0.942	**0.947**
	HL	0.013	**0.011**

projection can significantly boost the performance of R3P-Loc. Similar conclusions can be also drawn from Fig. 9.13(b), which shows the performance statistics of R3P-Loc on the eukaryotic dataset.

9.9.4 Comparing with State-of-the-Art Predictors

Table 9.28 and Table 9.29 compare the performance of R3P-Loc against several state-of-the-art multi-label predictors on the plant and eukaryotic dataset. All of the predictors use the information of GO terms as features. From the classification perspective, Euk-mPLoc 2.0 [158] uses an ensemble OET-KNN (optimized evidence-theoretic K-nearest

Table 9.29: Comparing R3P-Loc with state-of-the-art multi-label predictors using the multi-label eukaryotic dataset. *SCL*: subcellular locations, including acrosome (ACR), cell membrane (CM), cell wall (CW), centrosome (CEN), chloroplast (CHL), cyanelle (CYA), cytoplasm (CYT), cytoskeleton (CYK), endoplasmic reticulum (ER), endosome (END), extracellular (EXT), Golgi apparatus (GOL), hydrogenosome (HYD), lysosome (LYS), melanosome (MEL), microsome (MIC), mitochondrion (MIT), nucleus (NUC), peroxisome (PER), spindle pole body (SPI), synapse (SYN) and vacuole (VAC). "–" means the corresponding references do not provide the related metrics.

Label	SCL	LOOCV Locative Accuracy (LA)			
		Euk-mPLoc 2.0 [158]	iLoc-Euk [90]	mGOASVM	R3P-Loc
1	ACR	1/14 = 0.071	1/14 = 0.071	12/14 = 0.857	2/14 = 0.143
2	CM	452/697 = 0.649	561/697 = 0.805	643/697 = 0.923	525/697 = 0.753
3	CW	6/49 = 0.122	8/49 = 0.163	46/49 = 0.939	45/49 = 0.918
4	CEN	22/96 = 0.229	67/96 = 0.698	87/96 = 0.906	65/96 = 0.677
5	CHL	318/385 = 0.826	338/385 = 0.878	375/385 = 0.974	375/385 = 0.974
6	CYA	47/79 = 0.595	51/79 = 0.646	79/79 = 1.000	79/79 = 1.000
7	CYT	1418/2186 = 0.649	1677/2186 = 0.767	2020/2186 = 0.924	1960/2186 = 0.897
8	CYK	44/139 = 0.317	38/139 = 0.273	100/139 = 0.719	53/139 = 0.381
9	ER	348/457 = 0.762	407/457 = 0.891	441/457 = 0.965	426/457 = 0.932
10	END	2/41 = 0.049	3/41 = 0.073	28/41 = 0.683	12/41 = 0.293
11	EXT	858/1048 = 0.819	948/1048 = 0.905	1016/1048 = 0.970	969/1048 = 0.925
12	GOL	56/254 = 0.221	161/254 = 0.634	231/254 = 0.909	208/254 = 0.819
13	HYD	2/10 = 0.200	0/10 = 0.000	10/10 = 1.000	10/10 = 1.000
14	LYS	26/57 = 0.456	18/57 = 0.316	52/57 = 0.912	47/57 = 0.825
15	MEL	0/47 = 0.000	1/47 = 0.021	44/47 = 0.936	10/47 = 0.213
16	MIC	1/13 = 0.077	0/13 = 0.000	7/13 = 0.539	1/13 = 0.077
17	MIT	427/610 = 0.700	470/610 = 0.771	594/610 = 0.974	576/610 = 0.944
18	NUC	1501/2320 = 0.647	2040/2320 = 0.879	2194/2320 = 0.946	2157/2320 = 0.930
19	PER	56/110 = 0.509	60/110 = 0.546	108/110 = 0.982	104/110 = 0.946
20	SPI	23/68 = 0.338	45/68 = 0.662	65/68 = 0.956	42/68 = 0.618
21	SYN	0/47 = 0.000	18/47 = 0.383	40/47 = 0.851	26/47 = 0.553
22	VAC	101/170 = 0.594	122/170 = 0.718	166/170 = 0.977	156/170 = 0.918
OAA		–	5535/7766 = 0.713	6097/7766 = 0.785	6201/7766 = **0.799**
OLA		5709/8897 = 0.642	7034/8897 = 0.791	8358/8897 = **0.939**	7848/8897 = 0.882
Accuracy		–	–	0.849	**0.859**
Precision		–	–	0.878	**0.882**
Recall		–	–	**0.946**	0.898
F1		–	–	0.878	**0.880**
HL		–	–	0.014	**0.013**

neighbors) classifier; iLoc-Euk [90] uses a multi-label KNN classifier; mGOASVM [108] uses a multi-label SVM classifier[2]; and the proposed R3P-Loc uses ensemble RP and ridge regression classifiers. Here, for the plant dataset, only the comparison with mGOASVM is provided; more comprehensive comparisons of all related state-of-the-art multi-label predictors can be found in Section 9.10.

As shown in Table 9.28, the *OAA* of R3P-Loc is more than 2% (absolute) higher than that of mGOASVM, although a bit less than mGOASVM on the *OLA* and *Recall*. In terms of *Accuracy, Precision, F1* and *HL*, R3P-Loc perform better than mGOASVM. The results suggest that the proposed R3P-Loc performs better than the state-of-the-art classifiers. The individual locative accuracies of R3P-Loc are also comparable to mGOASVM.

Similar conclusions can be drawn from Table 9.29, which compares R3P-Loc with state-of-the-art predictors on the eukaryotic dataset. R3P-Loc performs significantly better than Euk-mPLoc 2.0 and iLoc-Euk in terms of all the measures. And R3P-Loc performs better than mGOASVM in terms of *OAA Accuracy, Precision, F1* and *HL*, while a bit worse on *OLA* and *Recall*. This is probably because the ensemble random projection makes R3P-Loc perform more stringently to control over-predictions than mGOASVM.

9.10 Comprehensive Comparison of Proposed Predictors

Table 9.30 and Table 9.31 show the comprehensive comparisons of the performance of all of the proposed multi-label predictors against state-of-the-art predictors on the virus and plant datasets based on leave-one-out cross validation. All of the predictors except HybridGO-Loc use the GO frequency features while HybridGO-Loc extracts the feature information not only from GO frequency features but also from GO semantic similarity

[2]We performed mGOASVM on the eukaryotic dataset.

Table 9.30: Comparing all proposed multi-label predictors with state-of-the-art predictors using the virus dataset. *OAA*: overall actual accuracy; *OLA*: overall locative accuracy. See Eqs. 8.9–8.15 for the definitions of the performance measures. "–" means the corresponding references do not provide the results on the respective metrics.

Predictors	OAA	OLA	Accuracy	Precision	Recall	F1	HL
Virus-mPLoc [154]	–	0.603	–	–	–	–	–
KNN-SVM [156]	–	0.807	–	–	–	–	–
iLoc-Virus [93]	0.748	0.782	–	–	–	–	–
mGOASVM	0.889	0.968	0.935	0.939	0.973	0.950	0.026
AD-SVM	0.932	0.960	0.953	0.960	0.966	0.960	0.019
mPLR-Loc	0.903	0.948	0.942	0.957	0.965	0.955	0.023
HybridGO-Loc	**0.937**	**0.972**	**0.961**	**0.965**	**0.976**	**0.968**	**0.016**
RP-SVM	0.918	0.968	0.950	0.957	**0.976**	0.961	0.020

features.. Virus-mPLoc [154] and Plant-mPLoc [96] use an ensemble OET-KNN (optimized evidence-theoretic K-nearest neighbors) classifier; iLoc-Virus [93] and iLoc-Plant [87] use a multi-label KNN classifier; KNN-SVM [156] uses an ensemble of classifiers combining KNN and SVM; mGOASVM [108] uses a multi-label SVM classifier; AD-SVM and mPLR-Loc use multi-label SVM and penalized logistic regression classifiers, respectively, both equipped with an adaptive decision scheme; RP-SVM and R3P-Loc use ensemble random projection to multi-label SVM and ridge regression classifiers, respectively; and HybridGO-Loc uses a multi-label SVM classifier incorporated with an adaptive decision scheme.

As can be seen from Table 9.30, All of the proposed predictors perform significantly better than Virus-mPLoc, KNN-SVM and iLoc-Virus in terms of the available performance metrics. Among the proposed predictors, HybridGO-Loc performs the best, which demonstrates that mining deeper GO information (i.e. semantic similarity information) is important to boost the performance of predictors. Based on mGOASVM, the other proposed predictors have made different improvement either from refinement of multi-label

Table 9.31: Comparing all proposed multi-label predictors with state-of-the-art predictors using the plant dataset. *OAA*: overall actual accuracy; *OLA*: overall locative accuracy. See Eqs. 8.9–8.15 for the definitions of the performance measures. "–" means the corresponding references do not provide the results on the respective metrics.

Predictors	OAA	OLA	Accuracy	Precision	Recall	F1	HL
Plant-mPLoc [96]	–	0.637	–	–	–	–	–
iLoc-Plant [87]	0.681	0.717	–	–	–	–	–
mGOASVM	0.874	0.962	0.926	0.933	0.968	0.942	0.013
AD-SVM	0.887	0.946	0.928	0.941	0.956	0.942	0.013
mPLR-Loc	0.908	0.937	0.939	0.956	0.952	0.949	0.010
HybridGO-Loc	**0.936**	0.956	**0.959**	**0.972**	0.968	**0.966**	**0.007**
RP-SVM	0.887	**0.971**	0.938	0.946	**0.979**	0.954	0.011
R3P-Loc	0.897	0.943	0.934	0.950	0.956	0.947	0.011

classifiers, dimensionality reduction or mining deeper into the GO database for feature extraction, of which deeper feature extraction contributes most to the performance gain.

Similar conclusions can be drawn from Table 9.31 except the differences that RP-SVM is superior to HybridGO-loc in terms of *OLA* and *Recall* while HybridGO-Loc performs the best in terms of the other metrics. This is probably because the adaptive decision scheme for HybridGO-Loc makes a compromise between higher *OAA* and a bit lower *OLA* and *Recall*. The analysis of the adaptive decision scheme can be found in Section 9.5.1.

9.11 Summary

This chapter elaborated experimental results for all the proposed predictors, including GOASVM and FusionSVM for single-location protein subcellular localization, mGOASVM, AD-SVM, mPLR-Loc, SS-Loc, HybridGO-Loc, RP-SVM and R3P-Loc for multi-label protein subcellular localization. In-depth analysis and detailed properties of all the predictors are specified. Comprehensive comparisons of all the proposed predictors are also provided.

Chapter 10: Discussion

Discussion chapters analyse the findings of the thesis. They should include a discussion of how the findings relate to the studies discussed in the literature review, identify any limitations of the study, and the implications the results have for researchers and the wider community.

This chapter is organised in the following way and is very effective partly because the writer includes the following:

Structure

(Introduction)	Not included
Limitations	Section 10.1
Analysis of Model 1	Section 10.2-3
Analysis of Model 2	Section 10.4
Analysis of Model 3	Section 10.5
Comparison of Models	Section 10.6
Summary	Section 10.7

Content

- Introduces the chapter with a short summary paragraph
- States aim of the chapter (e.g. Chapter 10, sentence 1)
- Outlines content of the chapter (e.g. Chapter 10, sentence 2-4)
- Discusses limitations of methodology (e.g. Section 10.1)
- Refers to studies that appeared in the Literature Review (e.g. Section 10.3.2, paragraph 1, sentence 2)
- Compares findings with findings of other studies (e.g. Section 10.3.2, paragraph 1, sentence 2)
- Develops sections in a logical way, e.g. Section 10.4.3:

Paragraph 1	Aim (sentence 1)
	Limitations (sentence 2)
	Explains limitations (sentence 3)
Paragraph 2	Possible Solutions
Paragraph 3	Discusses possible drawbacks with proposed solution (sentence 1-3)

Explains why the drawbacks are not applicable to this case (sentence 4).

- Outlines specific aspects of the topic that need further investigation following the results obtained from present study (e.g. Section 10.5.2, paragraph 2).

Language

- Generalises, e.g. *it is generally accepted that* (e.g. Section 10.2.1, paragraph 2, sentence 6)
- Uses tentative language to discuss possible reasons, e.g. are likely to, would probably (e.g. Section 10.6, paragraph 2, sentence 2 and sentence 7)

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

💡Highlight the uniqueness of the research

💡Use positive language to explain why the results are important.

💡Explain how the results fill a gap in knowledge.

💡Explain how the result will contribute the academic field.

💡Avoid vague statements e.g. *some people may wonder* (e.g. Section 10.2.2, sentence 2). It is better use passive voice and a more accurate verb e.g. *It has been suggested that…It may be thought that*

💡Avoid spoken language, e.g. *the answer is 'no'* (e.g. Section 10.2.2, sentence 4). It is better to use more formal language e.g. *This is not the case.*

💡Avoid using rhetorical questions for sub-headings (e.g. Section 10.2.2) and in the text of the document (e.g. Section 10.2.2, sentence 3).

Chapter 10

Discussions

This chapter will further discuss the advantages and disadvantages of the proposed predictors, especially for those multi-label predictors. First, the impact of noise data in the GOA database on the performance will be discussed. Then, based on mGOASVM, our proposed predictors make improvement in (1) multi-label classification (AD-SVM and mPLR-Loc), (2) feature extraction (SS-Loc and HybridGO-Loc) and (3) finding relevant subspaces (RP-SVM and R3P-Loc). Discussions from these perspectives will be presented. Next, the overall comparisons among all of the predictors will be discussed.

10.1 Noise Data in the GOA database

As stated in Section 4.1.1, the GOA database is constructed by various biological research communities around the world.[1] It is possible that some annotations for the same proteins are done by different GO consortium contributing groups around the world. In this case, it is likely that the annotations of the same biological process, molecular function or cellular component for the same protein by different research groups are different, or even contradictory, which may result in the inaccuracy or inconsistency of the GO annotations.

[1]http://geneontology.org/page/go-consortium-contributors-list

In other words, there are inevitably some noisy data or outliers in the GOA database. These noisy data and outliers may negatively affect the performance of machine-learning based approaches.

For this concern, first of all, we need to admit that these noisy data and outliers are likely to exist in the GOA database, and unfortunately we cannot easily distinguish them from correct GO annotations. Only wet-lab experimentalists can rely on their biological knowledge to discriminate these noisy data or outliers and remove them from the database. However, we can remain optimistic about our proposed predictors for the following reasons:

1. The GOA database has some guidelines to guarantee high-quality data in the GOA database. First, GO annotations are classified as electronic annotation, literature annotation and sequence-based annotation. Each annotation entry will be labeled by an evidence code to represent its sources. For example, the evidence code 'IEA' means the GO annotation is inferred from electronic annotation, and 'EXP' is inferred from experiments. This information may be conducive for users to distinguish different kinds of annotations.[2]

2. In this thesis, term-frequency information was used to emphasize those annotations that are confirmed by different research groups. From our observations, the same GO term for the same protein may appear more than once in the GOA database, but possibly with different evidence codes, or from different contributing databases. This means that this kind of GO terms are validated several times by different research groups and by different ways, which lead to the same annotation results.

[2]Note that in our proposed methods, we do not incorporate this information. This is because we have done some experiments (results not shown) by using this information in our feature vectors, but the performance remain almost the same.

On the contrary, if different research groups annotate the same protein by different GO terms whose annotations are contradictory with each other, the frequencies of these GO terms for this protein should be low. In other words, the higher the frequency a GO term appears, the more times this GO annotation is confirmed by different research groups, and the more credible the annotation of this GO term. By using the term-frequency in our feature vectors, we can enhance the influence of those GO terms which appear more frequently; or in other words, we can enhance the influence of those GO terms whose annotations are consistent with each other. Meanwhile, we can indirectly suppress the influence of those GO terms which appear less frequently; or in other words, we can suppress the influence of those GO terms whose annotations are contradictory with each other.

3. The noisy data and outliers may exist in both training and testing datasets, in which case the negative impact of the noisy data and outliers may be reduced. In our methods, we used homologous transfer methods to obtain the feature information for both training and testing proteins. Thus, if there are some noisy data and outliers in the GOA database, it is possible that both the training and testing proteins contain the same noise data and outliers. In such case, we conjecture that the noise data and outliers may contribute to the final decision, and more interestingly may improve the prediction performance instead of making the performance poor.

10.2 Analysis of Single-label Predictors

10.2.1 GOASVM versus FusionSVM

For single-location protein subcellular localization, this thesis proposed two GO-based predictors, namely GOASVM and FusionSVM. Although both predictors use GO information and advocate using the term-frequency method to replace the 1-0 value method for constructing GO vectors, the differences between these two predictors are definite, which are presented in the following perspectives:

1. **Retrieval of GO terms**. GOASVM exploits the GO terms from the GOA database, while FusionSVM extracts the GO terms from a program called InterProScan.
2. **Searching GO terms**. To guarantee that GOASVM is applicable to all of the proteins of interest, GOASVM adopts a successive-search strategy to incorporate remote yet useful homologous GO information for classifiation. On the other hand, FusionSVM only uses InterProScan to generate GO terms, which cannot make sure that it is applicable to all of the proteins. Thus, FusionSVM needs to use other features as a back-up.
3. **Feature information**. GOASVM only uses GO information as the features while FusionSVM uses both GO information and profile alignment information, whose scores are combined to make final decisions.

Besides the differences mentioned above, some other points are also worthy of note. First, the numbers of subcellular locations for the datasets used in these two predictors are different. The number of subcellular locations in the eukaryotic datasets (i.e. EU16 and the novel eukaryotic dataset) for GOASVM is 16 while that for FusionSVM is only 11. Moreover, the degree of sequence similarity in the datasets is different. The sequence similarity in the datasets for evaluating GOASVM – including EU16, HUM12 and the novel eukaryotic dataset – is cut off at 25%; on the other hand, the sequence similarity of the eukaryotic dataset for FusionSVM is only cut off at 50%. It is generally accepted that, the more similar the proteins of the dataset of interest is, the easier the predictors can get a high accuracy. Therefore, for the same predictor, the lower the sequence-similarity cut-off percentage, the lower the achievable accuracy. Nevertheless, even under the condition that the datasets for GOASVM are more stringent than that used for FusionSVM, GOASVM can still achieve a much better performance than FusionSVM (direct comparison results are not shown, but we can easily draw this conclusion from Tables 9.3 and 9.8).

10.2.2 Can GOASVM be combined with PairProSVM?

From the experimental results in Chapter 9 and the analysis above, we can see that GOASVM performs better than FusionSVM. However, some people may wonder: If the fusion of the GO-based InterProScan and the profile-based PairProSVM can boost the prediction performance, is it possible to combine the GO-based GOASVM with PairProSVM to further boost the performance? The answer is 'No'. This is because typically score-fusion methods can boost the performance only if the performance of the two methods of interest are more or less comparable [231, 232], as evident in Table 9.7, where InterProSVM and PairProSVM can achieve comparable accuracies (72.21% and 77.05%). On the contrary, GOASVM remarkably outperforms PairProSVM, as evident in Table 9.3, where the accuracies of GOASVM and PairProSVM are 94.55% and 54.52%, respectively. It is highly likely that fusing GOASVM and PairProSVM will perform worse than GOASVM. Therefore, it is unwise to fuse the scores of GOASVM and PairProSVM to make predictions.

10.3 Advantages of mGOASVM

mGOASVM possesses several desirable properties that make it outperform Virus-mPLoc [154], iLoc-Virus [93], Plant-mPLoc [96] and iLoc-Plant [87], which are specified subsequently.

10.3.1 GO-Vector Construction

Virus-mPLoc and Plant-mPLoc construct GO vectors by using 1-0 value to indicate the presence and absence of some predefined GO terms. This method is simple and logically plausible, but some information will be inevitably lost because it quantizes the frequency of occurrences of GO terms to either 1 or 0. The GO vectors in iLoc-Virus and iLoc-Plant contain more information than those in Virus-mPLoc and Plant-mPLoc, because the former two consider not only the GO terms of the query protein but also the GO terms of its homologs. Specifically, instead of using 1-0 value, each element of the GO vectors in the iLoc-Virus and iLoc-Plant represents the percentage of homologous proteins containing the corresponding GO term. However, the method ignores the fact that a GO term may be used to annotate the same protein multiple times under different entries in the GO annotation database. On the contrary, mGOASVM uses the frequency of occurrences of GO terms to construct the GO vectors. Intuitively, this is because proteins of the same subcellular localization tend to be annotated by similar sets of GO terms. The advantages of using the GO term-frequency count as features is evident by the superior results in Table 9.11.

10.3.2 GO Subspace Selection

To facilitate the sophisticated machine learning approach for the multi-label problem, GO subspace selection is adopted. Unlike the traditional methods [154, 96, 87, 93] which use all of the GO terms in the GO annotation database to form the GO-vector space, mGOASVM selects a relevant 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 March 2011, the number of GO terms is 18656, which means that without feature selection, the GO vectors will have dimension 18656. This imposes computational burden on the classifier, especially when leave-one-out cross validation is used for evaluation. There is no doubt that many of the GO terms in the full space are redundant, irrelevant or even detrimental to prediction performance. By selecting a set of distinct GO terms to form a GO subspace, mGOASVM can reduce the irrelevant information and at the same time retain useful information. As can be seen from Table 9.14, for the virus dataset, around 300 to 400 distinct GO terms are sufficient for good performance. Therefore, using GO subspace selection can tremendously speed up the prediction without compromising the performance.

10.3.3 Capability of Handling Multi-Label Problems

An efficient way to handle multi-label problems is to predict the number of labels for each sample first, and then to predict the specific label set for each sample according to the order of the scores. Let us compare mGOASVM with two kinds of existing approaches.

- When predicting the number of subcellular locations for a query protein, iLoc-Virus and iLoc-Plant determine the number of labels of a query protein based on the number of labels of its nearest training sample. mGOASVM, on the contrary, determines the number of labels for a query protein by looking at the number of positive-class decisions among all of the one-vs-rest SVM classifiers. Therefore, the number of labels depends on the whole training set as opposed to the query protein's nearest neighbor in the training set.

- As opposed to Virus-mPLoc and Plant-mPLoc which require a pre-defined threshold, our mGOASVM adopts a machine learning approach to solving the multi-label classification problem. The predicted class labels in mGOASVM are assigned based on the SVMs that produce positive responses to the query protein.

In summary, the superiority of mGOASVM in handling multi-label problems is evident in Table 9.9.

From the machine learning perspective, prediction of multi-location proteins is a multi-label learning problem. Approaches to addressing this problem can be divided into types: problem transformation and algorithm adaptation [145]. The multi-label KNN classifiers used in iLoc-Plant and iLoc-Virus belong to the first type whereas our multi-label SVM classifier belongs to the second type. While our results show that multi-label SVMs perform better than multi-label KNN, further work needs to be done to compare these two types of approaches in the context of multi-label subcellular localization.

10.4 Analysis for HybridGO-Loc

10.4.1 Semantic Similarity Measures

For HybridGO-Loc, we have compared three of the most common semantic similarity measures for subcellular localization, including Lin's measure [174], Jiang's measure [192], and relevance similarity measure [175].[3] In addition to these measures, many online tools are also available for computing the semantic similarity at the GO-term level and gene-product level [233, 234, 235, 98]. However, these measures are discrete measures whereas the measures that we used are continuous. Research has shown that continuous measures are better than discrete measures in many applications [172].

10.4.2 GO-Frequency Features versus SS Features

Note that when hybridizing GO information, we do not replace the GO frequency vectors. Instead, we augment the GO frequency feature with a more sophisticated feature, i.e.

[3]We excluded Resnik's measure because it ignores the distance between the terms and their common ancestors in the GO hierarchy.

the GO SS vectors, which are to be combined with the GO frequency vectors. A GO frequency vector is found by counting the number of occurrences of every GO term in a set of distinct GO terms obtained from the training dataset, whereas an SS vector is constructed by computing the semantic similarity between a test protein with each of the training proteins at the gene-product level. That is, each element in an SS vector represents the semantic similarity of two GO-term groups. This can be easily seen from their definitions in Eq. 4.3 and Eq. 6.7–6.12, respectively.

The GO frequency vectors and the GO SS vectors are different in two fundamental ways.

A) GO frequency vectors are more *primitive* in the sense that their elements are based on individual GO terms without considering the inter-term relationship, i.e., the elements in a GO frequency vectors are independent of each other.

B) GO SS vectors are more *sophisticated* in the following two senses:

  B1) *Inter-term relationship.* SS vectors are based on inter-term relationships. They are defined on a space in which each basis corresponds to one training protein and the coordinate along that basis is defined by the semantic similarity between a testing protein and the corresponding training protein.

  B2) *Inter-group relationship.* The pairwise relationships between a test protein and the training proteins are hierarchically structured. This is because each basis of the SS space depends on a group of GO terms of the corresponding training protein, and the terms are arranged in a hierarchical structure (parent-child relationship). Because the GO terms in different groups are not mutually exclusive, the bases in the SS space are not independent of each other.

10.4.3 Bias Analysis

Except for the new plant dataset, we adopted LOOCV to examine the performance of HybridGO-Loc, which is considered to be the most rigorous and bias-free [223]. Nevertheless, determining the set of distinct GO terms $\mathbb{W}$ (in Section 4.1.3) from a dataset is by no means without bias, which may favor the LOOCV performance. This is because the set of distinct GO terms $\mathbb{W}$ derived from a given dataset may not be representative for other datasets; in other words, the generalization capabilities of the predictors may be weakened when new GO terms outside $\mathbb{W}$ are found in the test proteins.

However, we have the following strategies to minimize the bias. First, the two multi-label benchmark datasets used for HybridGO-Loc were constructed based on the whole Swiss-Prot database (although in different years), which, to some extent, incorporated all the possible information of plant proteins or virus proteins in the database. In other words, $\mathbb{W}$ was constructed based on all of the GO terms corresponding to the whole Swiss-Prot database, which enables $\mathbb{W}$ to be representative for all of the distinct GO terms. Second, these two benchmark datasets were collected according to strict criteria (see Section 8.2.1) and the sequence similarity of both datasets was cut off at 25%, which enables us to use a small set of representative proteins to represent all of the proteins of the corresponding species (i.e., virus or plant) in the whole database. In other words, $\mathbb{W}$ will vary from species to species, yet still be statistically representative for all of the useful GO terms for the corresponding species. Third, using $\mathbb{W}$ for statistical performance evaluation is equivalent or at least approximate to using all of the distinct GO terms in the GOA database. This is because other GO terms that do not correspond to the training proteins will not participate in training the linear SVMs, nor will they play essential roles in contributing to the final predictions. In other words, the generalization capabilities of

HybridGO-Loc will not be weakened even if some new GO terms are found in the test proteins. A mathematical proof of this statement can be found in the Appendix B.

One may argue that the performance bias might arise when the whole $\mathbb{W}$ was used to construct the hybrid GO vectors for both training and testing during cross validation. This is because, in each fold of the LOOCV, the training proteins and the singled-out test protein will use the same $\mathbb{W}$ to construct the GO vectors, meaning that the SVM training algorithm can *see* some information of the test protein indirectly through the GO vector space defined by $\mathbb{W}$. It is possible that for a particular fold of LOOCV, the GO terms of a test protein do not exist in any of the training proteins. However, we have mathematically proved that this bias will not exist during LOOCV (see the Appendix B for proof). Furthermore, the results of the independent tests (See Table 8.8) for which no such bias occurs also strongly suggest that HybridGO-Loc outperforms other predictors by a large margin.

10.5 Analysis for RP-SVM

10.5.1 Legitimacy of Using RP

As stated in [200], if $\mathbf{R}$ and $\mathbf{q}_i$ satisfy the conditions of the basis pursuit theorem (i.e., both are sparse in a fixed basis), then $\mathbf{q}_i$ can be reconstructed perfectly from a vector that lies in a lower-dimensional space.

In fact, the GO vectors and our projected matrix $\mathbf{R}$ satisfy these conditions. Here we use the plant dataset (Table 8.6 in Chapter 8) as an example. There are 978 proteins distributed in 12 subcellular locations. After feature extraction, the dimension of the GO vectors is 1541. The distribution of the number of non-zero entries in the GO vectors are shown in Fig. 10.1. As shown in Fig. 10.1, the number of non-zero entries in the

GO vectors tends to be small (i.e. sparse) when compared to the dimension of the GO vectors. Among the 978 proteins in the dataset, a majority of them only have 9 non-zero entries in the 1541-dimensional vectors, and the largest number of non-zero entries is only 45. These statistics suggest that the GO vectors $\mathbf{q}_i$ in Eq. 7.1 are very sparse. Therefore, according to [200], RP is very suitable to be applied to GO vectors for dimensionality reduction.

10.5.2 Ensemble Random Projection for Robust Performance

Since $\mathbf{R}$ in Eq. 7.1 is a random matrix, the scores in Eq. 7.3 for each application of RP will be different. To construct a robust classifier, we fused the scores for several applications of RP and obtained an ensemble classifier, which was specified in Eq. 7.5. Actually, the performance achieved by a single application of random-projection, or single RP, varies considerably, as evident in Fig. 9.8.3. Therefore, single RP was not conducive to final prediction. However, by combining several applications of RP, the performance of RP-SVM can outperform mGOASVM, if the number of applications of RP is large enough and the projected dimensionality is no less than a certain value. These results demonstrate the significance of ensemble RP for boosting the final performance of RP-SVM. Besides, the results reveal that there is some tradeoff between the number of applications of RP and the projected dimensionality to guarantee an improved performance. It is also evident that RP can be easily applied to other methods, such as R3P-Loc in Section 7.3.

Nevertheless, some interesting questions remain unanswered: (1) Is there a threshold for the projected dimensionality above which RP-SVM will always outperform mGOASVM, or the performance by the ensemble RP will always be superior to that by the original features? (2) How to determine the threshold of the projected dimensionality? (3) At least how many applications of RP are needed, if the designated projected dimensionality

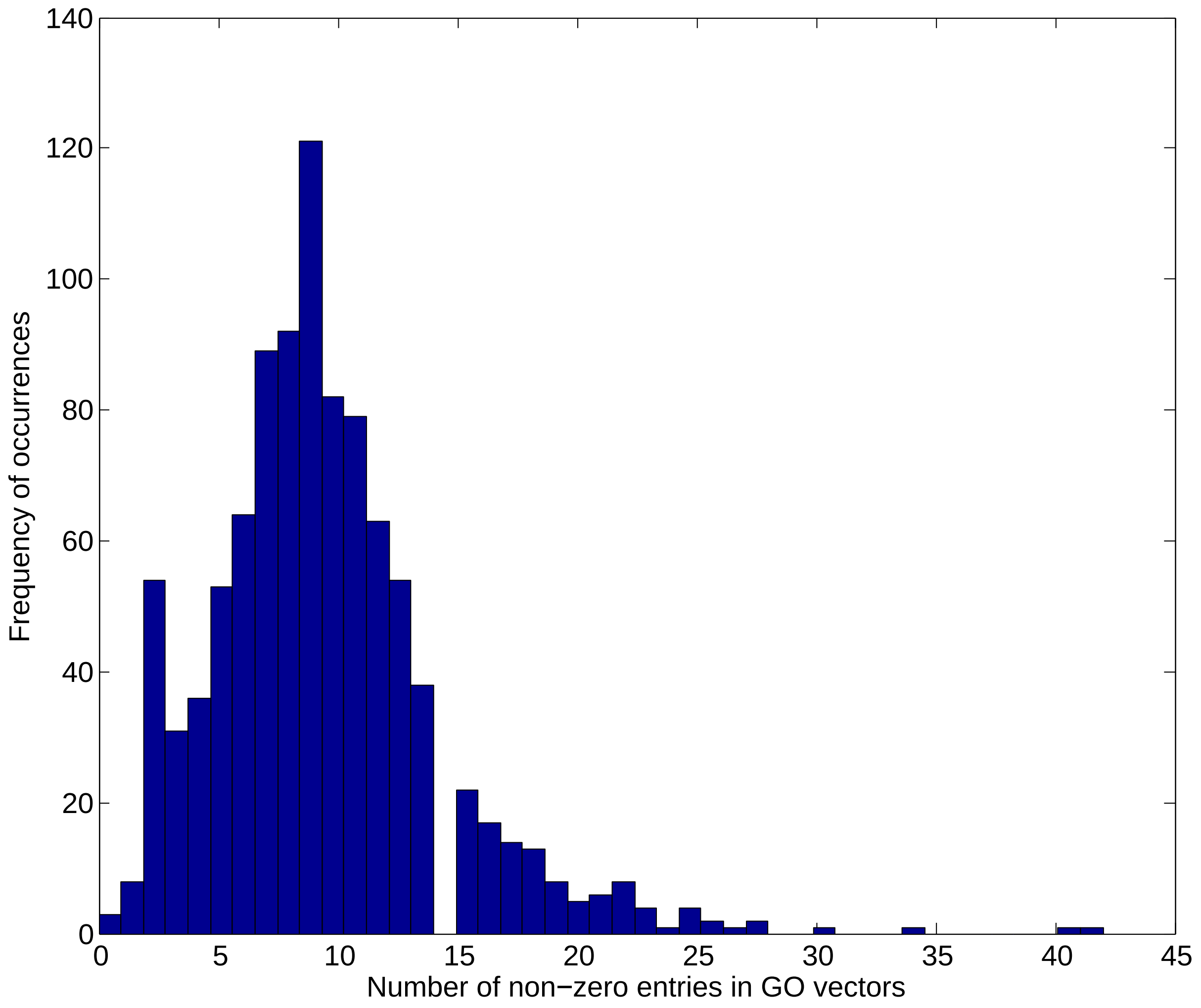


Figure 10.1: Histogram illustrating the distribution of the number of non-zero entries (spareness) in the GO vectors with dimensionality 1541. The histogram is plotted up to 45 non-zero entries in the GO vectors because among the 978 proteins in the dataset, none of their GO vectors have more than 45 non-zero entries.

is above the threshold, to guarantee a better performance of RP-SVM? (4) Can the projected dimensionality and the number of applications of RP be optimized to achieve the best possible performance of RP-SVM? These are possible some of the future directions for applying ensemble random projection to multi-label classification.

Table 10.1: Comparing the properties of the proposed multi-label predictors with state-of-the-art predictors: Virus-mPLoc [154], iLoc-Virus [93], Plant-mPLoc [96], iLoc-Plant [87], Euk-mPLoc 2.0 [158] and iLoc-Euk [90]. *Term Freq.*: term-frequency GO-vector construction method (Eq 4.3 in Section 4.1.3); *Succ. Search*: the new successive-search strategy to retrieve GO terms (Section 4.1.2); *Clas. Refinement*: multi-label classifier improvement or trying other efficient multi-label classifiers, compared to the baseline of multi-label SVM classifiers used in mGOASVM (Chapter 5); *Deeper Features*: using deeper GO features, i.e. GO semantic similarity features or hybridizing both GO frequency and semantic similarity features (Chapter 6); *Dim. Reduction*: using dimension-reduction methods, i.e. ensemble random projection for improving the performance of predictors (Chapter 7). '✗' means the predictor does not have the corresponding advantage; '✓' means the predictor has the corresponding advantage.

Predictors	Term Freq.	Succ. Search	Clas. Refinement	Deeper Features	Dim. Reduction
Virus-mPLoc	✗	✗	✗	✗	✗
iLoc-Virus	✗	✗	✗	✗	✗
Plant-mPLoc	✗	✗	✗	✗	✗
iLoc-Plant	✗	✗	✗	✗	✗
Euk-mPLoc 2.0	✗	✗	✗	✗	✗
iLoc-Euk	✗	✗	✗	✗	✗
mGOASVM	✓	✓	✗	✗	✗
AD-SVM	✓	✓	✓	✗	✗
mPLR-Loc	✓	✓	✓	✗	✗
SS-loc	✗	✓	✗	✓	✗
HybridGO-Loc	✓	✓	✓	✓	✗
RP-SVM	✓	✓	✓	✗	✓
R3P-Loc	✓	✓	✓	✗	✓

10.6 Comparing the Proposed Multi-Label Predictors

To further compare the advantages and disadvantages of all of the proposed multi-label predictors with those of state-of-the-art predictors, Table 10.1 summarized five perspectives that contribute most to the superiority of the proposed predictors over the state-of-the-art predictors. These five perspectives are: (1) whether the predictor of interest uses *Term Frequency*, namely the term-frequency GO-vector construction method (See Eq 4.3

in Section 4.1.3); (2) whether the predictor of interest uses *Successive Search*, namely the new successive-search strategy to retrieve GO terms (See Section 4.1.2); (3) whether the predictor of interest does *Classification Refinement*, namely multi-label classifier improvement or trying other efficient multi-label classifiers; (4) whether the predictor of interest uses *Deeper Features*, i.e. GO semantic similarity features or hybridizing both GO frequency and semantic similarity features (See Chapter 6); and (5) whether the predictor of interest uses *Dimension Reduction*, i.e. ensemble random projection for improving the performance of predictors (See Chapter 7).

As can be seen from Table 10.1, all of the proposed predictors have adopted the new successive-search strategy to avoid null GO vectors. On the other hand, Virus-mPLoc, iLoc-Virus, Plant-mPLoc, iLoc-Plant, Euk-mPLoc 2.0 and iLoc-Euk need to use back-up methods whenever null GO vectors occur, which are likely to make the prediction performance poorer than that by using only GO information. Among the proposed predictors, all predictors except SS-Loc do not use term-frequency method for constructing feature vectors. In terms of classifier refinement, AD-SVM and HybridGO-loc use an adaptive decision scheme based on the multi-label SVM classifier used in mGOASVM, while mPLR-Loc and R3P-Loc use multi-label penalized logistic regression and ridge regression classifiers, respectively. For deeper features, HybridGO-Loc and SS-Loc exploit GO semantic similarity for classification. As for dimensionality reduction, RP-SVM and R3P-Loc adopt ensemble random projection for reducing high dimensionality of feature vectors while at the same time boosting the prediction performance. As stated in Chapter 9, mining deeper into the GOA database would probably contribute more to the increase in the prediction accuracy of the predictors.

10.7 Summary

This chapter gave a further discussion about the advantages and limitations of proposed multi-label predictors, including mGOASVM, AD-SVM, mPLR-Loc, SS-Loc, HybridGO-Loc, RP-SVM and R3P-Loc. From the perspectives of refinement of multi-label classifiers, mGOASVM, AD-SVM and mPLR-Loc belong to the same category and share the same way of feature extraction, but with different multi-label classifiers. From the perspectives of mining deeper into the GO information, SS-loc and HybridGO-Loc exploit semantic similarity over GO information. From the perspectives of dimension reduction, RP-SVM and R3P-Loc apply ensemble random projection to reduce dimensionality and boost the performance at the same time. Among all of the multi-label predictors, HybridGO-Loc performs the best, demonstrating that mining deeper into the GOA database can significantly boost the prediction performance.

Chapter 11: Conclusions and Future Work

The final chapter of a thesis summarises the main findings of the research. It often also includes comments on limitations of the study, future work and how the findings will help both the academic field and the wider community.

This chapter is very effective partly because the writer includes the following:

Structure

(Introduction)	Not included
Conclusions	Section 11.1
Contributions and limitations	Section 11.2
Recommendation for Future Research	Section 1 1.3
(Summary)	Not included

Content

- Summarises each of the main findings (e.g. Section 11.1)
- Summarises the most important finding without unnecessary detail (e.g. Section 11.1, paragraph 5, sentence 1)
- Develops sections logically, e.g. Section 11.1:

Paragraph 1	Introduces the topic of the section
Paragraph 2	Compares and contrasts findings of models in theme 1
Paragraph 3	Compares and contrasts findings of models in theme 2
Paragraph 4	Explains how these findings were developed
Paragraph 5	Explains the importance of the findings

- Highlights the contribution the study has made (e.g. Section 11.2, paragraph 1 bullet points 1-6)

- Outlines limitations of the study (e.g. Section 11.2, paragraph 2, bullet points 1-2). Explains possible reasons for problems (e.g. Section 11.2, paragraph 2 bullet point 2, sentence 2)
- Outlines future work (e.g. Section 11.3)
- Explains how the results fill a gap in knowledge

Language

- Uses vocabulary the importance of the results e.g. *remarkably* (e.g. Section 11.1, paragraph 2, sentence 2)
- Lists key contributions in a vertical list (e.g. Section 11.2, paragraph 1)
- Highlights the limitations of the study and attempts to explain them using tentative language (e.g. Section 11.2, paragraph 2)

To Consider

This chapter of the thesis is effective. However, it could be further improved in the following aspects.

Start the chapter by briefly referring the reader to the objective of the research and provide a brief overview of the contents of the chapter.

Avoid introducing paragraphs with *for,* e.g. *for predicting single-location proteins,* (e.g. Section 11.1 paragraph 2 sentence 1 and paragraph 3, sentence 1).

Cite the most important authors that appear in the Literature Review and compare the findings of this study directly with their results. This should not be over detailed: at this stage of the thesis, a concise summary is expected.

End with a summary paragraph that includes a one sentence overall summary of the thesis followed by highlighting the importance of future studies in this field.

State future work as something that is required for the whole field.

Chapter 11

Conclusions and Future Work

11.1 Conclusions

This thesis presents several GO-based accurate predictors for subcellular localization of both single- and multi-location proteins.

For predicting single-location proteins, two predictors, namely **GOASVM** and **FusionSVM** are presented, which differ mainly in the way to retrieve GO information. **GOASVM** and **FusionSVM** extract the GO information from the GOA database and the InterProScan, respectively. To enhance the prediction performance, **FusionSVM** fuses the GO-based predictor–InterProScanSVM–with profile-alignment based method PairProSVM. Nevertheless, **GOASVM** still remarkably outperforms **FusionSVM**. Experimental results also show the superiority of **GOASVM** over other state-of-the-art single-label predictors.

For predicting multi-location proteins, several advanced predictors are proposed. First, **mGOASVM** is presented, which performs remarkably better than existing state-of-the-art predictors with the following advantages: (1) in terms of the GO-vector construction method, it uses term-frequency instead of conventional 1-0 value; (2) in terms of handling multi-label problems, it uses a more efficient multi-label SVM classifier than other classifiers; (3) in terms of GO-vector space selection, it selects a relevant GO-vector subspace by finding a set of distinct GO terms instead of using all of the GO terms to define the full space; and (4) in terms of retrieving GO information, it uses a successive-search strategy to incorporate more useful homologs instead of using back-up methods.

Based on these findings, several multi-label predictors are developed and enhancements from different perspectives are made, including (1) *refining classifiers*, such as **AD-SVM** which refines the multi-label SVM classifier with an adaptive decision scheme and **mPLR-Loc** which develops a multi-label penalized logistic regression classifier; (2) *exploiting deeper features*, such as **SS-Loc** which formulates the feature vectors from the GO semantic similarity (SS) information, and **HybridGO-Loc**, which hybridizes the GO frequency and GO SS features for better performance; and (3) *reducing the high dimensions of feature vectors*, such as **RP-SVM**, which applies ensemble random projection (RP) to multi-label SVM classifiers, and **R3P-Loc**, which combines ensemble random projection with ridge regression classifiers for multi-label prediction. Particularly, for **R3P-Loc**, instead of using the successive-search strategy, it creates two compact databases, namely ProSeq and ProSeq-GO, to replace the traditional Swiss-Prot and GOA database for efficient and fast retrieval of GO information.

Experimental results based on several benchmark datasets and novel datasets from species of virus, plant and eukaryote demonstrate that the proposed predictors can significantly outperform existing state-of-the-art predictors. In particular, among the proposed predictors, **HybridGO-Loc** performs the best, suggesting that mining deeper into the GO information can contribute more to boosting the prediction performance than classifier refinement and dimensionality reduction.

11.2 Contributions and Limitations

The key contributions of this thesis are summarized as follows:

1. This dissertation proposes using term-frequency information of GO terms for constructing GO vectors instead of using the traditional 1-0 value method.

2. This dissertation adopts a successive-search strategy to incorporate distant homologous information so that the use of back-up methods can be largely avoided.

3. This dissertation uses an adaptive-decision scheme for refining the multi-label classification process.

4. This dissertation extracts semantic similarity information from the GO database for deeper feature mining.

5. This dissertation utilizes an ensemble of random projections for reducing the dimensionality of GO vectors.

6. This dissertation replaces the Swiss-Prot and GOA databases by two compact databases for large-scale and efficient homologous information retrieval and GO-term search.

Despite of various contributions we made in this thesis, there are still some limitations that are worth noting, which are elaborated below:

1. Although remarkable performance improvement has been achieved by the predictors proposed in this dissertation, the biological significance of the predictors remains uncertain. This is possibly a common problem for machine-learning based approaches because it is usually difficult to correlate mathematical mechanism of machine-learning approaches with biological phenomena.

2. For the prediction of novel proteins, although our proposed predictors perform better than many state-of-the-art predictors, the overall accuracies are lower than the ones that can be achieved if the older (benchmark) datasets were used for evaluation. This is possibly because some novel proteins have very low sequence similarity with known proteins in sequence databases; and more importantly they may possess some new information that has not been incorporated in the current GO databases, causing incorrect prediction for these proteins. The situation may be improved when the GO databases continue to evolve.

11.3 Future Work

To deal with the limitations discussed above, we plan to do further search in the following perspectives:

1. We will try some algorithms which can yield sparse solutions for our prediction, so that the classification results can be interpreted easily. In other words, some biological significance may be found. For example, we may find which GO terms may play more important roles in determining the classification decisions, to what extent GO terms in categories of molecular functions and biological processes contribute to the performance, etc.

2. We will try to extract deeper features on GO information to boost the prediction performance. As stated in this thesis, the performance can be improved by incorporating the GO semantic similarity information. Therefore, it is likely that extracting further GO information, either through adopting more efficient semantic similarity measurements or by incorporating more GO-related information, can further boost the performance. Recently, the GO consortium introduced more com-

plicated relationships in the GO hierarchical graphs, such as ‘positively-regulates’, ‘negatively-regulates’, ‘has-part’, etc [165]. These relationships could be included in our GO feature vectors to reflect more biological-related information of GO terms, leading to better prediction performance.

Appendix A

Web-Servers for Protein Subcellular Localization

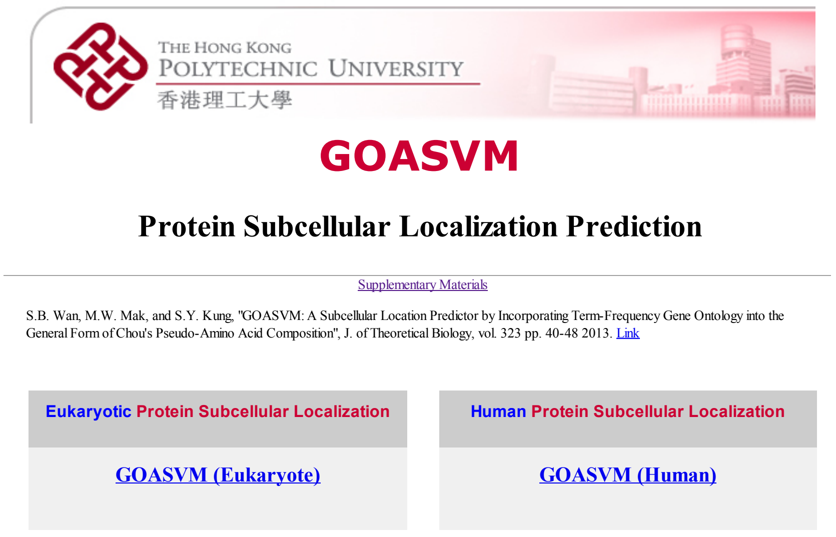
This appendix will simply introduce the online web-servers for some proposed predictors, namely GOASVM, mGOASVM, mPLR-Loc and HybridGO-Loc.

A.1 GOASVM Web-Server

The GOASVM web-server (See Fig. A.1) is to predict subcellular localization for single-label eukaryotic proteins or single-label human proteins. The URL link for GOASVM server is `http://bioinfo.eie.polyu.edu.hk/mGoaSvmServer/GOASVM.html`.

For eukaryotic proteins, GOASVM is designed to predict 16 subcellular locations of eukaryotic proteins. The 16 subcellular locations include: (1) cell wall; (2) centriole; (3) chloroplast; (4) cyanelle; (5) cytoplasm; (6) cytoskeleton; (7) endoplasmic reticulum; (8) extracellular; (9) Golgi apparatus; (10) lysosome; (11) mitochondrion; (12) nucleus; (13) peroxisome; (14) plasma membrane; (15) plastid; (16) vacuole.

For human proteins, GOASVM is designed to predict 12 subcellular locations of human proteins. The 12 subcellular locations include: (1) centriole; (2) cytoplasm; (3) cytoskeleton; (4) endoplasmic reticulum; (5) extracellular; (6) Golgi apparatus; (7) lysosome; (8)


THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學
GOASVM
Protein Subcellular Localization Prediction
Supplementary Materials

S.B. 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", J. of Theoretical Biology, vol. 323 pp. 40-48 2013. Link

Eukaryotic Protein Subcellular Localization
GOASVM (Eukaryote)
Human Protein Subcellular Localization
GOASVM (Human)

Figure A.1: The interface of GOASVM web-server.

microsome; (9) mitochondrion; (10) nucleus; (11) peroxisome; (12) plasma membrane.

GOASVM can deal with two different input types of proteins (See Fig. A.2), either protein accession numbers (ACs) in UniProtKB format or amino acid sequences in FASTA format. Large-scale predictions, i.e. a list of accession numbers or a number of protein sequences, are also acceptable for GOASVM. Examples for both cases are also provided. More information can be found in the instructions and supplementary materials on the GOASVM web-server.

A.2 mGOASVM Web-Server

The mGOASVM web-server (See Fig. A.3) is to predict subcellular localization for both single-label and multi-label proteins in two species (i.e., virus and plant). The URL link for mGOASVM server is `http://bioinfo.eie.polyu.edu.hk/mGoaSvmServer/mGOASVM.html`. Note that two different versions of mGOASVM are provided, one based on the GOA database released in March 2011 and one based on that released in July 2013. Typi-

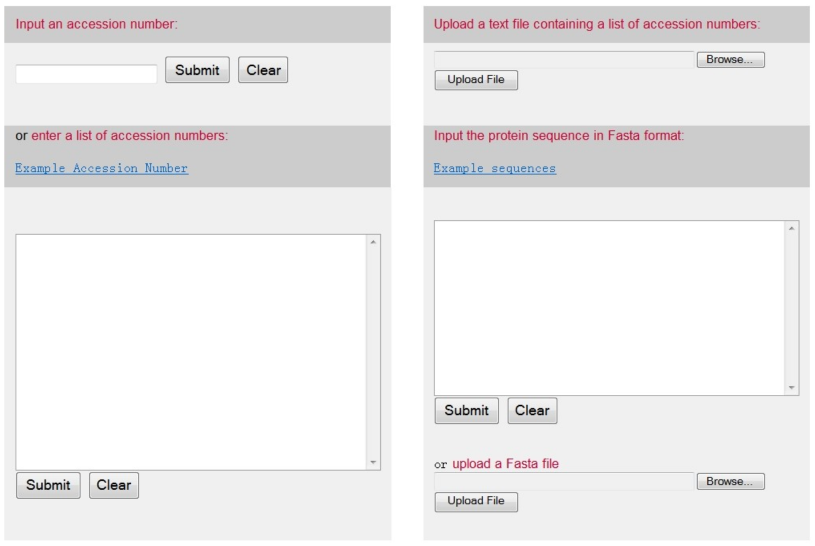
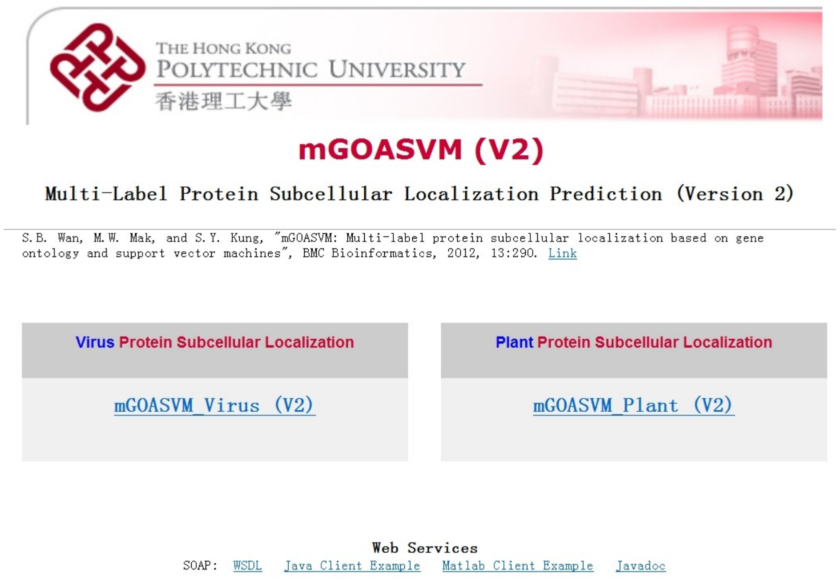

Input an accession number:
Submit
Clear
or enter a list of accession numbers:
Example Accession Number
Submit
Clear
Upload a text file containing a list of accession numbers:
Browse...
Upload File
Input the protein sequence in Fasta format:
Example sequences
Submit
Clear
or upload a Fasta file
Browse...
Upload File

Figure A.2: A snapshot of GOASVM web-server showing that GOASVM can deal with either protein ACs or protein sequences.


THE HONG KONG POLYTECHNIC UNIVERSITY 香港理工大學

mGOASVM (V2)

Multi-Label Protein Subcellular Localization Prediction (Version 2)

S.B. Wan, M.W. Mak, and S.Y. Kung, "mGOASVM: Multi-label protein subcellular localization based on gene ontology and support vector machines", BMC Bioinformatics, 2012, 13:290. Link

Virus Protein Subcellular Localization

mGOASVM_Virus (V2)

Plant Protein Subcellular Localization

mGOASVM_Plant (V2)

Web Services

SOAP: WSDL Java Client Example Matlab Client Example Javadoc

Figure A.3: The interface of mGOASVM web-server.

cally, using the latest version will give more accurate prediction for protein subcellular localization.

For virus proteins, mGOASVM is designed to predict 6 subcellular locations of multi-label viral proteins. The 6 subcellular locations include: (1) viral capsid; (2) host cell membrane; (3) host endoplasmic reticulum; (4) host cytoplasm; (5) host nucleus; (6) secreted.

For plant proteins, mGOASVM is designed to predict 12 subcellular locations of multi-label plant proteins. The 12 subcellular locations include: (1) cell membrane; (2) cell wall; (3) chloroplast; (4) cytoplasm; (5) endoplasmic reticulum; (6) extracellular; (7) golgi apparatus; (8) mitochondrion; (9) nucleus; (10) peroxisome; (11) plastid; (12) vacuole.

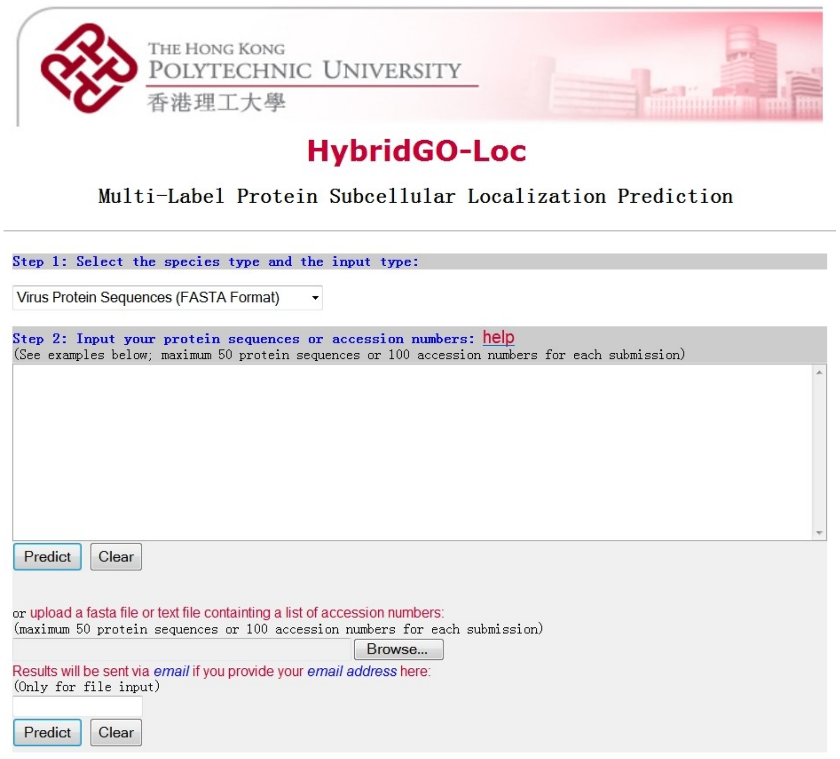
Like GOASVM, mGOASVM can deal with two different input types, either protein ACs or protein sequences. More information can be found in the instructions and supplementary materials on the mGOASVM web-server.

A.3 HybridGO-Loc Web-Server

Like mGOASVM, HybridGO-Loc (See Fig. A.4) is a subcellular-localization predictor which can deal with datasets with both single-label and multi-label proteins in two species (virus and plant) and two input types (protein ACs and protein sequences). The URL link for mGOASVM server is `http://bioinfo.eie.polyu.edu.hk/HybridGoServer/`. Also, the specific subcellular locations that HybridGO can predict for both species are the same as those in mGOASVM.

Different from mGOASVM, HybridGO-Loc integrates all possible combinations of different species and input types in one interface. Users can just follow two steps to make predictions on HybridGO-Loc: (1) select the species type and the input type (virus protein ACs, virus protein sequences, plant protein ACs or plant protein sequences); (2) input your protein sequences or accession numbers.

What's more, users can leave their email address in the corresponding space to receive their prediction results via email. For ease of dealing with large-scale prediction results,


THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學
HybridGO-Loc
Multi-Label Protein Subcellular Localization Prediction
Predict
Clear
Browse...

Step 1: Select the species type and the input type:

Virus Protein Sequences (FASTA Format)

Step 2: Input your protein sequences or accession numbers: help
(See examples below; maximum 50 protein sequences or 100 accession numbers for each submission)

or upload a fasta file or text file containting a list of accession numbers:
(maximum 50 protein sequences or 100 accession numbers for each submission)

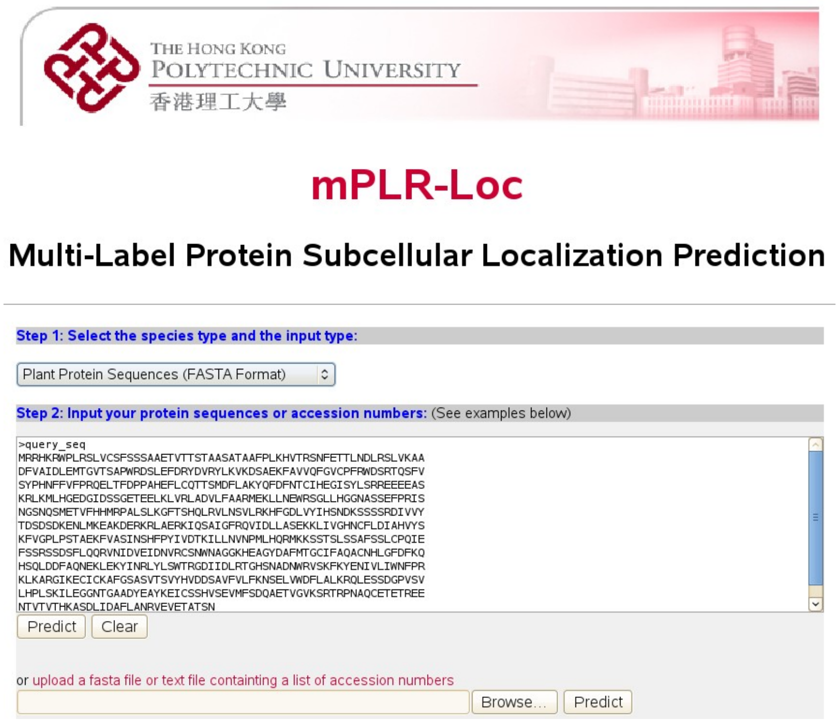
Results will be sent via email if you provide your email address here:
(Only for file input)

Figure A.4: The interface of HybridGO-Loc web-server.

a downloadable txt file will also be given on the webpage every time a prediction task is over. Detailed and comprehensive supplementary materials as well as instructions (also on the web-server page) are also provided for guiding users to use the HybridGO-Loc server.

A.4 mPLR-Loc Web-Server

The mPLR-Loc web-server also possess the capability of predicting single- and multi-location proteins in virus and plant species. Similar to HybridGO-Loc, the mPLR-Loc web-server also integrates all possible different inputs (combinations of different species and input types) in one interface. The URL link for mPLR-Loc server is `http://bioinfo.eie.polyu.edu.hk/mPLRLocServer/`. In addition to being able to rapidly and accurately



Step 1: Select the species type and the input type:

Plant Protein Sequences (FASTA Format)

Step 2: Input your protein sequences or accession numbers: (See examples below)

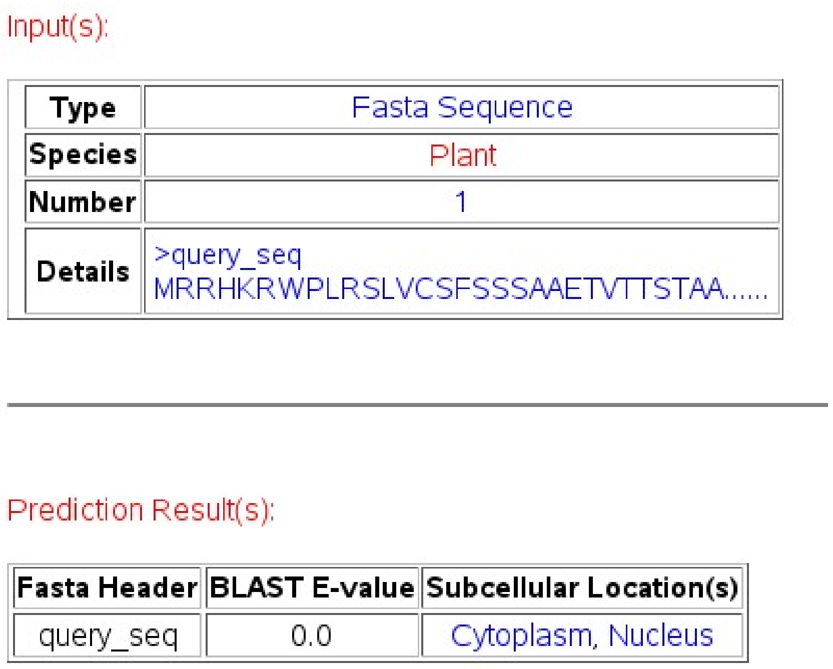
```
>query_seq
MRRHKRWPLRSLVCSFSSSAAETVTTSTAASATAAFPLKHVTRSNFETTLNDLRSLVKAA
DFVAIDLEMTGVTSAPWRDSLEFDRYDVRYLKVKDSAEKFAVVQFGVCPFRWDSRTQSFV
SYPHNFFVFPRQELTFDPPAHEFLCQTTSMDFLAKYQFDFNTCIHEGISYLSRREEEEAS
KRLKMLHGEDGIDSSGETEELKLVRLADVLFAARMEKLLNEWRSGLLHGGNASSEFPRIS
NGSNQSMETVFHHMRPALSLKGFTSHQLRVLNSVLRKHFGDLVYIHSNDKSSSSRDIVVY
TDSDSDKENLMKEAKDERKRLAERKIQSAIGFRQVIDLLASEKKLIVGHNCFLDIAHVYS
KFVGPLPSTAEKFVASINSHFPYIVDTKILLNVNPMLHQRMKKSSTSLSSAFSSLCPQIE
FSSRSSDSFLQQRVNIDVEIDNVRCSNWNAGGKHEAGYDAFMTGCIFAQACNHLGFDFKQ
HSQLDDFAQNEKLEKYINRLYLSWTRGDIIDLRTGHSNADNWRVSKFKYENIVLIWNFPR
KLKARGIKECICKAFGSASVTSVYHVDDSAVFVLFKNSELVWDFLALKRQLESSDGPVSV
LHPLSKILEGGNTGAADYEAYKEICSSHVSEVMFSDQAETVGVKSRTRPNAQCETETREE
NTVTVTHKASDLIDAFLANRVEVETATSN
```

or upload a fasta file or text file containting a list of accession numbers

Figure A.5: An example of using a plant protein sequence in Fasta format as input to the mPLR-Loc server.

predict subcellular localization of single- and multi-label proteins, mPLR-Loc can also provide probabilistic confidence scores for the prediction decisions.

Here a step-by-step guide on how to use the mPLR-Loc is provided. After going to the homepage of mPLR-Loc server, select a combination of species type and input type. Then input the query protein sequences or accession numbers or upload a file containing a list of accession numbers or proteins sequences. For example, Fig. A.5 shows the screenshot that uses a plant protein sequence in Fasta format as input. After clicking the button 'Predict' and waiting for around 13s, the prediction results as shown in Fig. A.6 and the probabilistic scores as shown in Fig. A.7 will be produced. The prediction result in Fig. A.6 include the Fasta header, BLAST E-value and predicted subcellular location(s). Fig. A.7 shows the confidence on the predicted subcellular location(s). In this figure,



Input(s):

Type	Fasta Sequence
Species	Plant
Number	1
Details	>query_seq MRRHKRWPLRSLVCSFSSSAAETVTTSTAA......

Prediction Result(s):

Fasta Header	BLAST E-value	Subcellular Location(s)
query_seq	0.0	Cytoplasm, Nucleus

Figure A.6: Prediction results of the mPLR-Loc server for the plant protein sequence input in Fig. A.5.

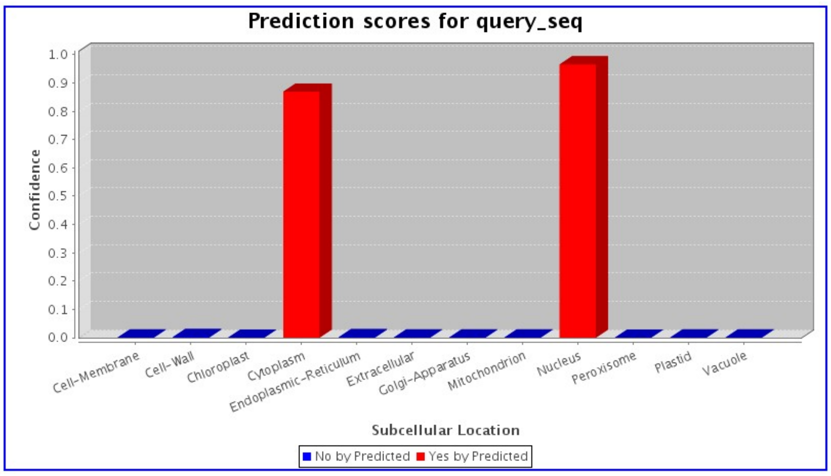


Figure A.7: Confidence scores of the mPLR-Loc server for the plant protein sequence input in Fig. A.5.

mPLR-Loc predicts the query sequence as ‘Cytoplasm’ and ‘Nucleus’ with confidence scores greater than 0.8 and 0.9, respectively.

Appendix B

Proof of No Bias in LOOCV

This appendix is to prove that when the whole $\mathbb{W}$ is used to construct the feature vectors (i.e. GO vectors) for both training and testing, there will be no bias during leave-one-out cross-validation (LOOCV) even if some new features (i.e., GO terms) are retrieved for a test sample (i.e., a test protein). Here 'new' means that the corresponding features (or GO terms) do not exist in any of the training samples (or proteins), but they are found in the test sample(s) (or test protein(s)).

Suppose a set of labelled samples are denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,n}$, where the i-th sample $\mathbf{x}_i$ is drawn from a d-dimensional domain $\mathcal{X} \in \mathcal{R}^d$ and the corresponding label $y_i \in \{-1, +1\}$. The soft-margin Lagrangian for SVM is:

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i(y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{n} \beta_i \xi_i, \tag{B.1}$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$. By differentiating $\mathcal{L}$ with respect to $\mathbf{w}$ and b, it can be shown that Eq. B.1 is equivalent to the following optimizer:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{B.2}$$

subject to $0 \leq \alpha_i \leq C, i = 1, \ldots, n, \sum_{i=1}^{n} \alpha_i y_i = 0$.

During LOOCV, if a new GO term is found in the test protein, then during the training part, we extend the d-dim feature vectors to $(d+1)$-dim to incorporate the GO term with the corresponding entry being 0, namely $\mathbf{x}_i$ becomes $\mathbf{x}'_i = \begin{bmatrix} \mathbf{x}_i \\ 0 \end{bmatrix}$.

Then, Eq. B.2 becomes:

$$\begin{aligned}\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j (\mathbf{x}'_i \cdot \mathbf{x}'_j) &= \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j + 0 \cdot 0) \\ &= \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j),\end{aligned} \tag{B.3}$$

subject to $0 \le \alpha_i \le C, i = 1, \ldots, n, \sum_{i=1}^{n} \alpha_i y_i = 0$. Therefore, α_i will not be affected by the extended feature vectors.

Based on this, the weight can be obtained as:

$$\begin{aligned}\mathbf{w}' &= \sum_{i=1}^{n} \alpha'_i y_i \mathbf{x}'_i \\ &= \sum_{i=1}^{n} \alpha_i y_i \begin{bmatrix} \mathbf{x}_i \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{w} \\ 0 \end{bmatrix}\end{aligned} \tag{B.4}$$

and the bias b can be expressed as:

$$\begin{aligned}b' &= 1 - \mathbf{w}'^{\mathsf{T}}\mathbf{x}'_k \\ &= 1 - [\mathbf{w}\ 0] \cdot \begin{bmatrix} \mathbf{x}_k \\ 0 \end{bmatrix} \\ &= 1 - \mathbf{w}^{\mathsf{T}}\mathbf{x}_k \\ &= b\end{aligned} \tag{B.5}$$

where $\mathbf{x}'_k$ is any support vector whose label $y_k = 1$.

Therefore, for any test protein with a feature vector written as $\mathbf{x}'_t = \begin{bmatrix} \mathbf{x}_t \\ a_t \end{bmatrix}$, where $a_t \neq 0$, the SVM score is:

$$\begin{aligned} f(\mathbf{x}'_t) &= (\mathbf{w}')^{\mathsf{T}}\mathbf{x}'_t + b' \\ &= [\mathbf{w}\ 0] \begin{bmatrix} \mathbf{x}_t \\ a_t \end{bmatrix} + b \\ &= \mathbf{w}^{\mathsf{T}}\mathbf{x}_t + b \\ &= f(\mathbf{x}_t) \end{aligned} \tag{B.6}$$

In other words, using the extended feature vectors during LOOCV will not cause bias compared to using the original vectors.

Appendix C

Derivatives for Penalized Logistic Regression

This appendix is to show the derivations for Eq. 5.15 and Eq. 5.16.

In Section 5.5.1 of Chapter 5, to minimize $E(\boldsymbol{\beta})$, we may use the Newton-Raphson algorithm to obtain Eq. 5.14, where the first and second derivatives of $E(\boldsymbol{\beta})$ are as follows:

$$\begin{aligned}\frac{\partial E(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= -\sum_{i=1}^{N} \mathbf{x}_i(y_i - p(\mathbf{x}_i;\boldsymbol{\beta})) + \rho\boldsymbol{\beta} \\ &= -\mathbf{X}^{\mathsf{T}}(\mathbf{y}-\mathbf{p}) + \rho\boldsymbol{\beta}\end{aligned} \tag{C.1}$$

and

$$
\begin{aligned}
\frac{\partial^2 E(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\mathsf{T}} &= \sum_{i=1}^{N} \left[\frac{\partial \mathbf{x}_i p(\mathbf{x}_i; \boldsymbol{\beta})}{\partial \boldsymbol{\beta}^\mathsf{T}} \right] + \rho \mathbf{I} \\
&= \sum_{i=1}^{N} \mathbf{x}_i \left[\frac{\partial}{\partial \boldsymbol{\beta}^\mathsf{T}} \left(\frac{e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}} \right) \right] + \rho \mathbf{I} \\
&= \sum_{i=1}^{N} \mathbf{x}_i \left[\frac{e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i} \mathbf{x}_i^\mathsf{T} (1 + e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}) - e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i} e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i} \mathbf{x}_i^\mathsf{T}}{(1 + e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i})^2} \right] + \rho \mathbf{I} \\
&= \sum_{i=1}^{N} \mathbf{x}_i \left[\frac{\mathbf{x}_i^\mathsf{T} e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}} \cdot \frac{1}{1 + e^{\boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i}} \right] + \rho \mathbf{I} \\
&= \sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^\mathsf{T} p(\mathbf{x}_i; \boldsymbol{\beta})(1 - p(\mathbf{x}_i; \boldsymbol{\beta})) + \rho \mathbf{I} \\
&= \mathbf{X}^\mathsf{T} \mathbf{W} \mathbf{X} + \rho \mathbf{I}.
\end{aligned}
\tag{C.2}
$$

In Eqs. C.1 and C.2, $\mathbf{y}$ and $\mathbf{p}$ are N-dim vectors whose elements are $\{y_i\}_{i=1}^N$ and $\{p(\mathbf{x}_i; \boldsymbol{\beta})\}_{i=1}^N$, respectively, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]^\mathsf{T}$, $\mathbf{W}$ is a diagonal matrix whose i-th diagonal element is $p(\mathbf{x}_i; \boldsymbol{\beta})(1 - p(\mathbf{x}_i; \boldsymbol{\beta})), i = 1, 2, \ldots, N$.

Bibliography

[1] C. L. Branden and J. Tooze, *Introduction to protein structure*, pp. 251–281, Garland Science, 1991. 1

[2] B. M. Gumbiner, "Cell adhesion: the molecular basis of tissue architecture and morphogenesis," *Cell*, vol. 84, no. 3, pp. 345–357, 1996. 2

[3] T. Bakheet and A. Doig, "Properties and identification of human protein drug targets," *Bioinformatics*, vol. 25, no. 4, pp. 451–457, 2009. 3

[4] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Molecular biology of the cell*, vol. 4, chapter 10-18, Garland Science, 2002. 3

[5] B. Rost, J. Liu, R. Nair, K. O. Wrzeszczynski, and Y. Ofran, "Automatic prediction of protein function," *Cellular and Molecular Life Sciences*, vol. 60, no. 12, pp. 2637–2650, 2003. 3

[6] G. S. Butler and C. M. Overall, "Proteomic identification of multitasking proteins in unexpected locations complicates drug targeting," *Nat. Rev. Drug Discov.*, vol. 8, pp. 935–948, 2009. 3

[7] M. D. Kaytor and S. T. Warren, "Aberrant Protein Deposition and Neurological Disease," *J. Biol. Chem.*, vol. 274, pp. 37507–37510, 1999. 3

[8] M. C. Hung and W. Link, "Protein localization in disease and therapy," *J. of Cell Sci.*, vol. 124, no. Pt 20, pp. 3381–3392, 2011. 3

[9] V. Krutovskikh, G. Mazzoleni, N. Mironov, Y. Omori, A. M. Aguelon, M. Mesnil, F. Berger, C. Partensky, and H. Yamasaki, "Altered homologous and heterologous gap-junctional intercellular communication in primary human liver tumors associated with aberrant protein localization but not gene mutation of connexin 32," *Int. J. Cancer*, vol. 56, pp. 87–94, 1994. 3

[10] Y. Chen, C. F. Chen, D. J. Riley, D. C. Allred, P. L. Chen, D. Von Hoff, C. K. Osborne, and W. H. Lee, "Aberrant Subcellular Localization of BRCA1 in Breast Cancer," *Science*, vol. 270, pp. 789–791, 1995. 3

[11] X. Lee, J. C. Jr Keith, N. Stumm, I. Moutsatsos, J. M. McCoy, C. P. Crum, D. Genest, D. Chin, C. Ehrenfels, R. Pijnenborg, F. A. Van Assche, and S. Mi, "Downregulation of placental syncytin expression and abnormal protein localization in pre-eclampsia," *Placenta*, vol. 22, pp. 808–812, 2001. 3

[12] A. Hayama, T. Rai, S. Sasaki, and S. Uchida, "Molecular mechanisms of Bartter syndrome caused by mutations in the BSND gene," *Histochem. and Cell Biol.*, vol. 119, no. 10, pp. 485–493, 2003. 3

[13] K. C. Chou and Y. D. Cai, "Predicting protein localization in budding yeast," *Bioinformatics*, vol. 21, pp. 944–950, 2005. 3, 10, 11

[14] G. Lubec, L Afjehi-Sadat, J. W. Yang, and J. P. John, "Searching for hypothetical proteins: Theory and practice based upon original data and literature," *Prog. Neurobiol*, vol. 77, pp. 90–127, 2005. 3, 9

[15] O.V. Stepanenko, V. V. Verkhusha, I. M. Kuznetsova, V. N. Uversky, and K. K. Turoverov, "Fluorescent proteins as biomarkers and biosensors: Throwing color lights on molecular and cellular processes," *Current Protein and Peptide Science*, vol. 9, no. 4, pp. 338–369, 2008. 4

[16] R. Yuste, "Fluorescence microscopy today," *Nature Methods*, vol. 2, no. 12, pp. 902–904, 2005. 4

[17] T. M. Mayhew and J. M. Lucocq, "Developments in cell biology for quantitative immunoelectron microscopy based on thin sections: a review," *Histochemistry and Cell Biology*, vol. 130, pp. 299–313, 2008. 4

[18] W. Margolin, "Green fluorescent protein as a reporter for macromolecular localization in bacterial cells," *Methods*, vol. 20, pp. 62–72, 2000. 4

[19] T. Kleffmann, D. Russenberger, A. von Zychlinski, W. Christopher, K. Sjolander, W. Gruissem, and S. Baginsky, "The arabidopsis thaliana chloroplast proteome reveals pathway abundance and novel protein functions," *Current Biology*, vol. 14, no. 5, pp. 354–362, 2004. 9

[20] H. Nakashima and K. Nishikawa, "Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies," *J. Mol. Biol.*, vol. 238, pp. 54–61, 1994. 10, 122, 127, 129

[21] K. J. Park and M. Kanehisa, "Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs," *Bioinformatics*, vol. 19, no. 13, pp. 1656–1663, 2003. 10, 11, 52, 122, 127, 129, 133

[22] K. Y. Lee, D. W. Kim, D. K. Na, K. H. Lee, and D. H. Lee, "PLPD: Reliable protein localization prediction from imbalanced and overlapped datasets," *Nucleic Acids Research*, vol. 34, no. 17, pp. 4655–4666, 2006. 10, 11, 81

[23] H. Nakashima and K. Nishikawa, "Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies," *J. Mol.Biol.*, pp. 54–61, 1994, 238. 10

[24] J. Cedano, P. Aloy, J. A. Perez-Pons, and E. Querol, "Relation between amino acid composition and cellular location of proteins," *J. Mol. Biol.*, vol. 266, pp. 594–600, 1997. 10

[25] A. Reinhardt and T. Hubbard, "Using neural networks for prediction of the subcellular location of proteins," *Nucleic Acids Res.*, vol. 26, pp. 2230–2236, 1998. 10, 52

[26] K. C. Chou and D. W. Elord, "Protein subcellular location prediction," *Protein Eng.*, vol. 12, pp. 107–118, 1999. 10

[27] K. C. Chou, "Prediction of protein structural classes and subcellular locations," *Current Protein Peptide Science*, vol. 1, pp. 171–208, 2000. 10

[28] A. Garg, M Bhasin, and G. P. S. Raghava, "Support Vector Machine-based method for subcellular localization of human proteins using amino acid compositions, their order and similarity search," *J. of Biol. Chem.*, vol. 280, pp. 14427–14432, 2005. 11, 52

[29] K. C. Chou, "Prediction of protein cellular attributes using pseudo amino acid composition," *Proteins: Structure, Function, and Genetics*, vol. 43, pp. 246–255, 2001. 11, 35, 89, 121, 122, 127, 129, 140

[30] K. C. Chou, "Prediction of protein subcellular locations by incorporating quasi-sequence-order effect," *Biochem. Biophys. Res. Commun.*, vol. 278, pp. 477–483, 2000. 11

[31] K. C. Chou, " Using amphiphilic pseudo amino acid composition to predict enzyme subfamily classes," *Bioinformatics*, vol. 21, pp. 10–19, 2005. 11, 35

[32] Y. X. Pan, Z. Z. Zhang, Z. M. Guo, G. Y. Feng, Z. D. Huang, and L. He, "Application of pseudo amino acid composition for predicting protein subcellular location: stochastic signal processing approach," *J. of Protein Chem.*, vol. 22, pp. 395–402, 2003. 11

[33] K. C. Chou and Y. D. Cai, "Predicting subcellular localization of proteins by hybridizing functional domain composition and pseudo-amino acid composition," *Journal of Cellular Biochemistry*, vol. 91, pp. 1197–1203, 2004. 11

[34] F. M. Li and Q. Z. Li, "Predicting protein subcellular location using chou's pseudo amino acid composition and improved hybrid approach," *Protein and Peptide Letters*, vol. 15, pp. 612–616, 2008. 11

[35] J. Y. Shi, S. W. Zhang, Q. Pan, Y. M. Cheng, and J. Xie, "Prediction of protein subcellular localization by support vector machines using multi-scale energy and pseudo amino acid composition," *Amino Acids*, vol. 33, pp. 69–74, 2007. 11

[36] H. Lin, H. Ding, F. B. Guo, A. Y. Zhang, and J. Huang, "Predicting subcellular localization of mycobacterial proteins by using chou's pseudo amino acid composition," *Protein and Peptide Letters*, vol. 15, pp. 739–744, 2008. 11

[37] S. W. Zhang, Y. L. Zhang, H. F. Yang, C. H. Zhao, and Q. Pan, "Using the concept of chou's pseudo amino acid composition to predict protein subcellular localization: an approach by incorporating evolutionary information and von neumann entropies," *Amino Acids*, vol. 34, pp. 565–572, 2008. 11

[38] K. C. Chou and H. B. Shen, "Large-scale predictions of gram-negative bacterial protein subcellular locations," *Journal of Proteome Research*, vol. 5, pp. 3420–3428, 2006. 11, 16, 17

[39] K. C. Chou and Y. D. Cai, "Prediction and classification of protein subcellular location: sequence-order effect and pseudo amino acid composition," *J. of Cell. Biochem.*, vol. 90, pp. 1250–1260, 2003. 11

[40] K. C. Chou and Y. D. Cai, "Prediction of protein subcellular locations by GO-FunD-PseAA predictor," *Biochem. Biophys. Res. Commun.*, vol. 320, pp. 1236–1239, 2004. 11, 14, 15

[41] H. B. Shen and K. C. Chou, "PseAAC: A flexible web-server for generating various kinds of protein pseudo amino acid composition," *Analytical Biochemistry*, vol. 373, pp. 386–388, 2008. 11

[42] K. Nakai, "Protein sorting signals and prediction of subcellular localization," *Advances in Protein Chemistry*, vol. 54, no. 1, pp. 277–344, 2000. 11, 52, 89

[43] L. M. Gierasch, "Signal sequences," *Biochemistry*, vol. 28, pp. 923–930, 1989. 11

[44] G. von Heijine, “The signal peptides,” *Journal of Membrane Biology*, vol. 115, no. 3, pp. 195–201, 1990. 12

[45] G. Schatz and B. Dobberstein, “Common principles of protein translocation across-membranes,” *Science*, vol. 271, no. 5255, pp. 1519–1526, 1996. 12

[46] G. Heijne, J. Steppuhn, and R. G. Herrmann, “Domain structure of mitochondrial and chloroplast targeting peptides,” *European Journal of Biochemistry*, vol. 180, no. 3, pp. 535–545, 1989. 12

[47] O. Emanuelsson, G. von Heijne, and G. Schneider, “Analysis and prediction of mitochondrial targeting peptides,” *Methods in Cell Biology*, vol. 65, pp. 175–187, 2001. 12

[48] K. Nakai and M. Kanehisa, “Expert system for predicting protein localization sites in gram-negative bacteria,” *Proteins: Structure, Function, and Genetics*, vol. 11, no. 2, pp. 95–110, 1991. 12

[49] P. Horton, K. J. Park, T. Obayashi, and K. Nakai, “Protein subcellular localization prediction with WOLF PSORT,” in *Proc. 4th Annual Asia Pacific Bioinformatics Conference (APBC06)*, 2006, pp. 39–48. 12

[50] H. Nielsen, J. Engelbrecht, S. Brunak, and G. von Heijne, “A neural network method for identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites,” *Int. J. Neural Sys.*, vol. 8, pp. 581–599, 1997. 12

[51] H. Nielsen, S. Brunak, and G. von Heijne, “Machine learning approaches for the prediction of signal peptides and other protein sorting signals,” *Protein Eng.*, vol. 12, no. 1, pp. 3–9, 1999. 12

[52] O. Emanuelsson, H. Nielsen, S. Brunak, and G. von Heijne, "Predicting subcellular localization of proteins based on their N-terminal amino acid sequence," *J. Mol. Biol.*, vol. 300, no. 4, pp. 1005–1016, 2000. 12, 18, 52

[53] O. Emanuelsson, H. Nielsen, and G. von Heijne, "Chlorop, a neural network-based method for predicting chloroplast transit peptides and their cleavage sites," *Protein Science*, vol. 8, pp. 978–984, 1999. 12

[54] R. Mott, J. Schultz, P. Bork, and C.P. Ponting, "Predicting protein cellular localization using a domain projection method," *Genome research*, vol. 12, no. 8, pp. 1168–1174, 2002. 13

[55] 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. 13

[56] R. Nair and B. Rost, "Sequence conserved for subcellular localization," *Protein Science*, vol. 11, pp. 2836–2847, 2002. 13

[57] Z. Lu, D. Szafron, R. Greiner, P. Lu, D. S. Wishart, B. Poulin, J. Anvik, C. Macdonell, and R. Eisner, "Predicting subcellular localization of proteins using machine-learned classifiers," *Bioinformatics*, vol. 20, no. 4, pp. 547–556, 2004. 13

[58] J. K. Kim, G. P. S. Raghava, S. Y. Bang, and S. Choi, "Prediction of subcellular localization of proteins using pairwise sequence alignment and support vector machine," *Pattern Recog. Lett.*, vol. 27, no. 9, pp. 996–1001, 2006. 13

[59] M.W. Mak, J. Guo, and S.Y. Kung, "PairProSVM: Protein subcellular localization based on local pairwise profile alignment and SVM," *IEEE/ACM Trans. on Computational Biology and Bioinformatics*, vol. 5, no. 3, pp. 416–422, 2008. 13, 43

[60] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, pp. 3389–3402, 1997. 13, 16, 29, 33, 89, 124, 131

[61] X. Wang and G. Z. Li, "Multilabel learning via random label selection for protein subcellular multilocations prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 10, no. 2, pp. 436–446, 2013. 13

[62] T. Liu, X. Geng, X. Zheng, R. Li, and J.Wang, "Accurate prediction of protein structural class using auto covariance transformation of psi-blast profiles," *Amino Acids*, vol. 42, no. 6, pp. 2243–2249, 2011. 13

[63] S. Wan, M. W. Mak, and S. Y. Kung, "Protein subcellular localization prediction based on profile alignment and Gene Ontology," in *2011 IEEE International Workshop on Machine Learning for Signal Processing (MLSP'11)*, Sept 2011, pp. 1–6. 14, 15, 53

[64] K. C. Chou and H. B. Shen, "Predicting eukaryotic protein subcellular location by fusing optimized evidence-theoretic K-nearest neighbor classifiers," *J. of Proteome Research*, vol. 5, pp. 1888–1897, 2006. 14, 16, 17, 19, 95, 96, 119, 120, 121, 124, 125

[65] S. Wan, M. W. Mak, and S. Y. Kung, "GOASVM: Protein subcellular localization prediction based on gene ontology annotation and SVM," in *2012 IEEE Interna-*

tional Conference on Acoustics, Speech, and Signal Processing (ICASSP'12), 2012, pp. 2229–2232. 14

[66] S. Mei, "Multi-label multi-kernel transfer learning for human protein subcellular localization," *PLoS ONE*, vol. 7, no. 6, pp. e37716, 2012. 14, 34

[67] S. Wan, M. W. Mak, and S. Y. Kung, "Semantic similarity over gene ontology for multi-label protein subcellular localization," *Engineering*, vol. 5, pp. 68–72, 2013. 14, 69

[68] R. Nair and B. Rost, "Sequence conserved for subcellular localization," *Protein Science*, vol. 11, pp. 2836–2847, 2002. 14

[69] Z. Lu, D. Szafron, R. Greiner, P. Lu, D. S. Wishart, B. Poulin, J. Anvik, C. Macdonell, and R. Eisner, "Predicting subcellular localization of proteins using machine-learned classifiers," *Bioinformatics*, vol. 20, no. 4, pp. 547–556, 2004. 14

[70] S. Brady and H. Shatkay, "EpiLoc: a (working) text-based system for predicting protein subcellular location," in *Pac. Symp. Biocomput.*, 2008, pp. 604–615. 14

[71] A. Fyshe, Y. Liu, D. Szafron, R. Greiner, and P. Lu, "Improving subcellular localization prediction using text classification and the gene ontology," *Bioinformatics*, vol. 24, pp. 2512–2517, 2008. 14

[72] 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, pp. 80, 2008. 14, 16, 17, 19, 23, 24, 25, 26, 52, 119, 124

[73] S.-M. Chi and D. Nam, "Wegoloc: accurate prediction of protein subcellular localization using weighted gene ontology terms," *Bioinformatics*, vol. 28, no. 7, pp. 1028–1030, 2012. 14, 19

[74] E. M. Zdobnov and R. Apweiler, "InterProScan – an integration platform for the signature-recognition methods in InterPro," *Bioinformatics*, vol. 17, pp. 847–848, 2001. 15

[75] The InterPro Consortium, "The interpro database, an integrated documentation resource for protein families, ddomain and functional sites," *Nucleic*, vol. 29, pp. 37–40, 2001. 15

[76] N. J. Mulder, R. Apweiler, T. K. Attwood, A. Bairoch, D. Barrell, A. Bateman, D. Binns, M. Biswas, P. Bradley, and P. Bork, "The InterPro Database, 2003 brings increased coverage and new features," *Nucleic Acids Res.*, vol. 31, pp. 315–318, 2003. 15

[77] K. Hofmann, P. Bucher, L. Falquet, and A. Bairoch, "The prosite database, its status in 1999," *Nucleic*, vol. 27, pp. 215–219, 1999. 15

[78] T. K. Attwood, M. D. R. Croning, D. R. Flower, A. P. Lewis, J. E. Mabey, P. Scordis, J. Selley, and W. Wright, "Prints-s: the database formerly known as prints," *Nucleic*, vol. 28, pp. 225–227, 2000. 15

[79] A. Bateman, E. Birney, R. Durbin, S.R. Eddy, K. L. Howe, and E. L. Sonnhammer, "The pfam protein families database," *Nucleic*, vol. 28, pp. 263–266, 2000. 15

[80] F. Corpet, F. Servant, J. Gouzy, and D. Kahn, "Prodom and prodom-cg: tools for protein domain analysis and whole genome comparisons," *Nucleic*, vol. 28, pp. 267–269, 2000. 15

[81] J. Schultz, R. R. Copley, T. Doerks, C.P. Ponting, and P. Bork, "Smart: A web-based tool for the study of genetically mobile domains," *Nucleic Acids Research*, vol. 28, pp. 231–234, 2000. 15

[82] T. Blum, S. Briesemeister, and O. Kohlbacher, "MultiLoc2: Integrating phylogeny and Gene Ontology terms improves subcellular protein localization prediction," *BMC Bioinformatics*, vol. 10, pp. 274, 2009. 15

[83] S. Y. Mei, W. Fei, and S. G. Zhou, "Gene ontology based transfer learning for protein subcellular localization," *BMC Bioinformatics*, vol. 12, pp. 44, 2011. 15, 29, 114

[84] T. Q. Tung and D. Lee, "A method to improve protein subcellular localization prediction by integrating various biological data sources," *BMC Bioinformatics*, vol. 10 (Suppl 1), pp. S43, 2009. 15

[85] K. C. Chou and H. B. Shen, "Hum-PLoc: A novel ensemble classifier for predicting human protein subcellular localization," *Biochem Biophys Res Commun*, vol. 347, pp. 150–157, 2006. 16, 17, 22, 95, 119, 124

[86] 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, pp. 1728–1734, 2007. 16, 29

[87] Z. C. Wu, X. Xiao, and K. C. Chou, "iLoc-Plant: A multi-label classifier for predicting the subcellular localization of plant proteins with both single and multiple sites," *Molecular BioSystems*, vol. 7, pp. 3287–3297, 2011. 16, 52, 87, 102, 103, 115, 119, 132, 142, 160, 179, 180, 187, 188, 196

[88] K. C. Chou, Z. C. Wu, and X. Xiao, "iLoc-Hum: using the accumulation-label scale to predict subcellular locations of human proteins with both single and multiple sites," *Molecular BioSystems*, vol. 8, pp. 629–641, 2012. 16, 22, 119

[89] 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 & Peptide Letters*, vol. 19, pp. 4–14, 2012. 16, 119

[90] 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, pp. e18258, 2011. 16, 22, 52, 87, 102, 119, 177, 196

[91] X. Xiao, Z. C. Wu, and K. C. Chou, "A multi-label learning classifier for predicting the subcellular localization of gram-negative bacterial proteins with both single and multiple sites," *PLoS ONE*, vol. 6, no. 6, pp. e20592, 2011. 16, 119

[92] K. C. Chou and H. B. Shen, "Cell-PLoc: A package of web-servers for predicting subcellular localization of proteins in various organisms," *Nature Protocols*, vol. 3, pp. 153–162, 2008. 16, 30

[93] 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, pp. 42–51, 2011. 16, 51, 52, 81, 88, 102, 108, 109, 110, 113, 119, 131, 132, 179, 187, 188, 196

[94] K. C. Chou and H. B. Shen, "Cell-PLoc 2.0: An improved package of web-servers for predicting subcellular localization of proteins in various organisms," *Nat. Sci.*, vol. 2, pp. 1090–1103, 2010. 16, 17

[95] H. B. Shen and K.C. Chou, "Gpos-PLoc: An ensemble classifier for predicting subcellular localization of Gram-positive bacterial proteins," *Protein Engineering, Design and Selection*, vol. 20, pp. 39–46, 2007. 17

[96] K. C. Chou and H. B. Shen, "Plant-mPLoc: A top-down strategy to augment the power for predicting plant protein subcellular localization," *PLoS ONE*, vol. 5, pp. e11335, 2010. 17, 52, 53, 87, 102, 103, 132, 142, 143, 160, 179, 180, 187, 188, 196

[97] W. L. Huang, C. W. Tung, S. W. Ho, S. F. Hwang, and S. Y. Ho, "Predicting protein subnuclear localization using GO-amino-acid composition features," *Biosystems*, vol. 98, no. 2, pp. 73–79, 2009. 17

[98] Z. Lei and Y. Dai, "Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction," *BMC Bioinformatics*, vol. 7, pp. 491, 2006. 17, 68, 190

[99] K. Y. Lee, H. Y. Chuang, A. Beyer, M. K. Sung, W. K. Huh, B. Lee, and T. Ideker, "Protein networks markedly improve prediction of subcellular localization in multiple eukaryotic species," *Nucleic Acids Research*, vol. 36, no. 20, pp. e136, 2008. 17

[100] O. Emanuelsson, S. Brunak, G. von Heijne, and H. Nielsen, "Locating proteins in the cell using TargetP, SignalP, and related tools," *Nature Protocols*, vol. 2, no. 4, pp. 953–971, 2007. 18

[101] W. Wang, M. W. Mak, and S. Y. Kung, "Speeding up subcellular localization by extracting informative regions of protein sequences for profile alignment," in *Proc. Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'10)*, 2010, pp. 147–154. 18

[102] 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. 19, 29, 89, 169

[103] P. W. Lord, R. D. Stevens, A. Brass, and C. A. Goble, "Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation ," *Bioinformatics*, vol. 19, no. 10, pp. 1275–1283, 2003. 23, 25, 68

[104] D. Binns, E. Dimmer, R. Huntley, D. Barrell, C. O'Donovan, and R. Apweiler, "QuickGO: a web-based tool for Gene Ontology searching," *Bioinformatics*, vol. 25, no. 22, pp. 3045–3046, 2009. 23, 26

[105] Z. Lu and L. Hunter, "GO molecular function terms are predictive of subcellular localization," in *In Proc. of Pac. Symp. Biocomput. (PSB'05)*, 2005, pp. 151–161. 28

[106] A. Hoglund, P. Donnes, T. Blum, H. 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. 29

[107] A. Pierleoni, P. Luigi, P. Fariselli, and R. Casadio, "Bacello: a balanced subcellular localization predictor," *Bioinformatics*, vol. 22, no. 14, pp. e408–e416, 2006. 29

[108] 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, pp. 290, 2012. 29, 87, 89, 108, 109, 110, 113, 158, 159, 161, 162, 168, 173, 174, 177, 179

[109] S. Briesemeister, T. Blum, S. Brady, Y. Lam, O. Kohlbacher, and H. Shatkay, "SherLoc2: A high-accuracy hybrid method for predicting subcellular localization of proteins," *Journal of Proteome Research*, vol. 8, pp. 5363–5366, 2009. 30, 124

[110] K. C. Chou, "Some remarks on predicting multi-label attributes in molecular biosystems," *Molecular BioSystems*, vol. 9, pp. 1092–1100, 2013. 30

[111] X. Wang and G. Z. Li, "A multi-label predictor for identifying the subcellular locations of singleplex and multiplex eukaryotic proteins," *PLoS ONE*, vol. 7, no. 5, pp. e36317, 2012. 30

[112] K. C. Chou, " Some remarks on protein attribute prediction and pseudo amino acid composition (50th Anniversary Year Review)," *Journal of Theoretical Biology*, vol. 273, pp. 236–247, 2011. 31, 35, 95

[113] Y. Hu, T. Li, J. Sun, S. Tang, W. Xiong, D. Li, G. Chen, and P. Cong, "Predicting Gram-positive bacterial protein subcellular localization based on localization motifs," *Journal of Theoretical Biology*, vol. 308, pp. 135–140, 2012. 31

[114] E. Camon, M. Magrane, D. Barrel, D. Binns, W. Fleischnann, P. Kersey, N. Mulder, T. Oinn, J. Maslen, and A. Cox, "The gene ontology annotation (GOA) project: Implementation of GO in SWISS-PROT, TrEMBL and InterPro," *Genome Res.*, vol. 13, pp. 662–672, 2003. 32

[115] R. Apweiler, A. Bairoch, C. H. Wu, W. C. Barker, B. Boeckmann, S. Ferro, E. Gasteiger, H. Huang, R. Lopez, M. Magrane, M. J. Martin, D. A. Ntale, C. O'Donovan, N. Redaschi, and L. S. Yeh, "UniProt: the Universal Protein knowledgebase," *Nucleic Acids Res*, vol. 32, pp. D115–D119, 2004. 32

[116] B. Boeckmann, A. Bairoch, R. Apweiler, M. C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider, "The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003," *Nucleic Acids Res.*, vol. 31, pp. 365–370, 2003. 32

[117] C. H. Wu, H. Huang, L. S. Yeh, and W. C. Barker, "Protein family classification and functional annotation," *Comput. Biol. Chem.*, vol. 27, pp. 37–47, 2003. 32

[118] N. J. Mulder and R. Apweiler, "The InterPro database and tools for protein domain analysis," *Current Protocols in Bioinformatics*, vol. 2, no. 7, pp. 1–18, 2008. 32

[119] D. Barrel, E. Dimmer, R. P. Huntley, D. Binns, C O'Donovan, and R. Apweiler, "The GOA database in 2009—an integrated Gene Ontology Annotation resource," *Nucl. Acids Res.*, vol. 37, pp. D396–D403, 2009. 34

[120] V. N. Vapnik, "The nature of statistical learning theory," in *Springer Verlag*, 2000. 37

[121] H. Rangwala and G. Karypis, "Profile-based direct kernels for remote homology detection and fold recognition," *Bioinformatics*, vol. 21, no. 23, pp. 4239–4247, 2005. 43

[122] S.F. Altschul, T.L. Madden, A.A. Schafer, J. Zhang, Z. Zhang, W. Miller, and D.J. Lipman, "Gapped BLAST and PSI-BLAST: A new generation of protein database serarch programs," *Nucleic Acids Res.*, vol. 25, pp. 3389–3402, 1997. 43

[123] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, no. 1, pp. 42–54, 2000. 45

[124] A. H. Millar, C. Carrie, B. Pogson, and J. Whelan, "Exploring the function-location nexus: using multiple lines of evidence in defining the subcellular location of plant proteins," *Plant Cell*, vol. 21, no. 6, pp. 1625–1631, 2009. 48

[125] R. F. Murphy, "Communicating subcellular distributions," *Cytometry*, vol. 77, no. 7, pp. 686–92, 2010. 48

[126] L. J. Foster, C. L. De Hoog, Y. Zhang, Y. Zhang, X. Xie, V. K. Mootha, and M. Mann, "A mammalian organelle map by protein correlation profiling," *Cell*, vol. 125, pp. 187–199, 2006. 48

[127] S. Zhang, X. F. Xia, J. C. Shen, Y. Zhou, and Z.R. Sun, "DBMLoc: A database of proteins with multiple subcellular localizations," *BMC Bioinformatics*, vol. 9, pp. 127, 2008. 48

[128] J. C. Mueller, C. Andreoli, H. Prokisch, and T. Meitinger, "Mechanisms for multiple intracellular localization of human mitochondrial proteins," *Mitochondrion*, vol. 3, pp. 315–325, 2004. 48

[129] R. Russell, R. Bergeron, G. Shulman, and H.: Young, "Translocation of myocardial glut-4 and increased glucose uptake through activation of ampk by aicar," *American Journal of Physiology*, vol. 277, pp. H643–649, 1997. 48

[130] S. Rea and D. James, "moving glut4: the biogenesis and trafficking of glut4 storage vesicles," *Diabetes*, vol. 46, pp. 1667–1677, 1997. 48

[131] M. L. Zhang and Z. H. Zhou, " A k-nearest neighbor based algorithm for multi-label classification," in *IEEE International Conference on Granular Computing*, 2005, pp. 718–721. 48

[132] C. Vens, J. Struyf, L. Schietgat, S. Dzeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Machine Learning*, vol. 2, no. 73, pp. 185–214, 2008. 48, 49

[133] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya, "Hierarchical multi-label prediction of gene function," *Bioinformatics*, vol. 22, no. 7, pp. 830–836, 2006. 48

[134] R. E. Schapire and Y. Singer, "Boostexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000. 48, 49, 50

[135] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor, "Kernel-based learning of hierarchical multilabel classification methods," *Journal of Machine Learning Research*, vol. 7, pp. 1601–1626, 2006. 48

[136] I. Katakis, G. Tsoumakas, and I. Vlahavas, " Multilabel text classification for automated tag suggestion," in *Proceedings of the ECML/PKDD 2008 Discovery Challenge*, 2008. 48

[137] R. Moskovitch, S. Cohenkashi, U. Dror, I. Levy, A. Maimon, and Y. Shahar, “Multiple hierarchical classification of free-text clinical guidelines,” *Artificial Intelligence in Medicine*, vol. 37, pp. 177–190, 2006. 48

[138] N. Ghamrawi and A. McCallum, “Collective multi-label classification,” in *Proceedings of the 2005 ACM Conference on Information and Knowledge Management (CIKM'05)*, 2005, pp. 195–200. 48

[139] T. Li and M. Ogihara, “Toward intelligent music information retrieval,” *IEEE Transactions on Multimedia*, vol. 8, no. 3, pp. 564–574, 2006. 48

[140] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, “Multilabel classification of music into emotions,” in *Proceedings of the 9th International Conference on Music Information Retrieval*, 2006, pp. 325–330. 48

[141] C. G. M. Snoek, M. Worring, J. C. van Gemert, J. M. Geusebroek, and A. W. M. S-meulders, “ The challenge problem for automated detection of 101 semantic concepts in multimedia,” in *Proceedings of the 14th annual ACM International Conference on Multimedia*, 2006, pp. 421–430. 48

[142] M. Boutell, J. Luo, X. Shen, and C. Brown, “Learning multi-label scene classification,” *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004. 48, 49

[143] A. Clare and R. D. King, “Knowledge discovery in multi-label phenotype data,” in *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, 2001, pp. 42–53. 49

[144] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook, O. Maimon, l. Rokach (Ed.). Springer, 2nd edition*, 2010, pp. 667–685. 49, 108

[145] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, pp. 1–13, 2007. 49, 190

[146] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2009, pp. 254–269. 49, 50

[147] D. Hsu, S. M. Kakade, J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 772–780. 49

[148] V. N. Vapnik, "Statistical learning theory," in *John Wiley and Sons*, 1998. 50

[149] A. Elisseeff and J. Weston, "Kernel methods for multi-labelled classification and categorical regression problems.," in *In Advances in Neural Information Processing Systems 14*. 2001, pp. 681–687, MIT Press. 50

[150] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2004, pp. 22–30, Springer. 50

[151] T. G. Dietterich and g. Bakari, "Solving multiclass learning problem via error-correcting output codes," *Journal of Artificial Intelligence Research*, pp. 263–286, 1995. 51

[152] U. Kressel, “Pairwise classification and support vector machines,” in *Advances in Kernel Methods: Support Vector Learning, Chap. 15. MIT Press*, 1999. 51

[153] B. Scholkopf and A. J. Smola, “Learning with kernels,” in *MIT Press*, 2002. 51

[154] H. B. Shen and K. C. Chou, “Virus-mPLoc: A fusion classifier for viral protein subcellular location prediction by incorporating multiple sites,” *J. Biomol. Struct. Dyn.*, vol. 26, pp. 175–186, 2010. 51, 53, 102, 131, 132, 179, 187, 188, 196

[155] H. B. Shen and K. C. Chou, “Virus-PLoc: A fusion classifier for predicting the subcellular localization of viral proteins within host and virus-infected cells,” *Biopolymers*, vol. 85, pp. 233–240, 2006. 51

[156] L. Q. Li, Y. Zhang, L. Y. Zou, Y. Zhou, and X. Q. Zheng, “Prediction of protein subcellular multi-localization based on the general form of Chou’s pseudo amino acid composition,” *Protein and Peptide Letters*, vol. 19, pp. 375–387, 2012. 52, 102, 131, 132, 179

[157] I. Small, N. Peeters, F. Legeai, and C. Lurin, “Predotar: A tool for rapidly screening proteomes for N-terminal targeting sequences,” *Proteomics*, vol. 4, pp. 1581–1590, 2004. 52

[158] K. C. Chou and H. B. Shen, “A new method for predicting the subcellular localization of eukaryotic proteins with both single and multiple site: Euk-mPLoc 2.0,” *PLoS ONE*, vol. 5, pp. e9931, 2010. 52, 87, 102, 176, 177, 196

[159] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*, Wiley, second edition, 2000. 61

[160] J. Zhu and T. Hastie, "Kernel logistic regression and the import vector machine," in *Journal of Computational and Graphical Statistics*. 2001, pp. 1081–1088, MIT Press. 61

[161] J. Zhu, "Classification of gene microarrays by penalized logistic regression," *Biostatistics*, vol. 5, no. 3, pp. 427–443, 2004. 61

[162] S. K. Shevade and S. S. Keerthi, "A simple and efficient algorithm for gen selection using sparse logistic regression," *Bioinformatics*, vol. 19, no. 17, pp. 2246–2253, 2003. 61

[163] M. Winston, R. Chaffin, and D. Herrmann, "A taxonomy of part-whole relations," *Cognitive Science*, vol. 11, pp. 417–444, 1987. 68

[164] J. Odell, "Six different kinds of aggression," in *Advanced object-oriented analysis and design using UML*, pp. 139–149. Cambridge University Press, 1998. 68

[165] The Gene Ontology Consortium, "The Gene Ontology: enhancements for 2011," *Nucleic Acids Res.*, vol. 40, pp. D559–D564, 2012. 68, 203

[166] The Gene Ontology Consortium, "The Gene Ontology in 2010: extensions and refinements," *Nucleic Acids Res.*, vol. 38, pp. D331–D335, 2010. 68

[167] M. Zhu, L. Gao, Z. Guo, Y. Li, D. Wang, J. Wang, and C. Wang, "Globally predicting protein functions based on co-expressed protein-protein interaction networks and ontology taxonomy similarities," *Gene*, vol. 391, no. 1-2, pp. 113–119, 2007. 68

[168] C. Pesquita, D. Faria, A. O. Falcao, P. Lord, and F. M. Counto, "Metrics for GO based protein semantic similarity: a systematic evaluation," *BMC Bioinformatics*, vol. 9, no. Suppl 5, pp. S4, 2008. 68

[169] X. Wu, L. Zhu, J. Guo, D. Y. Zhang, and K. Lin, "Prediction of yeast protein-protein interaction network: insights from the gene ontology and annotations," *Nucleic Acids Res.*, vol. 34, pp. 2137–2150, 2006. 68

[170] X. Guo, R. Liu, C. D. Shriver, H. Hu, and M. N. Liebman, "Assessing semantic similarity measures for the characterization of human regulatory pathways," *Bioinformatics*, vol. 22, pp. 967–973, 2006. 68

[171] T. Xu, L. Du, and Y. Zhou, "Evaluation of GO-based functional similarity measures using S. cerevisiae protein interaction and expression profile data," *BMC Bioinformatics*, vol. 9, pp. 472, 2008. 68

[172] D. Yang, Y. Li, H. Xiao, Q. Liu, M. Zhang, J. Zhu, W. Ma, C. Yao, J. Wang, D. Wang, Z. Guo, and B. Yang, "Gaining confidence in biological interpretation of the microarray data: the functional consistence of the significant GO categories," *Bioinformatics*, vol. 24, no. 2, pp. 265–271, 2008. 68, 71, 73, 190

[173] P. Resnik, "Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language," *Journal of Artificial Intelligence Research*, vol. 11, pp. 95–130, 1999. 68, 70, 72

[174] D. Lin, "An information-theoretic definition of similarity," in *Proceedings of the 15th International Conference on Machine Learning*, 1998, pp. 296–304. 68, 70, 73, 156, 190

[175] A. Schlicker, F. S. Domingues, J. Rahnenfuhrer, and T. Lengauer, "A new measure for functional similarity of gene products based on Gene Ontology," *BMC Bioinformatics*, vol. 7, pp. 302, 2006. 68, 73, 156, 190

[176] F. M. Couto, M. J. Silva, and P. M. Coutinho, "Semantic similarity over the gene ontology: Family correlation and selecting disjunctive ancestors," in *Proceedings of 14-th International ACM Conference in Information and Knowledge Management*, 2005, pp. 343–344. 68

[177] O. Bodenreider, M. Aubry, and A. Burgun, "Non-lexical approaches to identifying associative relations in the gene ontology," in *Pac. Symp. Biocomput.*, 2005, pp. 91–102. 68

[178] A. D. Pozo, F. Pazos, and A. Valencia, "Defining functional distances over gene ontology," *BMC Bioinformatics*, vol. 9, pp. 50, 2008. 68

[179] H. Wu, Z. Su, F. Mao, V. Olman, and Y. Xu, "Prediction of functional modules based on comparative genome analysis and gene ontology application," *Nucleic Acids Res.*, vol. 33, pp. 2822–2837, 2005. 68

[180] J. Cheng, M. Cline, J. Martin, D. Finkelstein, T. Awad, and et al., "A knowledge-based clustering algorithm driven by gene ontology," *Journal of Biopharmaceutical Statistics*, vol. 14, pp. 687–700, 2004. 68

[181] H. Yu, L. Gao, K. Tu, and Z. Guo, "Broadly predicting specific gene function with expression similarity and taxonomy similarity," *Gene*, vol. 352, pp. 75–81, 2005. 68

[182] J. L. Sevilla, V. Segura, A. Podhorski, E. Guruceaga, J. M. Mato, L. A. Martinez-Cruz, F. J. Corrales, and A. Rubio, "Correlation between gene expression and GO semantic similarity," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 2, no. 4, pp. 330–338, 2005. 69

[183] Y. Tao, L. Sam, J. Li, C. Friedman, and Y. A. Lussier, "Information theory applied to the sparse gene ontology annotation network to predict novel gene function," *Bioinformatics*, vol. 23, no. 13, pp. i529–i538, 2007. 69

[184] J. Z. Wang, Z. Du, R. Payattakool, P. S. Yu, and C. F. Chen, "A new method to measure the semantic similarity of GO terms," *Bioinformatics*, vol. 23, no. 10, pp. 1274–1281, 2007. 69

[185] R. M. Riensche, B. L. Baddeley, A. P. Sanfilippo, C. Posse, and B. Gopalan, "XOA: Web-enabled cross-ontological analytics," in *2007 IEEE Congress on Services*, 2007, pp. 99–105. 69

[186] D. W. Huang, B. T. Sherman, Q. Tan, J. R. Collins, W. G. Alvord, J. Roayaei, R. Stephens, M. W. Baseler, H. C. Lane, and R. A. Lempicki, "The DAVID Gene Functional Classification Tool: a novel biological module-centric algorithm to functionally analyze large gene lists," *Genome Biology*, vol. 8, no. R183, 2007. 69

[187] J. Chabalier, J. Mosser, and A. Burgun, "A trasversal approach to predict gene product networks from ontology-based similarity," *BMC Bioinformatics*, vol. 8, pp. 235, 2007. 69

[188] M. Mistry and P. Pavlidis, "Gene ontology term overlap as a measure of gene functional similarity," *BMC Bioinformatics*, vol. 9, pp. 327, 2008. 69

[189] B. Sheehan, A. Quigley, B. Gaudin, and S. Dobson, "A relation based measure of semantic similarity for Gene Ontology annotations," *BMC Bioinformatics*, vol. 9, pp. 468, 2008. 69

[190] C. Pesquita, D. Faria, A. O. Falcao, P. Lord, and F. M. Counto, "Semantic similarity in biomedical ontologies," *PLoS Computational Biology*, vol. 5, no. 7, pp. e1000443, 2009. 69

[191] P. H. Guzzi, M. Mina, C. Guerra, and M. Cannataro, "Semantic similarity analysis of protein data: assessment with biological features and issues," *Briefings in Bioinformatics*, vol. 13, no. 5, pp. 569–585, Sep 2012. 69

[192] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," in *Proceedings of International Conference Research on Computational Linguistics (ROCLING X)*, 1997, pp. 19–33. 73, 156, 190

[193] R. Nair and B. Rost, "Protein subcellular localization prediction using artificial intelligence technology," in *Functional Proteomics*, pp. 435–463. Springer, 2008. 76

[194] A. Adelfio, V. Volpato, and G. Pollastri, "SCLpredT: Ab initio and homology-based prediction of subcellular localization by N-to-1 neural networks," *SpringerPlus*, vol. 2, no. 1, pp. 1–11, 2013. 76

[195] R. Lotlikar and R. Kothari, "Adaptive linear dimensionality reduction for classification," *Pattern Recognition*, vol. 33, no. 2, pp. 185–194, 2000. 80

[196] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *The Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003. 80

[197] D. L. Swets and J. J. Weng, "Efficient content-based image retrieval using automatic feature selection," in *Proceedings of International Symposium on Computer Vision*. IEEE, 1995, pp. 85–90. 80

[198] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al., "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999. 80

[199] E. Bingham and H. Mannila, "Random projection in dimension reduction: Applications to image and text data," in *the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*, 2001, pp. 245–250. 80, 82

[200] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006. 80, 193, 194

[201] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent semantic indexing: A probabilistic analysis," in *Proceedings of the 17th ACM Symposium on the Principles of Database Systems*, 1998, pp. 159–168. 80

[202] M. Kurimo, "Indexing audio documents by using latent semantic analysis and som," in *Kohonen Maps*. 1999, pp. 363–374, Elsevier. 80

[203] S. Dasgupta, "Learning mixtures of Gaussians," in *40th Annual IEEE Symposium on Foundations of Computer Science,*, 1999, pp. 634–644. 80

[204] B. Yang, D. Hartung, K. Simoens, and C. Busch, "Dynamic random projection for biometric template protection," in *2010 Fourth IEEE International Conference on Biometrics: Theory Applications and Systems (BTAS)*, 2010, pp. 1–7. 80

[205] Y. Wang and K. N. Plataniotis, "An analysis of random projection for changeable and privacy-preserving biometric verification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 5, pp. 1280–1293, 2010. 80

[206] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," in *Conference in Modern Analysis and Probability*, 1984, pp. 599–608. 81

[207] P. Frankl and H. Maehara., "The Johnson-Lindenstrauss lemma and the sphericity of some graphs," *Journal of Combinatorial Theory, Series B*, vol. 44, pp. 355–362, 1988. 82

[208] D. Achlioptas, "Database-friendly random projections: Johnson-Lindenstrauss with binary coins," *Journal of Computer and Systems Sciences*, vol. 66, pp. 671–687, 2003. 82

[209] S. Wan, M. W. Mak, and S. Y. Kung, "HybridGO-Loc: Mining hybrid features on gene ontology for predicting subcellular localization of multi-location proteins," *PLoS ONE*, vol. 9, no. 3, pp. e89545, 2014. 87

[210] 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, pp. e37155, 2011. 87

[211] S. Wan, M. W. Mak, B. Zhang, Y. Wang, and S. Y. Kung, "An ensemble classifier with random projection for predicting multi-label protein subcellular localization," in *2013 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2013, pp. 35–42. 87

[212] L. Q. Li, Y. Zhang, L. Y. Zou, C. Q. Li, B. Yu, X. Q. 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, pp. e31057, 2012. 87

[213] G. R. Pasha and M. A. A. Shah, "Application of ridge regression to multicollinear data," *Journal of Research (Science)*, vol. 15, no. 1, pp. 97–106, 2004. 90

[214] A. Hadgu, "An application of ridge regression analysis in the study of syphilis data," *Statistics in Medicine*, vol. 3, no. 3, pp. 293–299, 1984. 90

[215] D. W. Marquardt and R. D. Snee, "Ridge regression in practice," *The American Statistician*, vol. 29, no. 1, pp. 3–20, 1975. 90

[216] G. L. Wang, Jr. Dunbrack, and R. L. PISCES, "A protein sequence culling server," *Bioinformatics*, vol. 19, pp. 1589–1591, 2003. 96

[217] Y. Huang and Y. D. Li, "Prediction of protein subcellular locations using fuzzy K-NN method," *Bioinformatics*, vol. 20, no. 1, pp. 21–28, 2004. 98

[218] M. W. Mak, J. Guo, and S. Y. Kung, "PairProSVM: Protein subcellular localization based on local pairwise profile alignment and SVM," *IEEE/ACM Trans. on Computational Biology and Bioinformatics*, vol. 5, no. 3, pp. 416 – 422, 2008. 100, 122, 129

[219] B.W. Matthews, "Comparison of predicted and observed secondary structure of t4 phage lysozyme," *Biochem. Biophys. Acta*, vol. 405, pp. 442–451, 1975. 100

[220] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hullermeier, “On label dependence and loss minimization in multi-label classification,” *Machine Learning*, vol. 88, no. 1-2, pp. 5–45, 2012. 111, 112

[221] Wei Gao and Zhi-Hua Zhou, “On the consistency of multi-label learning,” *Artificial Intelligence*, vol. 199-200, pp. 22–44, 2013. 111, 112

[222] K. C. Chou and C. T. Zhang, “Review: Prediction of protein structural classes,” *Critical Reviews in Biochemistry and Molecular Biology*, vol. 30, no. 4, pp. 275–349, 1995. 114, 115

[223] T. Hastie, R. Tibshirani, and J. Friedman, *The element of statistical learning*, Springer-Verlag, 2001. 115, 192

[224] H. Abdi and L. J. Williams, “Jackknife,” in *Encyclopedia of Research Design*, pp. 655–660. Sage, 2010. 115

[225] Q. McNemar, “Note on the sampling error of the difference between correlated proportions or percentages,” *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947. 131

[226] L. Gillick and S. J. Cox, “Some statistical issues in the comparison of speech recognition algorithms,” in *1989 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP’89)*. IEEE, 1989, pp. 532–535. 131

[227] M. Bhasin and G. P. S. Raghava, “ESLpred: SVM based method for subcellular localization of eukaryotic proteins using dipeptide composition and PSI-BLAST,” *Nucleic Acids Res.*, vol. 32, no. Webserver Issue, pp. 414–419, 2004. 133

[228] Y. Bengio, O. Delalleau, and N. L. Roux, “The curse of dimensionality for local kernel machines ,” Tech. Rep., Universite de Montreal, 2005. 134

[229] S. Wan, M. W. Mak, and S. Y. Kung, "Adaptive thresholding for multi-label SVM classification with application to protein subcellular localization prediction," in *2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13)*, 2013, pp. 3547–3551. 161, 162

[230] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, pp. 389–422, 2002. 163

[231] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998. 187

[232] S. Y. Kung and M. W. Mak, "On consistent fusion of multimodal biometrics," in *2006 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'06)*. IEEE, 2006, vol. 5, pp. V1085–V1088. 187

[233] C. Pesquita, D. Pessoa, D. Faria, and F. Couto, "CESSM: Collaborative evaluation of semantic similarity measures," *JB2009: Challenges in Bioinformatics*, vol. 157, 2009. 190

[234] D. Faria, C. Pesquita, F. M. Couto, and A. Falcão, "ProteInOn: A web tool for protein semantic similarity," 2007. 190

[235] G. Yu, F. Li, Y. Qin, X. Bo, Y. Wu, and S. Wang, "GOSemSim: An R package for measuring semantic similarity among GO terms and gene products," *Bioinformatics*, vol. 26, no. 7, pp. 976–978, 2010. 190