

PROTEIN SEQUENCE CLASSIFICATION WITH
BAYESIAN SUPERVISED AND SEMI-SUPERVISED
LEARNED CLASSIFIERS

by

Brian R. King

A Dissertation

Submitted to the University at Albany, State University of New York

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

Dissertation Advisor

Chittibabu Guda, Ph. D.

College of Computing and Information

Department of Computer Science

2008

Protein Sequence Classification with
Bayesian Supervised and Semi-Supervised
Learned Classifiers

by

Brian R. King

COPYRIGHT 2008

Keywords: protein sequence classification, supervised learning, semi-supervised learning, naïve Bayes, Bayesian classification, Expectation-Maximization

Table of Contents

Acknowledgements.....	vii
Abstract.....	x
Tables.....	xii
Figures.....	xiv
1 Introduction.....	1
1.1 Introduction of research area	3
1.2 Motivation.....	4
1.3 Structure of thesis	10
2 Essential Background.....	11
2.1 Machine learning and data mining.....	11
2.1.1 Five steps for classification.....	12
2.1.2 Random variables for probabilistic classification.....	15
2.1.3 Discriminative vs. generative classification	16
2.1.4 Bayesian classification.....	17
2.1.5 Objective function for Bayesian classification	19
2.1.6 Parameter estimation.....	19
2.1.7 Bayesian classifier as a mixture model.....	21
2.2 Molecular biology	23
2.2.1 Protein representation	23
2.2.2 Subcellular localization as a means to infer function	25
3 Literature Review.....	27
3.1 Supervised methods	28
3.1.1 Advanced sequence-based feature spaces.....	33
3.1.2 Non sequence-based feature spaces	35
3.1.3 Methods that predict multi-localization	37
3.1.4 Methods that predict suborganelle granularity	38
3.1.5 Estimation of the subcellular proteome	39
3.2 Semi-supervised learning methods	40
3.3 Conclusion	43
4 ngLOC – Supervised Learning for eukaryotic proteome estimation	45
4.1 The ngLOC method	45
4.1.1 The n-gram model for protein representation	46
4.1.2 Bayesian classification for subcellular localization prediction.....	47
4.1.3 Estimating model parameters.....	52
4.1.4 Probabilistic confidence score	53
4.2 ngLOC-X: Application of ngLOC to proteome-wide predictions.....	55
4.3 ngLOC data.....	57
4.3.1 Proteome datasets.....	58
4.4 Performance measurements	59
4.5 Results.....	61
4.5.1 Evaluation of different size n-grams.....	61
4.5.2 Prediction performance using a 7-gram model	62
4.5.3 Analysis of 7-gram distribution	66
4.5.4 Evaluation of the confidence score	69

4.5.5	Evaluation of the multi-localized confidence score (MLCS)	71
4.5.6	Comparison against other methods	72
4.5.7	Evaluating ngLOC-X for proteome-wide predictions	76
4.5.8	Estimation of subcellular proteoms of eight eukaryotic species.....	81
4.5.9	Biological significance of discriminatory <i>n</i> -grams.....	86
4.6	Discussion	88
4.6.1	ngLOC method development.....	88
4.6.2	Comparison with other methods	89
4.6.3	Estimation of subcellular proteoms	90
4.6.4	Discriminatory <i>n</i> -grams and functional domains.....	91
4.7	Conclusions.....	92
5	ngLOC-bact – Supervised learning for prokaryotic proteome estimation.....	94
5.1	Background	95
5.2	The ngLOC-bact method	98
5.2.1	ngLOC-bact dataset	98
5.2.2	Proteome datasets.....	99
5.2.3	ngLOC-bact – The ngLOC method for bacterial localization prediction	100
5.3	Results.....	100
5.3.1	Performance metrics in the midst of unbalanced data	101
5.3.2	The Receiver Operating Characteristic (ROC) Curve	101
5.3.3	Evaluation of different size <i>n</i> -grams.....	103
5.3.4	Class-wise analysis of the 6-gram model.....	107
5.3.5	Comparison with other methods	110
5.3.6	Generation of proteome-wide predictions	112
5.4	Discussion	117
5.5	Conclusions.....	120
6	Semi-Supervised Learning for Classification of Protein Sequence Data	122
6.1	Introduction.....	122
6.2	Semi-supervised protein classification through EM	125
6.2.1	ngLOC - The Bayesian supervised model	125
6.2.2	Parameter estimation.....	128
6.2.3	Incorporating unlabeled data through EM	130
6.2.4	Extending the EM algorithm.....	133
	Weighted EM using EM- λ	134
	Multi-EM – Multiple Mixture Components per Class.....	134
	EM-CS – Improved Selection of Unlabeled Data.....	137
6.3	Transductive SVMs as a comparative method.....	139
6.4	Results.....	141
6.4.1	Datasets	142
6.4.2	Performance measures	144
6.4.3	Empirical results	146
	Basic-EM Results.....	147
	EM- λ Results	149
	Multi-EM Results.....	152
	EM-CS Results.....	153
6.4.4	Comparison against Transductive SVM	156

6.5	Discussion	159
6.5.1	Comparison Against Text Classification Methods	160
6.5.2	Complexity of naïve Bayes and EM	162
6.5.3	Considerations of EM Enhancements	163
6.5.4	Comparison against transductive SVM	166
6.6	Conclusion	169
7	Conclusions	171
7.1	Future directions	171
7.1.1	Predictive vs. explanatory reasoning	172
7.1.2	Subsequences of biological or biomedical interest	172
7.1.3	Semi-supervised learning	173
7.1.4	Ensemble classification methods	174
7.1.5	Sample selection bias	174
7.1.6	Learning with multiple class labels	178
7.1.7	Prediction of suborganellar locations	178
7.1.8	Natural amino acid groupings	179
7.1.9	Substitution matrices	180
	Appendix	183
	References	197
	Index	207

Acknowledgements

To Mom and Dad, Barry and Heather – Thank you for your endless support and encouragement when I decided to take the risk to go back to school full time. You kept me smiling and persevering through it all.

Dabrina Dutcher – You helped me understand the challenges a PhD student would face right from the beginning. Nobody would have ever guessed that our lives would have turned in the direction they did this year. The countless struggles and challenges that we have had are struggles that no new couple should ever have to endure. Your perseverance and strength in the midst of life-threatening situations continue to amaze and inspire me, as well as everyone that comes across your path. Thank you for continually adapting to seemingly unending life changes and challenges together. (Let's never have a 2008 again, okay?) I look forward to our future together with baby Dylan! (Time to buy another 57 books on parenting!) I'm thrilled to be on this adventure with you. I love you dearly. (PS – Yes, the espresso machine was a big winner! And no, I am not an addict.)

Anne Dantz – Thank you for your amazing editing skills, and for your patience in teaching me how to be a better writer. Those things pale in comparison to how much I learned about life with you. You undoubtedly played the biggest role in encouraging me to pursue my PhD. I wish you much happiness and success in your life, as well your family. You will always be missed.

Dr. Chittibabu Guda – Thank you for your guidance and patience to help get me through this dissertation to the end. Thank you for taking a risk in hiring a CS PhD student. It has been a pleasure to work with you and your team. I've enjoyed our countless exchanges about academia and family. I'm looking forward to future collaborations.

Dr. George Berg – Thank you for your guidance in the early years of my research. Thank you for turning me toward the wealth of research opportunities in bioinformatics and machine learning. You have been inspiring as I've watched you model the delicate balance between academia and family.

Dr. Mohammed Zaki – Your enormous amount of high-quality research in this field is truly inspiring. Thank you for taking time out to be on my committee and help get me through this.

Dr. Ian Davison – Thank you for your excellent data mining and machine learning classes.

Dr. Frank Vellutino and Joyce Vellutino – Thank you for many inspiring conversations about academics, politics, and family. Thank you for your encouragement. Your advice and humor over the years meant more than you know.

In memory of
Lance W. Latham
August 12, 2008

This work has been supported by the startup funds to Dr. Chittibabu Guda from the State University of New York (SUNY) at Albany, the graduate student fellowship provided by the Gen*NY*sis Center for Excellence in Cancer Genomics, SUNY at Albany, and the Academic Research Enhancement Grant from NIGMS/NIH (1R15GM080681) to Dr. Chittibabu Guda.

Abstract

Bioinformatics, an interdisciplinary field between computer science and biology, has emerged primarily out of the need for methods to automate the analysis and annotation of newly discovered biological data. In the last decade, there has been an exponential growth in the size of gene and protein sequence repositories resulting from rapid advancements in high-throughput experimental techniques. This resulted in a huge inventory of genes with unknown function. Gene function is executed primarily at the protein level; hence, understanding the functional role of proteins in a species can yield substantial biological information about that species which may have potential applications in biomedical research. Unfortunately, experimental characterization of protein function is tedious and not feasible for all genes. Alternatively, computational methods can complement experimental efforts in annotating the vast amount of these data lacking functional and/or structural characterization.

The computational fields of machine learning and data mining continue to provide a much needed framework for numerous methods that can assist in categorizing these unannotated proteins. Supervised learning methods have played a dominant role in helping us better understand some newly discovered proteins with respect to their functional and/or structural characterizations. As with all supervised learning methods, all training data used for model induction must be labeled.

Semi-supervised learning methods, which learn from labeled and unlabeled data, can have a significant impact on this field of research due to the relatively large amount of unlabeled data available. In theory, this unlabeled data represents a large pool of untapped information that can be used to improve models based on labeled data. Unlike supervised methods, semi-supervised methods are only beginning to emerge in this field,

and are often developed with restrictive requirements that make them unsuitable for analysis and characterization of large-scale sequence data.

This dissertation explores the development of supervised and semi-supervised learned classification methods designed to classify large-scale protein sequence data. A central aim of this research is to develop methods that require only the protein sequence for classification. The majority of work is based on the well-known Naïve Bayes classification framework, which has been proven to perform well in the field of text classification. The parameterized, probabilistic model is developed through observing occurrences of fixed-length subsequences throughout the labeled data. Unlabeled data is used to improve the model by extending the method by incorporating the Expectation-Maximization algorithm.

Using the task of predicting the subcellular localization of a protein sequence, performance results from the supervised method show superior performance over existing methods. Moreover, the subcellular proteome of numerous eukaryotic and prokaryotic species are estimated with far greater coverage than any other method known at the time of this research. Performance results from the semi-supervised learning research will show that large repositories of unlabeled protein sequence data can indeed be used to improve predictive performance, particularly in situations where there are fewer labeled protein sequences available, and/or the data are highly unbalanced in nature. This dissertation has laid a foundation for exploration of numerous other characterizations of proteins on large-scale data.

Tables

Table 1 - Amino Acids.....	24
Table 2 – Overall accuracy on the Reinhardt and Hubbard NNPSL dataset	30
Table 3 - Results for the Park and Kanehisa PLOC dataset	33
Table 4 - Distribution of proteins over eukaryotic subcellular locations.....	58
Table 5 – Results for 7-gram model using entire dataset.....	62
Table 6 - Distribution of 7-grams in ngLOC dataset	67
Table 7 - Class-unique odds ratio for 7-gram model	67
Table 8 – Benchmarking the performance of ngLOC (7-gram) against its confidence score	69
Table 9 - Rank of correct class single- and multi-localized sequences using a 7-gram model.....	70
Table 10 - Evaluation of MLCS against single- and multi-localized sequences	71
Table 11 - Examples of prediction for multi-localized sequences.....	72
Table 12 - Examples of outdated annotations in the PLOC dataset discovered by ngLOC	74
Table 13 - Examples of multi-localized sequences in the PLOC dataset discovered by ngLOC.....	75
Table 14 - Example misclassifications of ngLOC method when training and testing on PLOC dataset	76
Table 15 – Comparison of location-wise prediction percentages for mouse and fruitfly.	80
Table 16 – Estimation of the subcellular proteomes of eight eukaryotic organisms	83
Table 17 - ngLOC subcellular estimates for eight proteomes (actual numbers)	83
Table 18 – A matrix showing estimated fractions of subcellular proteomes on the human proteome	85
Table 19 - Analysis of discriminatory n-grams in domain regions	87
Table 20 – Gram-negative and Gram-positive training data	99
Table 21 - Performance of different n -gram models	103
Table 22 - Coverage vs. precision for gram-negative data	105
Table 23 - Jack-knife performance of ngLOC on training data using a 6-gram model..	108
Table 24 - Comparison of ngLOC against other methods using <i>GP-PSORTb</i> test dataset	110
Table 25 - Performance of ngLOC on <i>GP-test</i> dataset.....	111
Table 26 - Estimation of the subcellular proteome of ten Gram-negative organisms	114
Table 27 - Estimation of the subcellular proteome of five Gram-positive organisms....	114
Table 28 – Labeled datasets of proteins over subcellular localizations.....	143
Table 29 - Performance improvement on 2-class data.....	148
Table 30 - Effect of unlabeled data on classification of 250 sequences from the D_L -2A dataset.	149
Table 31 – The performance results for the D_L -3 dataset across differing values of the λ parameter in the EM- λ algorithm and differing amounts of unlabeled data.....	150
Table 32 - Results for the Multi-EM algorithm.	152
Table 33 – Percentile of data in D_L -3 falling in different ranges of CS based on cross validation.....	154
Table 34 – Results for the EM-CS algorithm.	155

Table 35 – Comparing TSVM against the EM-CS algorithm on DL-2A.....	156
Table 36 – Running time comparison of TSVM against EM-CS.....	157
Table 37 – Results of EM- λ against the transductive SVM on the DL-3 dataset.	158
Table 38 - Most represented species in Swiss-Prot	175
Table 39 - Most represented species in ngLOC dataset.....	176
Table 40 - Amino acid groups	180
Table 41 - Mathematical notation	183
Table 42 - Table of abbreviations	184

Figures

Figure 1 - Number of protein sequences in TrEMBL	5
Figure 2 - The Central Dogma of Molecular Biology	8
Figure 3 - Example Protein Sequence.....	25
Figure 4 - Subcellular Compartments	26
Figure 5 - Derivation of multinomial Naive Bayes classifier	51
Figure 6 - The n-gram model for representing proteins in ngLOC.....	53
Figure 7 - Overall accuracy versus n-gram length.....	62
Figure 8 - ROC curve for 7-gram model	65
Figure 9 - Comparison of predictions from three methods on the ngLOC dataset	73
Figure 10 - ROC Curves for [1-8]-gram models on Gram-negative data.....	104
Figure 11 - ROC curves for [1-8]-gram models on gram-positive data	107
Figure 12 - ROC curves for 6-gram model on Gram-negative data	109
Figure 13 - Derivation of naive Bayes classifier	128
Figure 14 - Basic EM Algorithm	133
Figure 15 - Algorithm for Multi-EM	137
Figure 16 - EM-CS algorithm	139
Figure 17 - Basic EM Algorithm on 2-class dataset	147

1 Introduction

We live in a new era in biological research – the post-genome era. The genome, defined as the complete genetic material (DNA and RNA) of a species, has been sequenced for numerous species. Genomics has brought forth a variety of methods for generating and analyzing these data that have revolutionized biomedical research. However, genomic methods are not without limitation, sometimes resulting in noisy data, inconclusive results, and often requiring an expense on the order of millions of dollars. The proteome, defined as the entire set of proteins expressed in a species, has been the focus of biomedical research in recent years, attributed largely to the need to understand the functional roles of proteins in cellular metabolism that cannot be inferred solely from information derived from the genome. Recent studies have shown that more significant biological information can be determined through analysis of the proteome (Chung, et al., 2006). The greatest insight into the biological functions in a species can be had through a combined analysis at the genome and proteome level (Ideker, et al., 2001; Pandey and Mann, 2000).

Proteins, being the workhorse of a species, carry out all of the functions in the body. When a function in the body fails, it is often attributed to failure of some proteins to carry out their designated functions. Though these failures can often be backtracked to a problem at the gene level, many other problems are post-translational, meaning, the protein was expressed properly, but a problem occurred after the protein was successfully synthesized. Determining the various functional roles of proteins in the body aids in understanding what potential roles these proteins may play in metabolic and disease pathways. The combined analysis of the protein and DNA are resulting in important medical discoveries in diseases and cancer research, with some of these diseases being

attributed to a problem at the protein level alone (Chung, et al., 2006). Furthermore, proteomics can often give a different picture of the cell or tissue being observed that can not be had through analysis of the DNA in the tissue. The majority of drugs developed out of this field of research are designed to target the protein(s) involved in these problems (Smith, 2003). Determining the functional role of every protein in the body will have substantial benefits for the entire human population, and will represent the focus of biological and biomedical research for years to come.

The burgeoning interest in proteomics has resulted in an exorbitant rate in which new proteins sequences are deposited into public data repositories. These protein sequence repositories provide a pool of data containing a wealth of potential information related to a wide range of biological functions. Unfortunately, the rate at which new proteins are discovered substantially outpaces the rate at which these proteins are fully annotated through experimental means, resulting in large repositories of protein data whose characteristics are not known. Computational methods have become a crucial part of the ongoing efforts to understand and annotate these growing data repositories. Such methods mine these data for interesting and/or frequently occurring patterns that may be associated with particular functions of interest. Thus, discoveries from computational methods will set the platform for further experimental validation. There is a critical need for methods that can accurately annotate the functional role of proteins on large-scale protein datasets in order to reduce the widening gap between newly discovered and experimentally determined proteins (Hamady, et al., 2005). The current work attempts to address this issue by applying the supervised and semi-supervised concepts to the field of protein classification and annotation.

1.1 Introduction of research area

Bioinformatics, a joint discipline between computer science and biology, is loosely defined as the science of applying computational methods to biological research. In other words, bioinformatics derives knowledge from computer analysis of biological data. It has arisen out of the need to extract information from the growing biological data repositories in an efficient and accurate manner. The computational fields of data mining and machine learning are among the most used fields in bioinformatics, lending numerous methods to enable categorizing DNA and protein data using models inferred from data that has been experimentally determined. In fact, many general machine learning methods arose out of classification methods used in biological data.

A central aim of bioinformatics is in the development of methods that can aid in the functional and structural annotation of proteins. However, determining this information in an accurate and efficient manner has proven challenging, and has been the focus of the development of hundreds of methods in computational and experimental molecular biology.

A plethora of methods have been generated by the research community to mine through the numerous protein databases to extract structural and functional information from these sequences, as well as to induce predictive models to aid in the annotation of newly discovered protein sequences. Computational methods for protein classification aim to infer a model (i.e., a function) that map proteins to a set of categories, with the intent of using these models to annotate some functional attributes of unannotated proteins.

Proteins, being the functional workhorse of all living entities, have a wide array of possible descriptors used to aid in organizing and classifying them, most of which are

covered by the well-known Gene Ontology project (Ashburner, et al., 2000). Despite the large amount of possible classifications, almost all computational methods for protein classification focus on a narrow range of possible labels, in order to reduce the problem into a tractable one. By definition, proteins are linear chains of amino acids joined by peptide bonds. The information associated with the functional, structural and evolutionary characteristics of the protein is essentially contained in this sequence. The aim of all protein classification methods is to infer this information using knowledge learned from experimentally determined sets of proteins in the most accurate manner possible.

1.2 Motivation

Methods that can operate on the primary structure of a protein (the amino acid sequence representing the protein) are of particular importance for many reasons. First, the primary structure representation has a significant computational benefit, as it uses considerably less computational resources than representing the fully folded, three-dimensional structure (the tertiary structure) of a protein. Recent genome sequencing projects have identified an enormous number of potential protein-coding regions (a.k.a. open reading frames or ORFs), which are computationally translated to obtain hypothetical protein sequences. This has resulted in a significant abundance of protein sequences that are publicly available. The NCBI non-redundant sequence database (Pruitt, et al., 2005), version 27, released January 11, 2008, consisted of 4,426,609 protein sequences from over 100,000 different species, resulting in a total of 1.5 GB of data. The UniProt/TrEMBL database (Boeckmann, et al., 2003), version 38.0, released February 26, 2008, has 5,395,414 sequences from 155,282 species, consisting of 1.7 GB

of data. What is more astounding, however, is the exponential rate at which sequence data are being added to these databases. (See Figure 1 for a graph of the size of the UniProt/TrEMBL database by date.) Therefore, it is important to develop methods that not only operate on the primary structure of a protein, but also operate on large scale databases, particularly at the proteome-level. Most of the methods available today are not suitable for proteome-wide predictions, as they do not consider computational resources necessary for such large-scale predictions, and/or available computational resources are not effectively used.

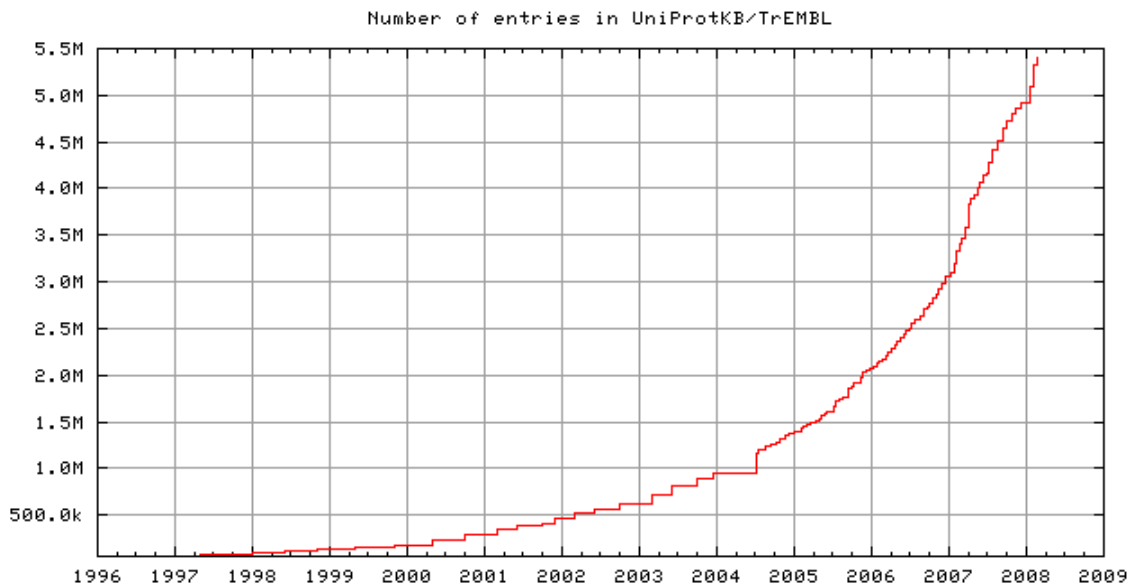


Figure 1 - Number of protein sequences in TrEMBL

This graph shows the number of protein sequences in the TrEMBL protein database by year. [See http://www.ebi.ac.uk/swissprot/sptr_stats/index.html]. TrEMBL is a component of the UniProt database, which contains computer generated protein sequences that lack experimentally derived functional annotations.

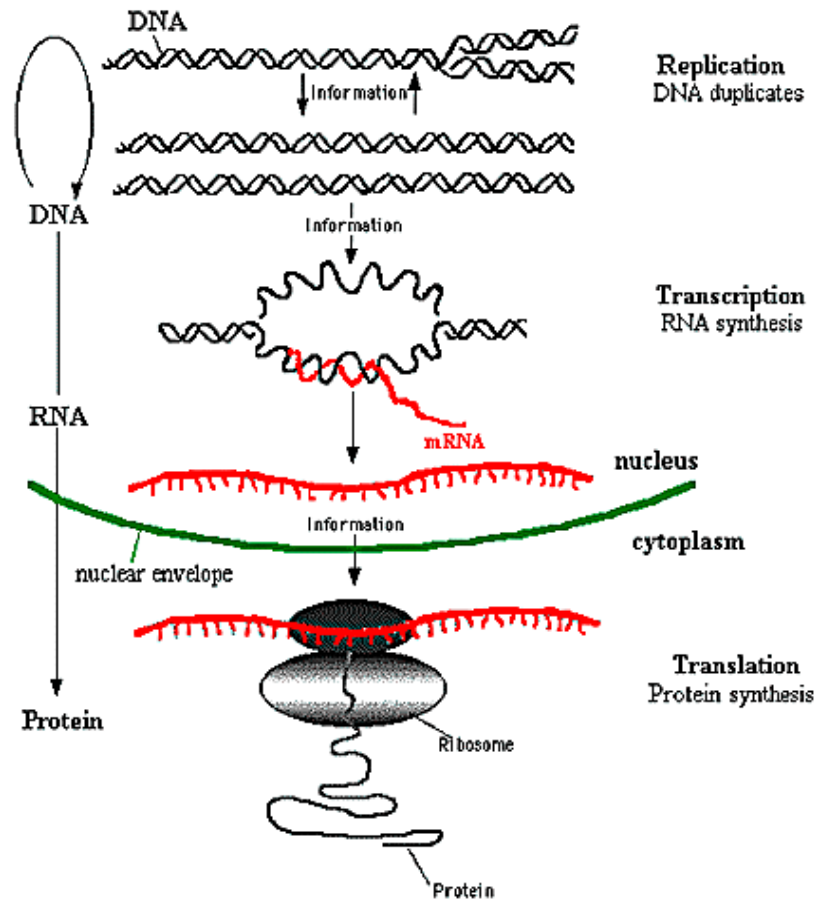
Though the number of proteins available in public databases is large, and growing larger by the day, only a small fraction of these sequences have some type of functional annotation available determined through experimental methods. The Swiss-Prot database (Boeckmann, et al., 2003) is a universally accepted standard sequence repository that

provides a high level of experimentally-derived annotation on each sequence. The majority of protein classification methods are built and/or tested using sequences obtained from this widely trusted source. A recent release of the Swiss-Prot database, version 55.0, released February 26, 2008, has only 356,194 entries, none of which are redundant in TrEMBL. This means that only 6.2% of all of the sequences in UniProt (Swiss-Prot + TrEMBL) have some experimentally-derived annotation. (The sequences in TrEMBL that have some level of description are annotated by automated computational methods, and thus are not considered reliable data for constructing training datasets for computational methods.) Moreover, the fraction of sequences in UniProt with high level annotation is continuing to decline at an alarming rate (e.g. the UniProt database, version 51.2, released November 28, 2006, had 7.1% of the database containing high-level annotation.)

Another significant challenge is the development of methods that can readily adapt to new and updated sequence annotations as they are made available. It is becoming well known that the results of genomic sequencing projects contain errors. These errors, once thought to contain mostly point mutations in the deposited genome, may also consist of other prevalent large-scale rearrangements and discrepancies in the genome (Mandel, et al., 2008). Such errors are often the result of frameshift errors, resulting in improper identification of ORFs. Even worse is the discovery that some errors are missed insertions and/or deletions of large, contiguous regions of the genome. These errors resulting from ongoing genomic sequencing studies are more common than once thought, stressing the need for ongoing quality control in these systems (Mandel, et al., 2008). Regardless, such errors can introduce errors in the inferred proteins resulting from these genomic studies, adding to an already difficult problem of inferring

information from these large protein sequence repositories. This again demonstrates the need for systems that are robust in the midst of errors and inconsistent data. These systems need to adapt to changes in protein sequence repositories, and can readily implement subsequent improvements and refinements in the inferred model. Protein classification methods also must be able to easily adapt to new sequence data as they become available.

Sometimes sequence discrepancies are a result of the biological mechanisms in place that we are only beginning to understand. New technologies have increased our understanding of life at the molecular level, and in some instances, the findings have dramatically altered long-standing theories and assumptions. One such example is the Central Dogma of Molecular Biology, which has provided a long-standing, simplistic model of the flow of information in the cell. It assumes a one-to-one mapping between DNA and RNA during transcription, and between RNA and the protein synthesized during translation. However, recent discoveries have shown that there are mechanisms in place that often violate these assumptions. In fact, many-to-many mappings have been found at each level. It is also known that certain amino acids in a protein can be altered or removed after translation, a process known as post-translational modification, implying that gene expression does not always translate into the changes observed in proteins (Hamady, et al., 2005; Pandey and Mann, 2000). These and other violations are a direct result of the fact that, though the genome is a relatively static entity, the proteome is highly dynamic, responding to various environmental changes both inside and outside the cell (Komatsu and Yano, 2006).



The Central Dogma of Molecular Biology

Figure 2 - The Central Dogma of Molecular Biology

The Central Dogma of Molecular Biology provides a simplistic model of the flow of information in the cell. [Copyright 1999, the National Health Museum]

Though it is clear that systems are needed that operate from the primary structure of a protein, the need for methods that operate from *only* the primary structure must be emphasized. Only a small fraction of protein data has any type of annotation available beyond the sequence. Therefore, it is important to develop protein classification methods that require only information that can be derived from the protein sequence as input to the classifier. Unfortunately, many methods require additional information beyond the sequence, such as phylogenetic, functional or structural information, thereby limiting

their coverage to only the small fraction of sequences which have the additional information available.

As stated previously, almost all computational methods work through analysis of a well-defined set of protein data that is labeled over some predefined set of annotations related to some attribute of protein function or structure of interest. Whether stated explicitly or implicitly, almost all methods establish a prior distribution either from expert knowledge or directly from the data used for model induction. Rarely is this prior distribution uniform across all classes considered in the model. Many computational methods perform poorly on unbalanced datasets. The solution many methods have adopted is to ignore prediction of small, under-represented classes, thereby predicting only the most populated classes. Other methods that use the classes of data with low-representation often result in a classifier that over-predicts the most highly represented class in the data, thereby showing a lack of balance between precision and sensitivity among the different classes. This limits the usefulness of such methods, particularly for proteome-wide predictions.

Finally, it is a fact that the proteins that have been experimentally annotated were not chosen randomly from the universe of proteins. They usually represent proteins that have some medical or other related significance to the research community, or perhaps they are proteins that can be easily analyzed through in vitro or in vivo methods. This presents a possible sample selection bias in the set of annotated proteins available, and thereby can potentially introduce a bias in classifiers.

It is clear that modern, high-throughput proteomic methods are presenting many significant challenges to current methods in protein classification. Protein data without full, high-level annotation represents a large fraction of the available protein data. These

unlabeled proteins represent untapped information that can be used to improve classifiers that are based on annotated sequences alone. Though semi-supervised learning methods are being actively researched in the machine learning community, the field of bioinformatics has only begun to research methods that incorporate both annotated and unannotated data to improve classifier performance.

The bottom line is that high-throughput genomic sequencing continues to generate an ever-increasing quantity of protein data that needs to be analyzed, but very few methods are available to do so. Existing methods exhibit a variety of limitations as outlined above. This research focuses on the development of a method that will address each one of the above issues, with the ultimate goal of inducing a machine-learned classifier from large-scale protein datasets consisting of both annotated and non-annotated data. The need for such a method has been addressed, and the success of this work will have a significant impact on the bioinformatics research.

1.3 Structure of thesis

This thesis will focus on the development of supervised and semi-supervised Bayesian classification methods that can be used to analyze large-scale, proteome-wide protein datasets. Essential background required for the research will be presented first, following by a literature review covering existing methods related to this field of work. A thorough exposition of a supervised and a semi-supervised approach to protein sequence classification will be reported. Results will be presented and discussed, ending with a conclusion, as well as future opportunities in which this work can be extended.

2 Essential Background

Bioinformatics research can be a difficult research area to fully grasp, due to its multi-disciplinary nature between biology and computer science. This chapter will review the most essential background to ensure that a molecular biologist or computer scientist understands the most important information required to understand the thesis. The majority of this chapter presents pertinent machine learning and data mining knowledge, which are discussed extensively in numerous standard texts in machine learning and mathematical statistics (Alpaydin, 2004; Bishop, 2006; Duda, et al., 2001; Hastie, et al., 2001; Hogg, et al., 2005; Mitchell, 1997). Pertinent biological information for computer scientists is presented at the end.

2.1 Machine learning and data mining

The field of classification is domain-independent, and has many names (e.g., statistical inference, pattern recognition, etc.). Regardless of the name, the goal is to develop a model that can be used to assign objects to a category, with the highest accuracy possible. Computational methods for classification draw from research in machine learning and data mining, which themselves are largely based on theory from probability and mathematical statistics.

Classification in machine learning is often broken up into three distinct areas: (i) *supervised learning*, (ii) *unsupervised learning*, and (iii) *semi-supervised learning*. In supervised learning, we are given a set of data that is labeled with the concept being learned, with the aim of inducing a model that can label new data with a high level of accuracy. In most cases, the algorithm outputs a discriminative function that maps data to a concept. Decision trees, artificial neural networks, and support vector machines are

all examples of supervised learning algorithms. In unsupervised learning, a model of the input is induced, but no mapping to any output is created, as the existing data is not labeled. This is appropriate in cases where a model is desired that learns natural groups or arrangements of the data. Clustering is a common form of unsupervised learning. In semi-supervised learning, a model is induced from both labeled and unlabeled data, again with the intent of labeling new data. Expectation maximization methods used in conjunction with supervised learning methods are often used in semi-supervised learning.

In the context of learning, statistical inference deals with problems where a set of data is given, and the method learns about the underlying probability distribution that generated the data. There may be one, several, or an infinite number of possible distributions that may have generated the data. Some methods may simply choose the most likely distribution to model the data, while other methods may consider all possible distributions in the hope of obtaining a better overall model. Whatever the method, the resulting model can be used to infer something useful about existing data, about future data yet to be seen, as well as to generate possible data that may be seen in the future (DeGroot and Schervish, 2002).

The rest of this chapter is focused on establishing the essential framework for classification. An emphasis on protein sequence classification will be covered at the end.

2.1.1 Five steps for classification

In general, there are five important steps that must be considered for successful classification:

1. **Data collection** – Let $\mathcal{D}$ denote the domain of interest. (In this research, $\mathcal{D}$ represents the set of all possible proteins, and will be referred to as the protein

space.) A sufficient number of instances from $\mathcal{D}$ must be collected for learning, or training a model of the data. The dataset of examples which will be used for training is called *training data*, and will be denoted as D . Thus, $D \subseteq \mathcal{D}$. Let d_i represent the i^{th} instance in D . Each instance will belong to a class (or category), denoted as y_j , from a set of possible classes, denoted as $\mathcal{C}$.

2. **Feature selection/Input representation** – A set of features must be selected to represent instances that are to be classified. The feature set must be easy to extract from the data, and highly discriminatory among the different classes, yet general enough to allow representation of data in the domain that is yet to be seen. Let $\mathcal{X}$ represent the *feature space* – the space of all possible features that could be used to represent any instance in $\mathcal{D}$. The function that maps instances in $\mathcal{D}$ to features over $\mathcal{X}$ is assumed to be neither injective nor surjective; that is, there may be distinct instances in $\mathcal{D}$ that map to the same feature vector, *and* there may be feature vectors over $\mathcal{X}$ for which no instance in $\mathcal{D}$ could possibly exist in nature. The only assumption is that each unique instance in $\mathcal{D}$ maps to one feature vector over $\mathcal{X}$. Let $\mathbf{x}_i$ represent the vector of values of features representing instance d_i in D . Then, D is a set of ordered pairs $(\mathbf{x}_i, y_j)$. However, the notation $d_i \in D$ and $\mathbf{x}_i \in D$ are both used in this research to refer to the same i^{th} instance in the training data D ; the representation is implied in the notation.
3. **Classification algorithm/model** – Machine learning and data mining offer many choices of classification algorithm. It is not always clear which algorithm is the best for which problem. (This is clearly an issue in protein classification, indicated by the numerous methods which have been used for predicting the same

concept.) Regardless of the choice, the classification algorithm will provide the framework for learning.

4. **Learning** – Once the algorithm has been established, and the form of the model is known, learning takes place. It is here where the training data is analyzed to establish patterns between the data and the classes the data is assigned to. Successful learning results in the induction of a hypothesis that best models (or explains) the data being analyzed. The hypothesis is output as a target function, denoted as h , which maps instances represented by their features to a class, denoted as:

$$h: \mathcal{X} \rightarrow \mathcal{C}$$

One can think of the feature vector as a uniform description of the object to be classified. The classifier evaluates the description of the new object, and assigns the category based on the categories that the best matching descriptions of known objects belong to. The classifier is a hypothesis that best explains the differences in the descriptions between objects belonging to each class. Learning involves determining what descriptors make the objects unique to the class, and adjusting the model to best exploit those descriptors.

5. **Evaluation** – In reality, the hypothesis will not perfectly explain everything. The classifier will likely make mistakes. We want to select the model that makes the fewest mistakes. One interpretation of machine learning is focused on the development of computational methods that optimize a *performance measure* using example data or past experience (Mitchell, 1997). The performance measure for a hypothesis is usually a quantitative measure of the error between the hypothesis and the true concept being learned on some data being tested.

(Machine learning algorithms "learn" in a sense that the performance of the method is designed to improve as more data becomes available.)

In this study, the focus will be on Bayesian classification, which uses a probabilistic, parametric model to learn the relationship between the data and the classes that the data may belong to. The best model induced by a probabilistic classifier will be the one that models a distribution that is most aligned with the true, underlying distribution that generates $\mathcal{D}$. The model will be assumed to have a parametric form that is established prior to learning. Probabilistic methods have significant value as they allow the classification model to inherently represent the uncertainty in the data and the classes that the data belong to. Uncertainty can arise simply due to noise in the data acquisition process, poor feature selection methods, or error in classification assignment. These are all very realistic, common occurrences in biological data, and hence, is the reason why a pure probabilistic approach is used for this research.

2.1.2 Random variables for probabilistic classification

Random variables are used to represent the data and the classes that the data can be assigned to. Let $\mathbf{X}$ be a random variable representing the features of the data to be classified. ($\mathbf{X}$ is a vector of random variables $X_1, X_2, \dots, X_n$ where each individual variable in the vector is a unique *feature* used to describe the instance.) Let Y be a random variable representing the class that the data can be assigned to. The goal of a classifier is to learn a model of $P(Y | \mathbf{X})$ – the *posterior* probability of the class, given the data.

2.1.3 Discriminative vs. generative classification

A *discriminative* classifier learns a model of the posterior probability $P(Y | \mathbf{X})$ directly. In this case, the classifier outputs a target function that directly maps instances $\mathbf{X}$ to classes Y . Logistic regression is one example of a discriminative classifier. A *generative* classifier learns a model of the joint distribution generating $(\mathbf{X}, Y)$. Simple probability theory allows the posterior to be calculated from the joint distribution as follows:

$$P(Y | \mathbf{X}) = \frac{P(\mathbf{X}, Y)}{P(\mathbf{X})} = \frac{P(\mathbf{X}, Y)}{\sum_{y_j \in \mathcal{C}} P(\mathbf{X}, Y = y_j)} \quad (2.1)$$

Classification methods that learn these parameters are known as generative classifiers, because the model can be used to generate example instances of the data for each possible category. Generative classifiers assign the most probable value of Y , given the data, as shown in Eq. (2.2).

$$h(\mathbf{x}_i) = \arg \max_{y_j} P(Y = y_j | \mathbf{X} = \mathbf{x}_i) \quad (2.2)$$

Modeling the joint distribution is often a difficult task to do. Instead of modeling the joint distribution directly, Bayes rule allows an alternative form for calculation of the posterior, resulting in:

$$P(Y | \mathbf{X}) = \frac{P(\mathbf{X}, Y)}{P(\mathbf{X})} = \frac{P(\mathbf{X} | Y) P(Y)}{P(\mathbf{X})} \quad (2.3)$$

In this research, the primary classification method discussed will be a generative method known as Bayesian classification, which is based on Eq. (2.3).

2.1.4 Bayesian classification

Eq. (2.3) and the product rule in probability show that the joint distribution $P(\mathbf{X}, Y)$ can be modeled through two factors – $P(Y)$ and $P(\mathbf{X} | Y)$. Learning a model for $P(\mathbf{X} | Y)$ assumes that data is generated according to a *conditional* distribution dependent on the category Y . This probability is referred to as the *likelihood*, implicitly modeling the likelihood of observing data $\mathbf{X}$, given the class Y . The underlying density for this probability is known as the *class-conditional density*. We assume that all categories have a *prior* probability of being the correct category of a data instance, denoted as $P(Y)$. This is the prior probability that the instance belongs to a specified class without any information about the instance being classified. The denominator, $P(\mathbf{X})$, is commonly referred to as the *evidence*, and represents the prior probability of observing the data, regardless of the category it belongs to. It can be estimated by marginalizing the joint distribution over all possible categories as follows:

$$P(\mathbf{X}) = \sum_{j=1}^{|C|} P(\mathbf{X} | Y = y_j) P(Y = y_j) \quad (2.4)$$

It is common to disregard the denominator in Bayesian classification methods, as it is independent of Y , and does not affect the final classification. Therefore, obtaining a model for classification involves obtaining models for the likelihood $P(\mathbf{X} | Y)$ and the prior probability $P(Y)$.

For $P(\mathbf{X} | Y)$, the class-conditional density is a probability density function (pdf) with a known parametric form. Each class will have its own set of parameters to be learned. These class-conditional parameters will be represented by the vector θ_j , the entire vector of all parameters will be represented by θ , and the *model space* (or *parameter space*) over all possible values of θ are represented by Θ .

Estimation of the prior $P(Y)$ is a known source of controversy among statisticians. "Frequentist" statisticians object to the use of a prior probability, as it requires representation of the prior through a random variable, even though in reality it is a fixed (yet unknown) value. Bayesian statisticians believe it is acceptable to model the prior as another distribution, because it allows the uncertainty that is present before seeing the data to be modeled. It also allows the uncertainty to be updated after the data is seen through the class posterior (DeGroot and Schervish, 2002; Hastie, et al., 2001). Because this research is using Bayesian classification, the prior will be modeled as a random variable whose value is estimated from the sample data, and when possible, through expert knowledge of the domain of interest. The parameters for the prior will be denoted as φ .

The Bayesian classifier induced, denoted as h , will allow classification on any data instance $\mathbf{x}_i$, using the model depicted in Eq. (2.3). Showing the explicit dependency on the model parameters $\boldsymbol{\theta}$ and φ , our classification model becomes:

$$h(\mathbf{x}_i) = \arg \max_j P(\mathbf{x}_i | y_j, \boldsymbol{\theta}_j) P(y_j | \varphi) \quad (2.5)$$

So far, the uncertainty surrounding the model parameters $\boldsymbol{\theta}$ and φ has been ignored. The crucial part of inducing a successful probabilistic classifier lies in determining not only the form to assume for the densities, but more importantly, finding the best estimates for the parameters of these densities. The best parameter estimates are those that give the best model performance. An *objective function* for classification is a function that maps the model parameter space to a quantitative value corresponding to the performance of the classifier. Hence, finding the best estimates for the model parameters is viewed as a problem of maximizing the objective function for the classification model.

2.1.5 Objective function for Bayesian classification

A theoretical interpretation of learning can be put in terms of maximizing an objective function that models the performance of the classifier. The performance is usually measured in terms related to the empirical measurement of the number of instances correctly classified. For a probabilistic method such as a Bayesian classifier, the objective is to find the parameters that maximize the likelihood of observing the data, given the parameters. Simply put, our objective function is:

$$\arg \max_{\boldsymbol{\theta}, \varphi} P(\mathcal{D} | \boldsymbol{\theta}, \varphi)$$

However, this is an impossible task, because we do not have the entire population of data to determine this. We could use the training data D as a basis for maximizing the objective, but doing so results in a classifier that suffers from *overfitting* the training data. A common solution is to obtain an independent set of test data assumed to be drawn from the same distribution that D is drawn from. The test data will serve the sole purpose of measuring the performance of the classifier, and is not used for establishing parameters.

Let D_{test} be the data used for testing the classifier. The objective function now becomes finding the values of the parameters that give the highest likelihood of observing the training *and* test data, given the parameters:

$$\arg \max_{\boldsymbol{\theta}, \varphi} P(D_{test}, D | \boldsymbol{\theta}, \varphi) \tag{2.6}$$

We now have a framework in which to derive the parameters to achieve this goal.

2.1.6 Parameter estimation

The objective function in Eq. (2.6) gives a fundamental basis for evaluating a proposed hypothesis induced by the classification algorithm. We assume all instances in D_{test} and

D are identically and independently distributed (IID) according to a fixed, unknown distribution specified by the model parameters θ and φ . However, the model parameters, which form the basis of the hypothesis, are determined from the training data only. Thus, the overall aim during the learning phase is to find the model parameters based on $P(D | \theta, \varphi)$ – the likelihood of observing our training data, given the parameters. The model will then be selected, rejected, or adjusted, based on $P(D_{test} ; \theta, \varphi)$ – the probability of observing the test data given the fixed parameter estimates determined during the learning phase.

Because the data is assumed to be IID, the probability in Eq. (2.6) now becomes:

$$P(D_{test}, D | \theta, \varphi) = P(D | \theta, \varphi) P(D_{test} ; \theta, \varphi)$$

As stated, learning parameter estimates are based on the training data only, and can be broken down as follows:

$$\begin{aligned}
P(D | \theta, \varphi) &= P(\{D_1 \cup \dots \cup D_C\} | \theta_1, \dots, \theta_C, \varphi) \\
&= \sum_{j=1}^{|C|} P(D_j | \theta_j, \varphi) \\
&= \sum_{j=1}^{|C|} \left(\prod_{\mathbf{x}_i \in D_j} P((\mathbf{x}_i, c(\mathbf{x}_i)) | \theta_j, \varphi) \right) \\
&= \sum_{j=1}^{|C|} \left(\prod_{\mathbf{x}_i \in D_j} P((\mathbf{x}_i, y_j) | \theta_j, \varphi) \right) \tag{2.7} \\
&= \sum_{j=1}^{|C|} \left(\prod_{\mathbf{x}_i \in D_j} P(\mathbf{x}_i | \theta_j) P(y_j | \varphi) \right) \\
&= \sum_{j=1}^{|C|} P(y_j | \varphi) \left(\prod_{\mathbf{x}_i \in D_j} P(\mathbf{x}_i | \theta_j) \right)
\end{aligned}$$

To explain, the training data can be broken up into disjoint subsets D_j whose members belong to the same class y_j . Recall that D is a set of ordered pairs $(\mathbf{x}_i, c(\mathbf{x}_i))$,

where $c(\mathbf{x}_i)$ represents the class that instance $\mathbf{x}_i$ is assigned to. Then $c(\mathbf{x}_i) = y_j$ for all instances in class y_j . For the class prior, the best estimate for parameter φ will depend on the observable class distribution in D . This derivation shows that the estimates for the prior φ and the likelihood parameters θ can be handled independently. This presents a potential problem in estimating the prior parameter φ . If the data in D represents a large sample that is drawn according to the *true* distribution, then the training data can be used as a basis to estimate the prior. Otherwise, φ will need to be set according to expert knowledge of the domain. Failure to recognize this problem can result in skewed classifications toward the classes most represented in the dataset due to a *sample selection bias* in the training data. (This certainly justifies the frequentist argument against Bayesian estimating of prior probabilities in this fashion.) Correction and detection of sample biases in training data is an area of active research, and is an area that will be considered in this research as well.

Two statistical methods that are commonly used for estimating these parameters are called maximum-likelihood estimation methods, which is used for finding the parameters that maximize the likelihood, and expectation maximization (EM), which can be used to estimate both the likelihood and the prior parameters. These methods are covered in the Appendix, sections A.3.1 and A.3.2.

2.1.7 Bayesian classifier as a mixture model

In a mixture model, the data is assumed to be generated according to some fixed number of distinct distributions, referred to as *components*. Each component has the same form for its pdf, but a different set of parameters. Each distribution also has a proportion factor that determines how much it contributes to the overall population. Mixture models are

typically used in unsupervised learning, where the goal is to learn which distribution (or component) generated what data. Learning involves determining all of the component parameters and the proportions such that the probability that the mixture model generated all of the data is maximized. (The k -means clustering can be thought of as a mixture model of k components.)

Formally, let p denote the pdf for component j , where each component has its own parameter vector θ_j . Let α_j denote the mixture proportion for component j . Once again, θ will represent the entire set of parameters over all the components. Then, in a mixture model, the probability of any instance $\mathbf{x}_i$, given θ , is specified according to Eq. (2.8):

$$P(\mathbf{x}_i | \theta) = \sum_{j=1}^K \alpha_j p(\mathbf{x}_i | \theta_j) \quad (2.8)$$

Recall that in supervised training, we have a set labeled data D , consisting of ordered pairs $(\mathbf{x}_i, y_i)$. To model this data using a mixture model, we first assume that each component is generating one class of data. Thus, there are $|C|$ components in the mixture model. Furthermore, the data is labeled, and thus we have prior knowledge of which component is generating which data elements. The key difference is in the interpretation of the prior, where the prior probability of class y_j interpreted as the mixture proportion α_j in a mixture model. Eq. (2.9) below shows that this model has precisely the same form as the result of the derivation in Eq. (2.7):

$$\begin{aligned} P(D | \theta) &= \prod_{\mathbf{x}_i \in D} p(\mathbf{x}_i, y_i | \theta) \\ &= \sum_{j=1}^{|C|} \alpha_j \prod_{\mathbf{x}_i \in D_j} p(\mathbf{x}_i | \theta_j) \end{aligned} \quad (2.9)$$

2.2 *Molecular biology*

This section will cover the knowledge from molecular biology that is required for computer scientists. Proteins and how proteins are represented will be covered. Additionally, subcellular localization will be presented, as it represents the primary functional attribute of proteins that the thesis focuses on.

2.2.1 Protein representation

Part of developing a successful classifier is determining the feature space $\mathcal{X}$ that will be used to represent the objects being classified. By definition, a protein is a sequence of amino acids joined by peptide bonds. But, it is much more complicated. A newly synthesized protein, being the workhorse of any living species, has a distinct function to carry out during its lifetime. This single protein is most likely part of a large network of interacting proteins. The successful function of these networks of proteins forms the metabolic pathways in any living organism, and thus is essential to the life of that organism. A single protein failing to function properly can have a cascading effect on a pathway, resulting in a variety of problems in the organism, including numerous diseases, and sometimes even death.

Determining the function of each protein in an organism is essential to understanding the causes of the problems that can occur in different species. In order to understand protein function, one needs to consider the various structural representations that are used to represent proteins. The sequence of amino acids of a protein is referred to as its *primary structure*. Local segments of primary structure can fold by hydrogen bonding, yielding *secondary structures*. To carry out its function, a newly synthesized protein sequence needs to fold into a unique three-dimensional shape, which is referred to as the *tertiary structure* of the protein. Both the secondary and tertiary structures are

determined mostly by the primary structure. In other words, in general, a protein sequence determines its shape and, therefore, determines its function. (Though this represents the general rule, there are exceptions, as the unique fold taken by a protein can be altered by a variety of factors in the surrounding aqueous environment in the cell in which the protein resides, such as charge and pH.)

Given these complexities, one needs to determine how a highly organic, living object such as a protein will be represented in a computer for computational methods. The majority of methods, as well as this research, will use the *primary structure* of the protein. There are two reasons for this choice: it is the simplest representation of the protein to use, and there is an overwhelming amount of protein sequence data available.

Since a protein is a sequence of amino acids, a unique letter is typically used to represent each of the possible amino acids in the protein. There are 20 possible amino acids, which are specified in Table 1 below. The letter code representing each amino acid is specified in the table.

Table 1 - Amino Acids

Amino Acid	Abbreviation	Letter Code
Alanine	ALA	A
Arginine	ARG	R
Asparagine	ASN	N
Aspartic Acid	ASP	D
Cysteine	CYS	C
Glutamic Acid	GLU	E
Glutamine	GLN	Q
Glycine	GLY	G
Histidine	HIS	H
Isoleucine	ILE	I
Leucine	LEU	L
Lysine	LYS	K
Methionine	MET	M
Phenylalanine	PHE	F
Proline	PRO	P
Serine	SER	S
Threonine	THR	T

Tryptophan	TRP	W
Tyrosine	TYR	Y
Valine	VAL	V

Therefore, the primary structure of a protein is represented in a computer by a string of amino acids over the alphabet Σ , where:

$$\Sigma = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}, |\Sigma| = 20 \quad (2.10)$$

Figure 3 shows an example protein sequence in Fasta format, representing a protein found in the human liver:

```
>FABL_HUMAN (P07148) Fatty acid-binding protein, liver
(L-FABP) [Homo sapiens (Human)]
MSFSGKYQLQSQENFEAFMKAIGLPEELIQKGKDIKGVSEIVQNGKHFKFTITA
GSKVIQNEFTVGEECELETMTGEKVKTVVQLEGDNKLVTTFKNIKSVTELNGDI
ITNTMTLGDIVFKRISKRI
```

Figure 3 - Example Protein Sequence

2.2.2 Subcellular localization as a means to infer function

Proteins are synthesized in the cytoplasm of a cell, but are destined to localize into specific subcellular compartment(s) before they carry out their intended functions. This process is known as protein subcellular localization (Figure 4 illustrates the various subcellular compartments that are found in animal cells). Accurate knowledge of the subcellular localization of proteins is highly sought in biomedical research because most cellular processes are spatially constrained in the three-dimensional cell. Knowledge of this spatial context is essential in order to gain a better understanding of the functions of proteins involved in various metabolic and disease pathways that span across subcellular boundaries. The aim of this research is to develop a computational method that can aid in accurate prediction of the subcellular localization of proteins. Such a method can be used

to infer significant information with respect to the function of that protein. For example, proteins that localize to the mitochondria are related to and/or directly involved with energy metabolism. Knowing the organelle in which a protein performs its function can also aid in understanding protein-protein interactions in a spatial context. It has been discovered that the failure of proteins to be transported to their correct destinations in the cell plays a significant role in several human diseases, including cancer (Donnes and Hoglund, 2004).

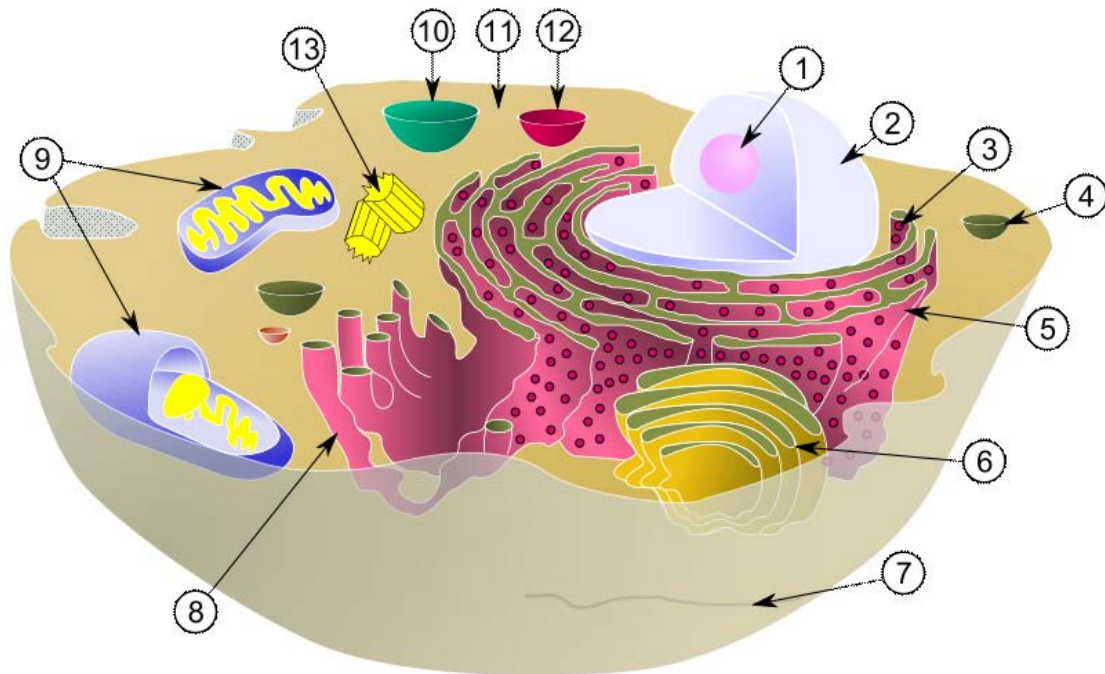


Figure 4 - Subcellular Compartments

This figure illustrates the various compartments, or organelles, that can be found in a typical animal cell. The individual organelles are: (1) nucleolus, (2) nucleus, (3) ribosome, (4) vesicle, (5) rough endoplasmic reticulum, (6) golgi apparatus, (7) cytoskeleton, (8) smooth endoplasmic reticulum, (9) mitochondrion, (10) vacuole, (11) cytoplasm, (12) lysosome, and (13) centriole. (Wikipedia, 2006)

In protein subcellular localization prediction, predictive models are induced from experimentally annotated datasets containing subcellular localization information, with the objective of using these models to predict the subcellular localization of a protein with an unknown localization.

3 Literature Review

The purpose of this chapter will be to review the current status of this field of research. Appropriate background material will be included from various sources. Strengths and weaknesses of research conducted by others will be discussed. The reader will have a clear understanding of the limitations of current methods in this field, thereby understanding the justification for this proposal.

There are hundreds of papers and dozens of books that have been published over the past 20 years covering a wide array of methods in protein classification. The differences between the methods segregate into three areas: (i) the model used to represent the protein, (ii) the target concept being learned by the classifier, and (iii) the classification algorithm used to learn the mapping between proteins and the target concepts. As computers continue to evolve, so does the complexity of both the protein model and the classification algorithms. Methods that could not be considered 15-20 years ago are being pursued today, thanks to increasing computational resources that are available.

This chapter will present an overview of the most influential methods in protein subcellular localization, as well as the most recent trends in this field. The focus will be on comparisons between the different protein representation models and classification algorithm used in these methods. It is difficult to give an unbiased, uniform comparison between all of the methods, as each method reports different success rates for different datasets. Furthermore, there is no known uniform, unbiased comparison of these methods on a standard dataset. Therefore, the results of these different methods will not

be discussed unless a comparison can be made against other methods using the same training and testing data.

3.1 Supervised methods

The study that is considered to be one of the most influential and seminal work in protein subcellular localization prediction is the research completed by Nakai and Kanehisa on a method called PSORT (Nakai and Kanehisa, 1992). Their work resulted in a knowledge base of ‘if-then rules’ that can predict 14 possible localizations on eukaryotic proteins from the amino acid sequence. These rules were extracted by considering amino acid compositions and known motifs in protein targeting against target localizations. Their overall accuracies have served as the benchmark against which numerous other methods have been tested. Another pioneering method was the work of Nakashima and Nishikawa. Their work was based on knowledge that proteins could be classified into natural groupings based on amino acid compositions alone (Nakashima and Nishikawa, 1994; Nishikawa, et al., 1983). Their method performed basic statistical analysis on both single and pair-wise amino acid compositions. The method was very limited, in that it was only a binary classifier, distinguishing between intracellular and extracellular proteins. However, it was significant in that it was one of the few methods that required only amino acid sequences for training the classifier.

Pure statistical-based classification methods continued to abound. In 1999, another significant entry into the field was the work by Chou and Elrod, which employed covariant discriminant analysis to predict 12 possible subcellular localizations (Chou and Elrod, 1999). Their algorithm learns a discriminant function based on the correlation between the subcellular location of a protein and its amino acid composition. Chou later

extended this algorithm by introducing additional features derived solely from the protein sequence, which he calls the “pseudo amino acid composition” (Chou, 2001).

Essentially, in addition to the basic amino acid composition and pair-wise composition, this method adds features derived by considering single-gap pairs of amino acids, 2-gap pairs, etc., up to a fixed maximum gap pair. For even more features, they converted the amino acid sequence into different sequences of real numbers, consisting of hydrophobic values, hydrophilic values, and the side-chain mass of the amino acid at each position of the sequence. This representation model significantly increased the number of features derived from the protein sequence, and the richer feature space allowed more discrimination between subcellular localizations, thereby producing the best results at that time. Many methods expand the feature space by manually combining amino acid composition in tandem with other known functional attributes of the protein in question. For example, the pTARGET method (Guda and Subramaniam, 2005) requires known location-specific functional domains, as well as the amino acid composition of each protein being predicted.

Numerous methods continued to use variants of single and pair-wise amino acid compositions, as it was clear that methods that operated on the sequence alone were needed. At the time subcellular localization prediction was gaining significant attention, machine learning methods were making a considerable contribution to a variety of protein classification tasks. One of the first such methods was the work of Rost and Sander, where they implemented an artificial neural network (ANN) for secondary structure prediction (Rost and Sander, 1993). It was after this seminal work that many researchers took notice and accepted the potential that could result from these methods. In protein subcellular localization prediction, Reinhardt and Hubbard were among the

first to use a machine-learned classifier with their NNPSL method (Reinhardt and Hubbard, 1998). In their work, they assembled a standard dataset of 997 prokaryotic proteins and 2,427 eukaryotic proteins, with a sequence identity of 80%. This dataset formed one of the first standard datasets used for this task, often used by other methods for comparison purposes. The NNPSL method employed an artificial neural network (ANN) to predict the localization over four eukaryotic and three prokaryotic subcellular compartments. The method of Hua and Sun was among the first to use the support vector machine (SVM) classification method for subcellular localization prediction (Hua and Sun, 2001). This method, along with the method of Chou and Elrod mentioned above, were trained and tested using the same dataset as Reinhardt and Hubbard. All three methods predict the same three prokaryotic and four eukaryotic subcellular localizations, and were compared using the same measures. Table 2 shows the comparative results between these methods.

Table 2 – Overall accuracy on the Reinhardt and Hubbard NNPSL dataset

	Reinhardt and Hubbard (ANN)	Chou and Elrod (Covariant discrimination)	Hua and Sun (SVM)
Prokaryotic	81%	86.5%	91.4%
Eukaryotic	66%	73.0%	79.4%

These results showed the success that can result from using SVM methods and, without surprise, methods that used SVMs as the classifier in subcellular localization started to appear in leaps and bounds, and still do today. (In fact, a simple search at any of the bioinformatics research repositories will reveal that SVM classification methods are currently one of the most widely used methods in biological classification tasks.) These SVM methods are distinguished by differences in the dataset used for training,

different numbers of subcellular localizations predicted, by the organization of the individual SVM classifiers, or by the feature space used to represent the protein into the SVM. Briefly, SVMs are binary classifiers. They learn to classify data by determining a hyperplane through the feature space that maximizes the margin required to separate two classes of data. Therefore, an ensemble of SVM classifiers is always required when dealing with any classification over more than two target concepts.

The primary drawback of most of these methods was in the limited number of localizations predicted. The majority of these methods ignored classification of the smaller organelles in the cell, likely because they have fewer proteins localized to them. Adding these classes into the training data resulted in unbalanced data. More recent methods started addressing this concern.

In 1997, Horton and Nakai were among the first to use a k -nearest neighbors classifier in this prediction task, where they used the method to predict the subcellular localization for 10 distinct organelles for yeast proteins (Horton and Nakai, 1997). This method was not revisited until 2004, when Huang and Li used a fuzzy k -nearest neighbors method to predict the subcellular localization of proteins over 11 distinct organelles for all eukaryotic species (Huang and Li, 2004). The k -nearest neighbors algorithm is a rather simplistic, non-parametric algorithm that, given a new protein sequence, finds k proteins in the training data that are the closest (nearest neighbors) to the new sequence, and classifies the new sequence according to the label of these k nearest neighbors. The determination of a good distance measure is essential to the success of any nearest-neighbor algorithm. To calculate the distance between two arbitrary proteins, Huang and Li converted proteins into a fixed-size vector consisting of frequencies of all amino acid pairs, and then calculated a straightforward Euclidean

distance metric. For any new sequence being classified, the algorithm first extracts the k nearest neighbor sequences (those having the smallest distance metric) and applies simple fuzzy set theory on these sequences to predict the subcellular localization. Unfortunately, they did not make their datasets easily accessible, and therefore no comparisons were made.

The PLOC method of Park and Kanehisa was an SVM method that appeared in 2003, a couple years after Hua and Sun's initial SVM study. The PLOC method used SVMs to predict 12 different subcellular localizations over all eukaryotic species (Park and Kanehisa, 2003). Their first important contribution was construction of a new, significantly larger dataset consisting of 7,589 sequences. This became a new standard dataset on which many future methods would be tested against, replacing the Reinhardt and Hubbard standard dataset. PLOC uses a one-against-rest SVM scheme, where a single SVM was built for each subcellular localization, thereby requiring 12 different SVMs. However, they also chose five different amino acid compositions, all derived from the protein sequence: single amino acid, amino acid pairs, one-gapped pairs, two-gapped pairs, and three-gapped pairs. Therefore, their model used a total of 60 SVM classifiers. The final classification was determined by a simple voting scheme between the individual classifiers.

Several other methods appeared that adopted the Park and Kanehisa dataset, which also used some variant of the SVM classification method. The LOCSVMPSI method (Xie, et al., 2005) used a position-specific scoring matrix generated from PSI-BLAST profiles to create their feature space to represent their proteins into the SVM. The CELLO II method implements a hierarchy of SVM classifiers (Yu, et al., 2006). In this method, the first layer of SVMs are individual classifiers trained on a different

representation of the amino acid sequence, which are derived from various physicochemical and structural coding schemes. The second layer implements a voting scheme between the individual classifiers to generate the final prediction. Table 3 shows the overall accuracy of these different methods using the same dataset that was used in the PLOC method. These results show that significant improvements can result from choosing a rich feature space to represent proteins.

Table 3 - Results for the Park and Kanehisa PLOC dataset

Park and Kanehisa (SVM)	LOCSVMPSI (SVM + PSI-BLAST)	Yu et al. (2-layer SVM)
78.2%	83.5%	85.0%

3.1.1 Advanced sequence-based feature spaces

Almost all of the seminal methods in this field used the most basic feature spaces to represent the protein into the classifier. Most of these methods are based on single or pair-wise amino acid composition. Using a simplistic feature space certainly reduces the computational burden on training the classifier, and on the computational resources required to accurately represent the protein. Sometimes it is the classification method itself that forces small feature spaces to be considered. For instance, the SVM is known to scale poorly on high-dimensional data. Thus, most methods have strived to extract as much information out of the sequence as possible through preprocessing of the sequence through one or several methods that transform the sequence of amino acids to a fixed-length vector of feature values.

A feature space using basic amino acid composition consists of 20 numbers, each of which represents the frequency of occurrence of a distinct amino acid in a given protein. A major limitation of amino acid composition is that it fails to incorporate any

sense of order of amino acids in the sequence. An early method that considered pair-wise and gapped-pair amino acids in a sequence was the method of Park and Kanehisa, previously discussed (Park and Kanehisa, 2003). KC Chou extended the concept of single and pair-wise amino acid composition, introducing what he has termed the *pseudo amino acid composition* (PseAA) (Chou, 2001). It incorporates the order of amino acids in a protein sequence through calculation of a correlation value of adjacent and gapped pairs of amino acids in the protein sequence. This correlation value considers various physicochemical properties of each amino acid in the pair, including hydrophobicity, hydrophilicity, and the side-chain mass of each amino acid. The majority of Chou's methods, as well as numerous other methods from colleagues and affiliates of Chou's, have capitalized on the PseAA representation, resulting in marginal improvements over prior methods.

Recent methods are beginning to emerge that allow larger feature spaces derived from sequence to be used, owing in part to the ever-expanding technological advancement of computational resources. One noteworthy method, called PSLdoc (Chang, et al., 2008), views a protein as a long string of gapped-dipeptides, which are defined as two residues separated by one or more positions. These gapped-dipeptides are weighted according to the information from derived from a position specific scoring matrix (PSSM) generated by PSI-BLAST (Altschul, et al., 1997). They reduced the feature space through application of probabilistic latent semantic analysis (PLSA), which essentially introduces a latent variable into a probabilistic model, with the aim of grouping together gapped-dipeptides that have similar distributions over the latent variable (Hofmann, 2001). The final classifier uses five distinct SVMs, each one trained for one of five localization classes considered over gram-negative bacteria. The method

was tested using the standard dataset from the popular PSORT v2.0 method (Gardy, et al., 2005), and was shown to improve the prediction of sequences that had very low homology with any other sequence in the training data. The authors postulate that the improvement in remote homologs was primarily achieved through incorporating evolutionary information inherently represented in the PSSM.

Some recent methods have considered a protein sequence as a series of wavelet transformations. These methods usually start by mapping a protein sequence to a digital signal by converting each amino acid to a unique index number, or using some numeric physicochemical property of each amino acid in the sequence, with the aim of converting the sequence into a digital signal. This signal is viewed as a waveform. Due to the non-periodic nature of the resulting waveform, it is analyzed by dividing the resulting waveform into wavelets through a wavelet transformation function. A wavelet transformation decomposes a signal into several vectors of coefficients, which contain information about various characteristics of the sequence at different scales (Lio, 2003). One recent method applied wavelets to the problem of subcellular localization, claiming improved predictive performance over existing methods, though the feature space also included other representations derived from sequence (Shi, et al., 2007a; Shi, et al., 2007b; Zhang, et al., 2007). It is not clear how much the wavelet approach played a part in improving the performance results over existing methods.

3.1.2 Non sequence-based feature spaces

Many other methods have appeared that require information beyond the primary structure for localization prediction. Some methods require phylogenetic information for classification (Marcotte, et al., 2000). The LOC3Dnet method requires prior knowledge

of evolutionary information derived from multiple alignments along with facets of protein structural information (Nair and Rost, 2003a). The LOC3Dnet classification method is particularly limiting as it requires the structural information as prior knowledge, and it only predicts over four possible localizations. Their classifier combines statistical rules along with a variety of ANNs to determine the final classification. This classifier formed part of their ensemble classifier, called LOC3D, that was used to annotate subcellular localization on proteins with known three-dimensional structures (Nair and Rost, 2003b). In addition to requiring the structure information, this method requires known nuclear-localization signals in the sequence and known Swiss-Prot keywords. A recent method by Su and colleagues incorporate information from secondary structural alignments along with biological features extracted from known translocation pathways (Su, et al., 2007). Some methods scan documents and/or GO annotations related to the proteins in their dataset in search of discriminative keywords that can be used as predictive indicators (Eskin and Agichtein, 2004; Hoglund, et al., 2006; Li and Sheng, 2007; Lu and Hunter, 2005).

Methods that require additional information beyond the protein sequence have an added benefit of making the protein representation feature space richer. As promising as that sounds, it thwarts one of the central aims of the development of methods that annotate newly discovered protein sequences based on sequence alone. It would be rare for a new sequence to have additional annotations available. However, there simply may be cases where there are no sequences with some level of similarity in the training data to the sequence being predicted. There also may be instances (as in the case with isozymes or splice variants) in which the sequences have a close homolog in the training data, and yet the localization may be different. In these cases, as some studies have noted, methods

that require information beyond the primary sequences may be beneficial (Nair and Rost, 2003a). Regardless, as stressed in the introduction, methods that only use information derived solely from the primary sequence are needed for predicted proteome-scale datasets. This research focuses on development of such methods, and thus methods requiring further information are not extensively covered beyond what is presented here.

3.1.3 Methods that predict multi-localization

It has been noted that a significant number of sequences can shuttle between or localize into multiple compartments in the cell in order to carry out their function. However, the majority of current methods in subcellular localization prediction fail to take this into consideration. One study considered 22 protein localizations in budding yeast and attempted to output information on multi-localization (Chou and Cai, 2005). Their method required prior knowledge of InterPro motifs as well as GO annotation on sequences being predicted. The PLPD method of Lee *et al.* considered the same task, using Chou and Cai's research for direct comparisons, but introduced the novel Density-induced Support Vector Data Description (D-SVDD) approach to this field (Lee, et al., 2006). The D-SVDD is a general extension of the conventional Support Vector Data Description (C-SVDD), largely inspired by the well-known SVM. The basic idea of the C-SVDD and D-SVDD methods is to find the hypersphere (a, R) with the least volume that contains the most target data, where a and R are the center and the radius of the solution, respectively. This becomes an optimization problem that focuses on minimizing an objective function subject to certain constraints, solved using similar methods to the SVM. The SVDD methods suffer the same limitations as the SVM, in that it is a one-class classification problem. The method solves this problem by creating a real-valued

function that reports the degree of which each protein belongs to each class. However, it fails to report whether any given protein is indeed multi-localized, leaving the interpretation of the ranked order of classes up to the user. Moreover, they combined general features derived from amino-acid composition in combination with the prior requirements of Chou's method.

Another method that has considered multi-localization is the CELLO method, which predicts subcellular localization of proteins for Gram-negative bacteria (Yu, et al., 2004). Both of these methods were limited to predictions of proteins in only the simplest of species in the eukaryotic and prokaryotic kingdoms, respectively. The only method known to consider multi-localization prediction on higher-order, eukaryotic species was the PSLT method (Scott, et al., 2004)..

3.1.4 Methods that predict suborganelle granularity

Experimental methods have recently advanced to allow the subcellular localization of a protein to be determined at suborganelle granularity. Data available at this level is miniscule compared to the amount of data available at the subcellular level. Computational methods are only beginning to emerge that allow the prediction at this level. The nucleus and mitochondria in eukaryotes have been the focus of recent methods attempting to achieve this level of granularity.

At least three methods have recently surfaced for subnuclear prediction. One of the first methods developed is an SVM-based method that uses a kernel based on high-scored pairs of k -peptide vectors based on BLOSUM62 scores (Lei and Dai, 2005). They subsequently refined their method by incorporating protein similarity determined through GO annotation (Lei and Dai, 2006). Shen and Chou developed a method called Nuc-

PLoc that is based on Chou's numerous pseudo amino acid composition methods, combined with evolutionary information derived from PSSMs. (Shen and Chou, 2007). The final classification in the system is the result of a majority vote from an ensemble of 310 individual *k*-nearest-neighbors classifiers. Another method was developed by Li *et al.* for subnuclear localization prediction that uses quadratic discriminant analysis over a feature space represented by Chou's pseudo amino acid composition (Li and Li, 2008).

3.1.5 Estimation of the subcellular proteome

If the proteome is defined as the entire set of proteins for a species, then the *subcellular proteome* is the entire set of proteins that localize into each of the subcellular compartments in a species. Only a handful of methods have appeared over the past few years that have the capability to estimate the subcellular proteome for a species. Because the complete subcellular proteome has not been determined for any species, there is no experimental way of determining the accuracy of their estimates, and therefore there is no means of inferring which method is the best for such wide-scale estimates. Regardless, these speculative estimates of the distribution of proteins by organelle have drawn interest by the community. Their results indicate that estimation of the proteome based on computational methods should be used in conjunction with experimental methods, until more data has been experimentally determined.

The *k*-nearest neighbors method of Huang and Li estimated the subcellular proteome for 6 different species, over 6 different organelles (Huang and Li, 2004). The BaCellLo method, an SVM based method that stressed feature improvements using a training dataset of low-redundancy (30% sequence identity), generated estimates for 5 different species, over 4 organelles (5 for plant) (Pierleoni, et al., 2006). These methods

considered limited number of organelles, and did not consider multi-localization. The only classifier that addressed these limitations was the PSLT method, which used a Bayesian network based on the combinatorial presence of known motifs (InterPro), signal peptides, and prior knowledge of specific membrane domains in human proteins (Scott, et al., 2004). The PSLT method was used to estimate the subcellular proteome of human and yeast. The method was later refined to incorporate known protein interaction data, and considered predictions at a suborganelle granularity (Scott, et al., 2005). This new method, PSLT2, concentrated less on subcellular proteome estimation, and focused more on the additional features incorporated into PSLT2. The most important such feature incorporated was protein interaction data. To explain, the method capitalized on a rather logical inference rule: For any given protein pair A and B, if the localization of A is known, and B is not known, yet it is known that A interacts with B, then one can infer that B is likely to localize where A does. While this is a sound inference rule, this again limits the training data to only those sequences that have known interaction data available. Furthermore, this method can generate a significant number of false positives on proteins that are known to multi-localize, which represent a significant fraction of the subcellular proteome.

3.2 Semi-supervised learning methods

Semi-supervised learning attempts to create a better model for classification by using both labeled and unlabeled data. Most semi-supervised learning methods are built upon existing supervised classification methods in machine learning, and aim to adjust the parameters of the underlying model to better capture information about all of the data available, including both labeled and unlabeled data. For some methods, the

representation of the unlabeled data use as input to the classifier is the same as the labeled data, though quite often the model is improved through changing the representation of the data input to the classifier by incorporating the observed structure of all of the data (Chapelle, et al., 2003; Szummer and Jaakkola, 2001; Zhu, et al., 2003).

The concept of semi-supervised learning has been around for quite some time; however, interest grew in the research community in the 1990s due to successful application in text classification and natural language problems (Chapelle, et al., 2006; Joachims, 1999; Nigam, et al., 2000). Though semi-supervised learning has been proven to be useful in a wide array of domains, its use in protein classification is slowly beginning to be recognized, with only a handful of methods being available. Recently, string kernels for protein sequence data were used with SVMs for classification of protein domains into superfamilies (Weston, et al., 2005; Weston, et al., 2004). In this work, the string kernels were enhanced through development of scalable cluster kernel techniques that allow incorporation of unlabeled protein sequence data. In related areas of bioinformatics, semi-supervised learning was used to improve the prediction of protein function by using a graph-based representation of protein interactions. Specifically, the unlabeled data was used to improve the assignment of weights among multiple graphs representing protein interaction networks (Shin, et al., 2004). The technique also has been recently applied to improving the analysis of gene expression data resulting from microarray experiments (Lu, et al., 2007).

In contrast to the inductive inference methods prevalent in supervised learning, semi-supervised learning is significantly tied to a more recent area of research known as transductive inference (Vapnik, 2006). Classification methods based on inductive inference use a set of training data to induce a model that maps the entire sample space to

a set of categories. In contrast, methods based on transductive inference focus on learning a model that can be used to classify a finite set of test data. Though it was first introduced in the mid 1990's by Vapnik (Vapnik, 1995; Vapnik, 1998), the value and potential of transductive inference went largely unnoticed until Joachims demonstrated its utility for text classification (Joachims, 1999). In this work, he introduced the Transductive Support Vector Machine (TSVM), and reported substantial improvement over standard SVMs and other inductive methods. Proponents of transductive inference state that predictive performance can be improved by reducing the learning problem to one of inferring a local function to classify a specific set of data. However, this can result in a model that fails to generalize over the entire sample space, often requiring a new model to be inferred when new data are encountered. There are very few examples of transductive inference applied in bioinformatics tasks. Krogel and Scheffer (Krogel and Scheffer, 2004) conducted an extensive study on the caveats of incorporating semi-supervised learning and transduction for predicting various functional properties of proteins corresponding to genes in the yeast genome. Kasabov and Pang used a transductive SVM for promoter recognition (Kasabov and Pang, 2004), improving predictive performance by 55% over the standard inductive SVM results.

Unfortunately, there are relatively few studies conducted that can be used for comparative analysis, owed in part to the dearth of semi-supervised-learning methods available, and partly to the nascent status of this field for protein classification. Please refer to the excellent surveys presented in (Chapelle, et al., 2006) and (Zhu, 2006), both of which present much of the current research in semi-supervised learning in a wide variety of domains.

3.3 Conclusion

The field of protein subcellular localization prediction has been an active area of research in bioinformatics for the past 15 years, and will continue to be due to the ease in which important information about function can be inferred. Despite the abundance of methods, and the current high activity in this research area, there are many areas where current methods have known limitations, providing numerous opportunities to improve on existing methods. These limitations are summarized as follows:

- Most methods predict only a limited number of subcellular organelles.
- It is known that there are a significant number of proteins that are multi-localized, meaning, they localize into more than one organelle. However, the majority of classifiers assign only one organelle for a protein.
- Most methods output only a simple true/false measure for each organelle predicted. Rarely do classifiers report a confidence or probability measure associated with the output. (This is particularly true of the SVM-based methods.)
- Prior information beyond the protein sequence is required for many of the methods reviewed, thereby limiting coverage to only those sequences that have the extra annotation available.
- For those methods that predict most of the organelles in the cell, there is a significant lack of balance between sensitivity and specificity across all classes. These methods usually perform the best on those classes that have the highest representation in the training data.
- Most methods do not consider the suborganelle level in the output. Only the main organelles are considered.
- Many existing methods use outdated datasets that make the methods untrustworthy for predicting new data.

- Many methods can not accommodate larger datasets that would be required for training and testing on proteome-wide data.
- There is a relatively large repository of protein sequences that have not been functionally annotated. The field of semi-supervised learning should be able to offer insight as to how to incorporate this data for improved classification performance.

Despite the number of methods developed recently, the classification algorithm has varied only between a few of the most popular algorithms, with the SVM being the most widely used in current research. The largest sources of variation between the methods are in the protein representation feature space and in the structure or parameter space of the classification model itself. The aim of recent methods has been to increase the number of organelles being predicted, including increasing the sensitivity for the prediction of smaller, less represented organelles, without losing the sensitivity on the large, most populated organelles.

4 ngLOC – Supervised Learning for eukaryotic proteome estimation

A new method called ngLOC – an n-gram-based Bayesian classifier – has been developed out of this dissertation. It predicts the localization of a protein sequence over ten distinct subcellular organelles. A ten-fold cross-validation result shows an accuracy of 89% for sequences localized to a single organelle, and 82% for those localized to multiple organelles. An enhanced version of ngLOC was developed to estimate the subcellular proteomes of eight eukaryotic organisms including yeast, nematode, fruitfly, mosquito, zebrafish, chicken, mouse, and human.

This work was published in Genome Biology in 2007 (King and Guda, 2007). The layout of this chapter largely follows the published material, with a more exhaustive theoretical exposition on the method included. Supplemental material was incorporated into the main text.

4.1 The ngLOC method

This section will present the materials and methods behind research completed on a Bayesian classifier for subcellular localization, called ngLOC. Bayesian predictive models have been effectively used in a variety of classification problems, including both document and protein classification tasks (Cheng, et al., 2005; Duda, et al., 2001; McCallum and Nigam, 1998; Mitchell, 1997; Witten and Frank, 2005). (The general Bayesian predictive model was presented in section 2.1.4.) The Bayesian classifier will be derived in the context of protein subcellular localization prediction, using similar notation as depicted by McCallum and Nigam (1998).

4.1.1 The n-gram model for protein representation

Letting $\mathcal{S}$ denote the feature space used to represent all proteins, we develop $\mathcal{S}$ and our predictive model in light of the significant work that has been accomplished in the field of document classification. Cheng (Cheng, et al., 2005) has shown that using document classification techniques on the primary sequence can achieve good results on classifying protein families. In a typical document classification model, $\mathcal{S}$ is constructed by considering all possible words that may appear throughout the entire set of documents. Here, we consider subsequences of a protein of fixed-length n as the equivalent of words in a document. In literature, these protein subsequences have been commonly called n -grams, n -mers, n -peptides, or simply words or subsequences of length n (Fofanov, 2004; Ganapathiraju, et al., 2002). We adopt the term n -gram. In protein classification tasks using the n -gram model, $\mathcal{S}$ is constructed by considering all possible n -grams.

Formally, we let Σ represent the set of all possible amino acids, $|\Sigma| = 20$. Given a dataset of protein sequences $\mathcal{D}$, let d_i be a sequence in $\mathcal{D}$ having k residues in length, where $d_i = (s_1 s_2 \dots s_k)$ and each $s_i \in \Sigma$. In an n -gram model, the size of the feature space grows exponentially with n , as $|\mathcal{S}| = |\Sigma|^n$. To illustrate, an integer variable typically requires 4-bytes of memory. If such a variable was used to keep track of the frequency of each of the possible 5-grams, the model would require 4 bytes * 20^5 features = 12.8 MB of memory, a 6-gram model would require 256 MB, and a 7-gram requires 5.1 GB of memory. Fortunately, for large values of n , relatively few n -grams actually occur in nature due to the evolutionary selection process that requires a delicate mixture of various amino acid combinations in a peptide to sustain a fold. A simple analysis on the entire NCBI non-redundant dataset (2.7 million protein sequences at the time of this analysis)

showed that an n -gram length as small as $n=5$ had examples that never occurred.

Therefore, to allow n -gram models for $n > 5$, we take advantage of the sparse nature of higher-order n -grams by developing a one-to-one mapping between unique n -grams and the set of integers to be used as indices, as needed. This requires memory allocation only for n -grams that occur in the training data, thereby allowing exploration of large values of n .

4.1.2 Bayesian classification for subcellular localization prediction

Given a protein sequence d_i , the aim is to develop a probabilistic model to estimate the probability that d_i is localized into organelle $y_j \in \mathcal{C}$, where $\mathcal{C}$ represents the set of all possible localizations. The classifier h predicts the class that has the highest posterior probability, given the sequence. From Eq. (2.5), this is depicted as:

$$\begin{aligned} h(d_i) &= \arg \max_{y_j} P(y_j | d_i) \\ &= \arg \max_{y_j} \frac{P(d_i | y_j) P(y_j)}{P(d_i)} \\ &= \arg \max_{y_j} P(d_i | y_j) P(y_j) \end{aligned} \tag{4.1}$$

The prior probability of each subcellular localization y_j , denoted $P(y_j)$, is estimated from $\mathcal{D}$ by counting the number of sequences assigned to class y_j , divided by the total number of sequences in $\mathcal{D}$ (see Eq. (4.2)). For the purposes of this preliminary research, the assumption will be made that the dataset $\mathcal{D}$ is drawn according to the true distribution and, therefore, does not exhibit any sample bias. This forms the estimation of the model parameter for the class prior, ϕ .

$$P(y_j) = \frac{|\mathcal{D}_{y_j}|}{|\mathcal{D}|} \tag{4.2}$$

Determining $P(d_i | y_j)$ – the likelihood of protein sequence d_i given subcellular localization y_j – is the difficult parameter to estimate. The first critical assumption made is that each sequence in the dataset is IID with respect to every other sequence. Therefore, the probability of the entire dataset is theoretically determined by the product of the individual probabilities of each sequence. Hence, determining a model for a single sequence will be analyzed.

Suppose we have an arbitrary sequence d_i drawn from D . In the context of the n -gram model, if sequence d_i has a length of k residues, then there are $k-n+1$ n -grams in d_i . Let w_t represent the t^{th} n -gram in feature space $\mathcal{X}$. The second critical assumption made is that every occurrence of n -gram w_t in d_i is independent of every other occurrence. (This implicitly states that the order in which n -gram w_t occurs in sequence d_i is ignored.) Under these assumptions, the occurrence of w_t at position j in d_i can be represented as a *Bernoulli trial*, an event with an outcome of success if w_t occurred in that position, and failure otherwise. And likewise, there are $k-n+1$ of these identical events for sequence d_i . A Bernoulli random variable is used to model each individual event. It has a single parameter θ_b , and in this context, it represents $P(w_t)$ – the probability that any given position in d_i is n -gram w_t .

Basic probability theory teaches that the *binomial distribution* can model the probability of some number of successes occurring out of some larger (or equal) number of independent Bernoulli trials (DeGroot and Schervish, 2002). Let random variable X_t have a binomial distribution, indicating the number of occurrences of n -gram w_t . If N_{it} represents the number of occurrences of w_t in sequence d_i , then the probability of the event that sequence d_i has N_{it} occurrences of n -gram w_t is given by:

$$P(X_t = N_{it}) = \binom{k-n+1}{N_{it}} P(w_t)^{N_{it}} (1 - P(w_t))^{(k-n+1)-N_{it}} \quad (4.3)$$

Since there are $|\mathcal{X}|$ possible n -grams, this implies that there are $|\mathcal{X}|$ distinct random variables X_t , each following a binomial distribution. The *multinomial distribution* is the extension of the binomial distribution, used in situations where each Bernoulli trial has m possible outcomes, not only success or failure. Here, m is equal to $|\mathcal{X}|$, as each position j in d_i has a probability of $P(x_t)$ (or θ_t) being any arbitrary n -gram w_t in $\mathcal{X}$. Furthermore, $\mathcal{X}$ is totally exhaustive, implying that the sum of the probabilities of all possible n -grams is:

$$\sum_{t=1}^{|\mathcal{X}|} P(w_t) = 1$$

We now have the theoretical basis and justification to understand how a protein sequence d_i will be converted to its feature vector $\mathbf{x}_i$ and how the distribution over the feature space will be modeled, which will give us the form for calculation of the likelihood of the data. A protein sequence will be represented as a vector of ordered n -gram occurrence events. Each entry x_{it} in vector $\mathbf{x}_i$ is equal to N_{it} , – the number of occurrences of n -gram w_t in sequence d_i . Therefore, given a vector of parameters $\boldsymbol{\theta}$, the probability of sequence d_i is estimated by the probability of the occurrence of all n -grams that occurred in sequence d_i , and is given in Eq. (4.4). Here, the model parameters $\boldsymbol{\theta}$ are explicitly listed as a conditional dependency:

$$\begin{aligned} P(d_i | \boldsymbol{\theta}) &= P(\mathbf{x}_i | \boldsymbol{\theta}) \\ &= P(X_1 = N_{i1}, \dots, X_{|\mathcal{X}|} = N_{i|\mathcal{X}|} | \boldsymbol{\theta}) \\ &= \binom{k-n+1}{N_{i1}, \dots, N_{i|\mathcal{X}|}} P(w_1)^{N_{i1}} \dots P(w_{|\mathcal{X}|})^{N_{i|\mathcal{X}|}} \end{aligned} \quad (4.4)$$

At first, this may seem intractable, as we previously showed how large the feature space can be for values of $n \geq 5$. However, N_{it} is zero for any w_t that did not occur in the data.

Therefore, the calculation is simple, taking only $O(k-n+1)$ time, which is linear with respect to the length of the sequence being determined.

Section 2.1.4 showed that $P(\mathbf{x}_i | y_j)$, which is modeling the likelihood of sequence d_i given the class y_j , needs to be determined. Specifically, separate parameters θ_j need to be determined for each class-conditional density. Upon learning these parameters, as well as the parameter φ required for the class prior according to Eq. (4.2), classification can be performed.

There were two critical independence assumptions made to make this classification problem tractable, and both can cause potential problems if the dataset deviates from these assumptions. The first assumption, that each sequence is IID with respect to the true underlying distribution is often violated, as many datasets exhibit a sample selection bias. Furthermore, the second assumption, that individual n -grams in a given sequence are IID, is also an incorrect assumption, as n -grams are clearly not independent of each other. An accurate probabilistic model based on n -grams would need to consider a joint distribution over all possible combinations of n -grams – an impossible task due to the exponential amount of memory required and because sequences vary in length. Fortunately, these assumptions have worked well in practice. With these assumptions, each sequence d_i is viewed as a collection of unordered n -grams, generated by a random process following a multinomial distribution. These independence assumptions are classically known as Naïve Bayes assumptions (McCallum and Nigam, 1998; Mitchell, 1997), and are very common when there are a large number of features modeled by random variables (such as when modeling documents in text classification problems). Figure 5 shows the derivation of the Naive Bayes classifier, and shows how sequence d_i will be classified with the most probable

class. It shows that a sequence will be classified according to the highest product of the probabilities for each n -gram that occurred in the sequence and the prior probability of that class.

$$\begin{aligned}
h(d_i) &= \arg \max_{y_j} P(y_j | \mathbf{x}_i) \\
&= \arg \max_{y_j} \frac{P(\mathbf{x}_i | y_j) P(y_j)}{P(\mathbf{x}_i)} \\
&= \arg \max_{y_j} \frac{P(\mathbf{x}_i | y_j) P(y_j)}{\sum_m P(\mathbf{x}_i | y_m) P(y_m)} \\
&= \arg \max_{y_j} \frac{\binom{k-n+1}{N_{i1}, \dots, N_{i|\mathcal{X}|}} \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_j)^{N_{it}} \right) P(y_j)}{\sum_m \binom{k-n+1}{N_{i1}, \dots, N_{i|\mathcal{X}|}} \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_m)^{N_{it}} \right) P(y_m)} \\
&= \arg \max_{y_j} \frac{\binom{k-n+1}{N_{i1}, \dots, N_{i|\mathcal{X}|}} \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_j)^{N_{it}} \right) P(y_j)}{\binom{k-n+1}{N_{i1}, \dots, N_{i|\mathcal{X}|}} \sum_m \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_m)^{N_{it}} \right) P(y_m)} \\
&= \arg \max_{y_j} \frac{\left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_j)^{N_{it}} \right) P(y_j)}{\sum_m \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_m)^{N_{it}} \right) P(y_m)}
\end{aligned}$$

Figure 5 - Derivation of multinomial Naive Bayes classifier

Because the denominator is independent of the class y_j , it can be discarded, resulting in Eq. (4.5):

$$h(d_i) = \arg \max_{y_j} \left(\prod_{t=1}^{|\mathcal{X}|} P(w_t | y_j)^{N_{it}} \right) P(y_j) \quad (4.5)$$

To prevent loss of precision in these calculations, the log of the probability is calculated, allowing a sum of each of the log probabilities to be calculated:

$$h(d_i) = \arg \max_{y_j} \left(\sum_{t=1}^{|\mathcal{X}|} N_{it} \log P(w_t | y_j) + \log P(y_j) \right) \quad (4.6)$$

The running time for classifying a new protein sequence of length k is $O(|\mathcal{C}|k)$.

4.1.3 Estimating model parameters

As discussed in section 2.1.6, the aim of a successful Bayesian classifier is to find the model parameters θ and φ that maximize the likelihood of the data, given the parameters. Estimation of the class prior parameter φ was given in Eq. (4.2). Learning θ_j , the parameters for the multinomial model representing class j , turns out to be a straightforward application of counting the occurrence of each n -gram in the data belonging to that class. To estimate $P(w_t | y_j)$, the posterior probability of n -gram w_t given class y_j , we use the LaPlace correction to prevent zero probabilities from being calculated and count the number of occurrences of w_t in all sequences belonging that class (see Eq. (4.7)).

$$\begin{aligned} \theta_{jt} &= P(w_t | y_j) \\ &= \frac{1 + \sum_{d_i \in \mathcal{D}_j} N_{it}}{|\mathcal{X}| + \sum_{d_i \in \mathcal{D}_j} (\text{Length}(d_i) - n + 1)} \\ &= \frac{1 + (\text{Count}(w_t) \text{ in class } y_j)}{|\mathcal{X}| + (\text{Total } n\text{-grams in class } y_j)} \end{aligned} \quad (4.7)$$

Figure 6 depicts the process of learning in an n -gram Naive Bayes classifier.

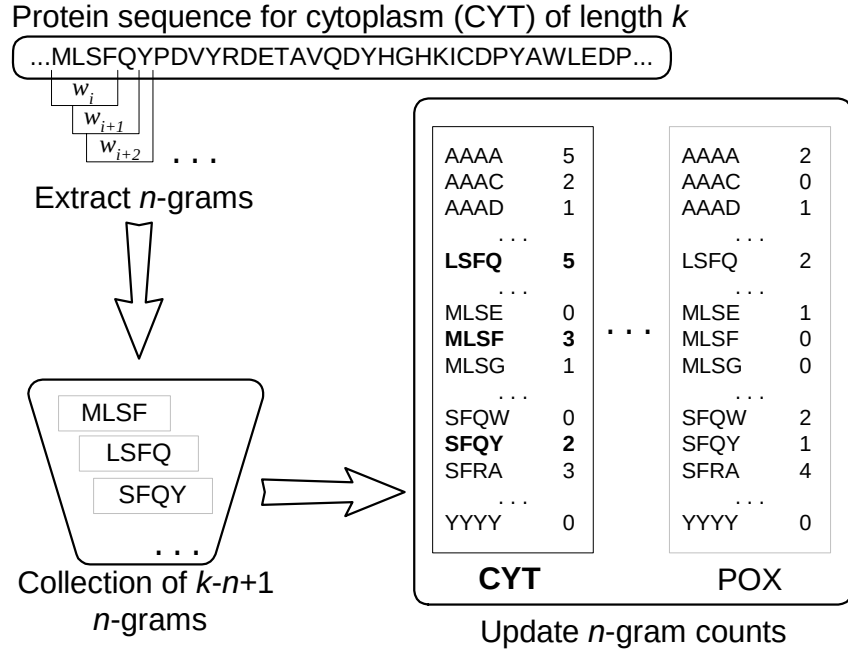


Figure 6 - The n -gram model for representing proteins in ngLOC

This figure depicts the process of extracting n -grams from an example protein sequence for cytoplasm (CYT), and shows how the table of frequencies of n -grams maintained by the model is updated accordingly. For this example, $n = 4$.

The running time for the learning phase of this classification model is bounded by the size of the training data. If the average length of a sequence in training data D is $\bar{k}$, then the running time is $O(|D|\bar{k})$. The space requirements for this model is bounded by the size of the feature space $\mathcal{X}$ as well as the number of classes being predicted, and is $O(|\mathcal{X}||\mathcal{C}|)$.

4.1.4 Probabilistic confidence score

An advantage in using a probabilistic model, such as a Naïve Bayes classifier or a Hidden Markov Model, over non-probabilistic approaches is that the probability of the model generating a given sequence is inherently reported for every class. While the prediction is based on the model with the highest probability, the probability score can also be used as a comparative measure against other classes. If the predicted class had a

low probability, it might suggest that the second or third highest predictions should also be considered. If the top two classes predicted were relatively high compared to the rest of the classes, it might suggest the possibility that the sequence is localized into both locations. We develop a probabilistic confidence score for sequence d_i for each possible class c_j , denoted $CS(d_i | c_j)$, and show how this is used to address each of the above issues. We also derive a multi-localized confidence score, denoted $MLCS(d_i)$, that gives a probabilistic measure of sequence d_i being multi-localized.

For a given sequence d_i , we define d_{null} to be a sequence of null symbols of a length that is equal to the length of d_i . (A null symbol can be any symbol s such that $s \notin \Sigma$.) Each n -gram in d_{null} is guaranteed to never occur in the model. To calculate the probability that each class generated d_{null} , we can use this fact to simplify the calculations of Equations (4.5) and (4.7), giving us:

$$P(d_{null} | y_j) = \left(\frac{1}{|\mathcal{X}| + (\text{Total } n\text{-grams in class } y_j)} \right)^{k-n+1} \quad (4.8)$$

We then set $minNullProb$ to be the minimum joint probability of d_{null} and class c_j across all classes:

$$minNullProb = \min_{c_j \in \mathcal{C}} (P(y_j) P(d_{null} | y_j)) \quad (4.9)$$

A log-odds ratio that sequence d_i is targeted for location c_j against $minNullProb$ is calculated and then normalized by dividing by the sum over all log-odds scores, to create a separate score for each subcellular location c_j for a given sequence d_i as follows:

$$CS(y_j | d_i) = \frac{\log(P(d_i | y_j) P(y_j)) - \log(minNullProb)}{\sum_m (\log(P(d_i | y_m) P(y_m)) - \log(minNullProb))} * 100 \quad (4.10)$$

The range for each score will always be between 0-100, with the sum of the scores over all classes totaling 100. This confidence score can also be interpreted as an estimate of the conditional probability of class c_j , given sequence d_i and the n -gram model used.

The multi-localized confidence score for sequence d_i (Equation (4.11)) is derived from the confidence scores of the two most probable classes for that sequence, denoted as CS_1 and CS_2 , respectively. This score is designed to give a relative measure of the likelihood that sequence d_i is multi-localized into two organelles.

$$\begin{aligned} MLCS(d_i) &= (CS_1 + CS_2) \left(100.0 - (CS_1 + CS_2) \right) \left(\frac{1}{100.0} \right) \\ &= (CS_1 + CS_2) - \frac{(CS_1^2 - CS_2^2)}{100.0} \end{aligned} \quad (4.11)$$

4.2 *ngLOC-X: Application of ngLOC to proteome-wide predictions*

Most protein classification models, including ngLOC, are built using datasets from sequences over many species. This introduces another variable that can be observed. Though some methods indirectly observe the relationship between the species and the dependent variable by incorporating phylogenetic information in their model, it is usually not directly observed. Yet, it is known that in the case of subcellular localization, the distribution between classes varies among species. For example, one study of mitochondrial proteins estimates that 9.9% of the yeast proteome is localized in the mitochondria, versus an estimated 4.8% of the human proteome (Guda, et al., 2004). Another study estimated that as much as 13% of the yeast proteome is localized in the mitochondria (Kumar, et al., 2002). It is clear that the predictions for the proteome of a

specific species can be improved if the prior probability $P(c_j)$ is known for the species being predicted. Unfortunately, this information is not available prior to classification.

We extend Equation (4.1) by introducing another random variable X that will represent the species being predicted. We will refer to the model that incorporates the proteome in this manner as ngLOC-X:

$$P(c_j | d_i, X) = \frac{P(d_i | c_j, X) P(c_j | X)}{P(d_i | X)} \quad (4.12)$$

We only need to consider the two terms in the denominator for reasons stated previously. We solve for $P(d_i | c_j, X)$ – the probability of a protein sequence, given subcellular localization c_j and species X – by assuming a mixture model of two independent conditional probability distributions over the space of protein sequences. One distribution is over proteins with known subcellular localization but unknown species, and the other distribution is over proteins of a known species but unknown subcellular localization. This forms a mixture model of two distributions, formally stated as follows:

$$P(d_i | c_j, X) = \alpha_j P(d_i | c_j) + (1 - \alpha_j) P(d_i | X) \quad (4.13)$$

The component $P(d_i | c_j)$ is estimated as given in the derivation in Figure 5. The component $P(d_i | X)$ is estimated in the same way; however, we let N_{it} be the count of the number of occurrences of n -gram w_t in the proteome for species X , and $P(w_t | c_j)$ is replaced by $P(w_t | X)$ – the probability of n -gram w_t , given species X . To estimate α , the mixture proportion for $P(d_i | c_j)$, we use the following:

$$\alpha_j = \frac{(Total\ n\text{-grams in class } c_j)}{(Total\ n\text{-grams in class } c_j) + (Total\ n\text{-grams in species } X)} \quad (4.14)$$

The mixture model allows us to consider the distribution of n -grams over an entire proteome by adjusting the probabilities of each n -gram in the training data to more accurately represent the distributions of n -grams in the proteome being classified. The result is that the distribution of $P(w_t | c_j)$ over all n -grams will be more similar to that of the proteome, while retaining the relative probabilities of each class within individual n -grams learned from the labeled data.

Solving for $P(c_j | X)$ is difficult, as we do not know the distribution of subcellular localizations in a given species, nor can it be observed prior to classification. However, we incorporate the proteome for species X using a LaPlacean-type of estimate, adding the proteome data to each class. The result is shown in Equation (4.15), where $\mathcal{D}_x$ represents the dataset consisting of sequences from the proteome of species X :

$$P(c_j | X) = \frac{|\mathcal{D}_{c_j}| + |\mathcal{D}_x|}{|\mathcal{D}| + (|\mathcal{C}| |\mathcal{D}_x|)} \quad (4.15)$$

This method can cause the probability estimates for $P(c_j | X)$ to become more uniform in proportion to the size of the dataset of the proteome ($\mathcal{D}_x$) being considered. We accept this tendency, as it implicitly factors in a measure of uncertainty proportional to the size of the proteome being considered – meaning, the larger the proteome, the more uncertainty there is in regard to the exact prior probabilities of each subcellular localization. In this case, the priors should not be based solely on the exact percentages of the ngLOC training data.

4.3 *ngLOC data*

The dataset used for this task is a set of protein sequences taken from the Swiss-Prot database, release 50.0 (Boeckmann, et al., 2003) containing experimentally

determined annotations on subcellular localization. We have applied the following filters to obtain high-quality data for testing and training our program: (i) only eukaryotic, non-plant sequences were considered, (ii) sequences with predicted or ambiguous localizations were removed, (iii) sequences shorter than 10 residues in length were removed, (iv) all redundant sequences were removed, and (v) sequences known to localize in multiple locations were manually checked and sorted to avoid errors caused by automated keyword-based sorting. The final set of training data consisted of a set of 28,056 sequences. Of these, 2,169 sequences were annotated with two distinct subcellular localizations, of which 1,120 of them were localized to both cytoplasm and nucleus. Location-wise distribution of this dataset is shown in Table 4. This dataset contains proteins from 1,923 different species in the eukaryotic kingdom.

Table 4 - Distribution of proteins over eukaryotic subcellular locations

Organelle	Code	Number of sequences
Cytoplasm	CYT	2,884
Cytoskeleton	CSK	248
Endoplasmic Reticulum	END	939
Extracellular	EXC	7,536
Golgi Apparatus	GOL	282
Lysosome	LYS	166
Mitochondria	MIT	2,442
Nuclear	NUC	4,658
Plasma Membrane	PLA	6,530
Perioxosome	POX	202
2 Localizations annotated		2,169
TOTAL:		28,056

4.3.1 Proteome datasets

We downloaded the proteomes of eight eukaryotic model organisms from the Integr8 database (Kersey, et al., 2005), which include *Saccharomyces cerevisiae* (yeast),

Caenorhabditis elegans (worm), *Drosophila melanogaster* (fruitfly), *Anopheles gambiae* (mosquito), *Danio rerio* (zebrafish), *Gallus gallus* (chicken), *Mus musculus* (mouse) and *Homo sapiens* (human). Because 25-40% of these proteomes are hypothetical proteins (with the exception of yeast), we separated the curated proteome subsets containing annotation for at least one of the three GO concepts, but not including those with GO evidence codes – ND (no biological data available), RCA (reviewed computational analysis), and NAS (non-traceable author statement).

4.4 Performance measurements

For each subcellular localization class y_j , the following counts are tracked:

- True positives (TP_j) – the number of sequences correctly predicted to localize into y_j .
- True negatives (TN_j) – the number of sequences correctly predicted to not localize into y_j .
- False positives (FP_j) – the number of sequences predicted to localize into y_j , but actually localize elsewhere.
- False negatives (FN_j) – the number of sequences that actually localize into y_j , but were predicted to localize elsewhere.

Using these four quantitative measures, we report a number of standard measurements in judging classifier performance:

- *Overall Accuracy* – a measure of the overall classifier performance. It is defined as the fraction of the data tested that is classified correctly. Though it is a poor measure to consider on highly unbalanced datasets, it is still reported it as a general overall comparative measure:

$$Accuracy = \frac{\sum_j TP_j}{|D|} \quad (4.16)$$

- *Sensitivity* (a.k.a. *Recall*, *TP-rate*) – the fraction of data in class y_j that was correctly predicted. This gives a measure of individual class accuracy. Poor sensitivity indicates that the classifier is under-predicting class y_j :

$$Sens_j = \frac{TP_j}{TP_j + FN_j} = \frac{TP_j}{|D_{y_j}|} \quad (4.17)$$

- *Precision* – the fraction of data predicted to be in class y_j that was actually correct. Poor precision indicates that the classifier is over-predicting class y_j :

$$Prec_j = \frac{TP_j}{TP_j + FP_j} \quad (4.18)$$

- *Specificity* – the fraction of data *not* in class y_j that was correctly predicted:

$$Spec_j = \frac{TN_j}{TN_j + FP_j} = \frac{TN_j}{\sum_{k \neq j} |D_{y_k}|} \quad (4.19)$$

- *False Positive Rate* – the fraction of data *not* in class y_j that was incorrectly predicted to be in class y_j :

$$FPR_j = \frac{FP_j}{TN_j + FP_j} = \frac{FP_j}{\sum_{k \neq j} |D_{y_k}|} = 1.0 - Spec_j \quad (4.20)$$

- *Matthews Correlation Coefficient* (Matthews, 1975) – a measure of overall predictive performance for class y_j . It has a range of $(-1, +1)$, where 1 implies a perfect prediction, 0 implies random, and a value of less than 0 implies that the prediction is worse than random guessing:

$$MCC_j = \frac{TP_j * TN_j - FP_j * FN_j}{\sqrt{(TP_j + FP_j)(TP_j + FN_j)(TN_j + FP_j)(TN_j + FN_j)}} \quad (4.21)$$

4.5 Results

To evaluate the performance of the method, we apply a standard validation technique called ten-fold cross validation, where sequences from each class are divided into ten parts – the model is built using nine parts, and predictions are generated and evaluated on the data contained in the remaining part. This process is repeated for all ten possible combinations. We report standard performance measures over each subcellular location, including sensitivity (recall), precision, specificity, false positive rate (FPR), Matthews Correlation Coefficient (MCC), and Receiver Operating Characteristic (ROC) curves. For a measure of the overall classifier performance, we report overall accuracy as the fraction of the data tested that was classified correctly. To demonstrate the usefulness of our probabilistic confidence measures, we show how these measures can be used to consider situations where a sequence may have multiple localizations, as well as to consider alternative localizations when confidence is low.

4.5.1 Evaluation of different size n-grams

The optimal value of n for the n -gram model was determined by evaluating the predictive performance of ngLOC over different size n -gram models up to 15-grams. For this test, only single-localized sequences were used, and the minimum allowable length sequence was 15 residues, to enable testing of models up to 15-grams. The results show that the 7-gram model had the highest performance, with an overall accuracy of 88.43%. However, both the 6-gram and 8-gram models are close, with accuracies of 88.12% and

87.53%, respectively Figure 7. The results reported in the remainder of this paper use the 7-gram model unless otherwise stated.

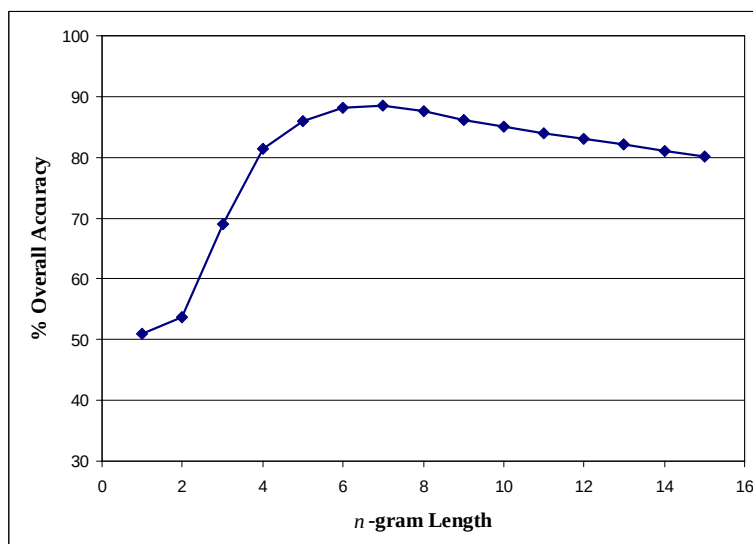


Figure 7 - Overall accuracy versus n-gram length

This graph shows how different values of n affect the overall accuracy of ngLOC on the standard dataset, based on a 10-fold cross validation.

4.5.2 Prediction performance using a 7-gram model

All of our tests are based on the standard ngLOC dataset (detailed in the methods section), which was selected with a minimum sequence length of 10 residues allowed. We ran a test using only single-localized sequences, as well as the entire dataset including multi-localized sequences. For a 7-gram model, the overall accuracy of both models on single-localized sequences only was 88.8% and 89%, respectively. The results for the model built using the entire dataset is shown in Table 5, and will be the model of choice because it will enable prediction of multi-localized sequences as well.

Table 5 – Results for 7-gram model using entire dataset

Location	Code	Precision	Sensitivity	FPR	Specificity	MCC
----------	------	-----------	-------------	-----	-------------	-----

Cytoplasm	CYT	0.828	0.775	0.020	0.980	0.777
Cytoskeleton	CSK	0.882	0.452	0.001	0.999	0.629
Endoplasmic Reticulum	END	0.961	0.789	0.001	0.999	0.867
Extracellular	EXC	0.949	0.939	0.021	0.979	0.921
Golgi Apparatus	GOL	0.891	0.550	0.001	0.999	0.697
Lysosome	LYS	0.953	0.855	0.000	1.000	0.902
Mitochondria	MIT	0.964	0.799	0.003	0.997	0.867
Nuclear	NUC	0.807	0.906	0.048	0.952	0.821
Plasma Membrane	PLA	0.883	0.958	0.043	0.957	0.892
Peroxisome	POX	0.938	0.748	0.000	1.000	0.836
Single-Localized % Overall Accuracy:						89.03
Multi-Localized % Overall Accuracy (At Least 1 Correct):						81.88
Multi-Localized % Overall Accuracy (Both Correct):						59.70

The performance results of ngLOC on a 10-fold cross-validation are displayed. The overall accuracy is also reported for multi-localized sequences, comparing at least one localization predicted correctly against both localizations predicted correctly.

Referring to Table 5, precision is high across all classes (0.81-0.96), while sensitivity ranged between 0.75 to 0.96, with the exception of golgi (GOL, 0.55) and cytoskeleton (CSK, 0.45), which is likely due to low representation in the dataset. Though CSK and GOL had the lowest sensitivity, their precision was very good, which is typical when a class is under-predicted. Specificity is very high across all classes (0.95-1.0), though the classes with the largest representation in the dataset – EXC, PLA, NUC, and CYT – had the lowest specificity, which is typical for highly represented classes that are often prone to over-prediction. Regardless, the MCC values of these three classes were still between 0.82-0.92. On the other end are the classes with the smallest representations in the dataset, including LYS, POX, CSK and GOL, whose MCC values range between 0.63-0.90. Surprisingly, LYS and POX, the two classes with the smallest representation in the dataset, had good MCC values (0.902 and 0.836 respectively). We determined the percentage of *n*-grams that were unique (occurred in only one organelle) in each of these four organelles (LYS, POX, CSK, GOL) and discovered that LYS and POX had the highest percentage of unique *n*-grams with respect to the total number of *n*-

grams in the organelle (data not shown). This suggests that the proteins in these locations are highly specific and distinctive compared to those proteins localized elsewhere, and could explain the superior performance of these locations despite having the smallest representation in the training dataset. We also observed that n -grams in CSK and GOL had the lowest percentage of unique n -grams compared to any other class in the data, suggesting that n -grams in these organelles are more likely to be in common with n -grams in other organelles, and therefore the proteins in these organelles will be difficult to predict. The remaining classes performed well, with MCC values ranging between 0.78-0.87.

A receiver operating characteristic (ROC) curve depicts the relationship between specificity and sensitivity for a single class. The ROC curve for the perfect classifier would result in a straight line up to the top left corner, and then straight to the top right corner, indicating that a single score threshold can be chosen to separate all of the positive examples of a class from all of the negative examples. Figure 8 shows the ROC curve for each class in ngLOC. Each point in the curve is plotted based on different confidence score thresholds. For all classes except CYT and NUC, the ROC curves remain very close to the left side of the chart, primarily because the majority of classes have very high specificity at all confidence score thresholds. This is a desirable characteristic of ROC curves. Though PLA and MIT have a high rate of false positives at the lowest score thresholds, the rate of true positives remains high, indicating that a good discriminating threshold exists for these classes. CYT has a high rate of false positives for lower score thresholds, again confirming that CYT is a class prone to over-prediction. This is also confirmed by its low precision (0.828). The other class that is prone to over-prediction is NUC, exhibiting the lowest precision of all 10 classes (0.807). NUC has the

lowest specificity as well. This is likely a result of the characteristics of the short nuclear localization signals (NLSs) that exist on nuclear proteins. These NLSs can vary significantly between species. The ngLOC method, which uses a 7-gram peptide to explore the protein sample space along the entire length of the protein, is likely discovering many of these NLSs in the nuclear sequences. Because the dataset contains many examples of nuclear proteins among many species, many candidate NLSs will be discovered, thereby leading to over-prediction of nuclear proteins.

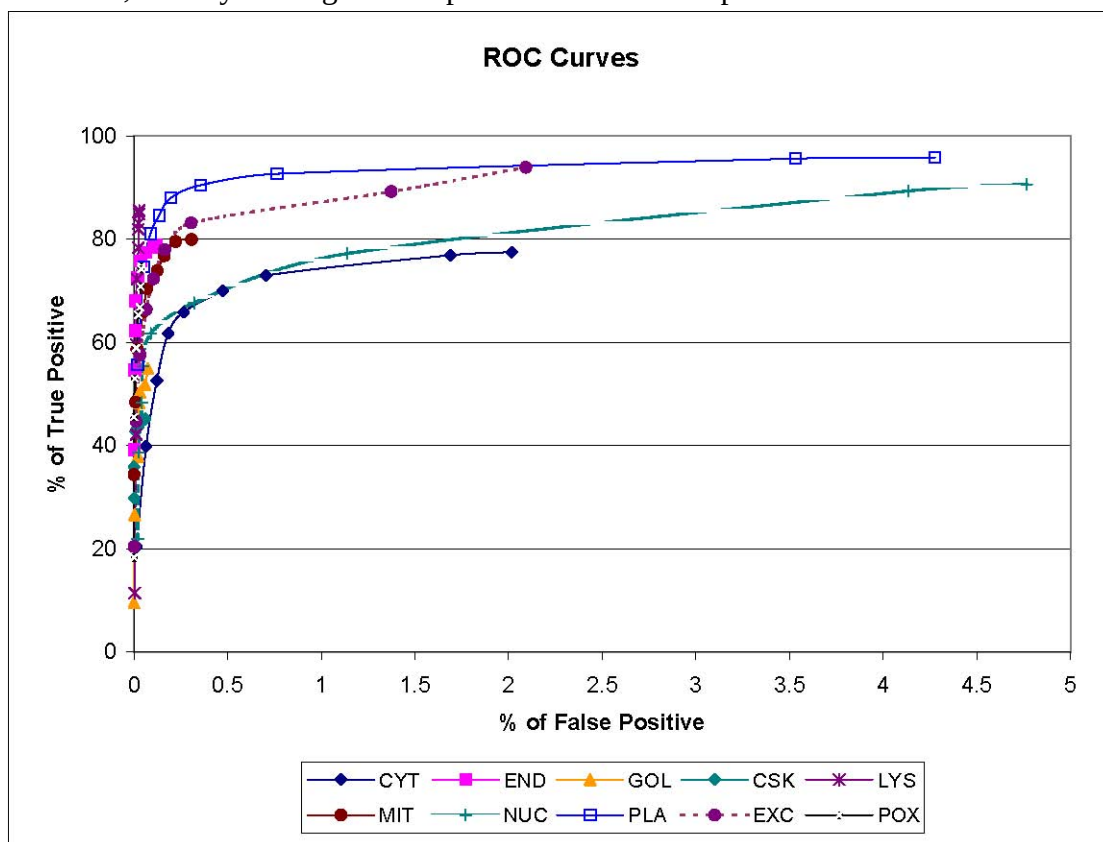


Figure 8 - ROC curve for 7-gram model

A plot of the receiver operating characteristic (ROC) curve for each class is shown. A typical ROC would have the x-axis plotted to 100%. We plot only up to 5%, to reduce the amount of overlap in the individual class plots along the y-axis and to improve clarity. Because the minimum specificity is 0.952, plotting up to 5% is a sufficient maximum for the x-axis.

To obtain the sensitivity for multi-localized sequences, we consider two types of true positive measures: (i) at least one of the two localizations had the highest probability, and (ii) both localizations had the top two probabilities. The overall accuracy of at least

one localization being correctly predicted was 81.88%, and for both localizations being correctly predicted was 59.7%. When considering the accuracy of both localizations being predicted to be within the top three most probable classes, the accuracy increased to 73.8%, suggesting the usefulness of this method in predicting multi-localized sequences.

4.5.3 Analysis of 7-gram distribution

An analysis of the occurrence of 7-grams was performed to better understand the distribution of those n -grams that were actually useful, discriminatory features. In the ngLOC dataset, there were a total of 7,316,401 unique 7-grams. Table 6 displays the actual number of 7-grams that occurred for different frequencies of occurrence. It shows that 78.5% of these n -grams occurred only once in the dataset. Therefore, only 1.6 million of the 7.3 million n -grams can potentially be useful in a 10-fold classification. To explain, in any type of cross validation, a feature needs to occur at least twice in the dataset to have the *potential* to be a useful feature for classification. If a feature occurs only once, then it will either be in the test set, or the training set, and in either case, it is guaranteed to not be used in the validation. This table also shows that only 0.6% of the 7-grams occur more than 10 times. These are likely to be n -grams that are occurring across multiple classes, or those that are occurring from many highly homologous sequences in the same class. The latter situation is the ideal one, as these will result in the most discriminatory features. Regardless, these results show that there can be a significant reduction in the size of the feature space, and may have a potential of allowing better generalization on more data, by allowing the most informative features to be selected on a 5-gram or 6-gram model.

Table 6 - Distribution of 7-grams in ngLOC dataset

# of occurrence	# 7-grams	% of occurrence
1	5745619	78.53
2	899249	12.29
3	321795	4.40
4	147130	2.01
5	68190	0.93
6	37054	0.51
7	24902	0.34
8	16963	0.23
9	11802	0.16
>= 10	43697	0.60
Total	7316401	100.00

A second analysis was done on the location-wise distribution of 7-grams that occurred in only one organelle, defined as *class-unique n-grams*. Class-unique *n-grams* have the most discriminatory power in a typical *x*-fold cross validation, provided they occur at least two times in the dataset. This analysis was performed through calculation of the *class-unique odds ratio* – the odds that a class will be using features that are unique to it when classifying the data. This measure is calculated by taking the fraction of class-unique *n-grams* that occurred at least twice in each class, divided by the fraction of all *n-grams* that occurred in each class. In this calculation, a value greater than one indicates that the class was more likely to have *n-grams* that were unique to that class, and therefore prediction of sequences belong to that class is more likely to perform well on any *x*-fold cross validation technique. The results are displayed in Table 7.

Table 7 - Class-unique odds ratio for 7-gram model

Class	Class-unique odds ratio
CYT	0.80
CSK	0.63
END	1.13

EXC	1.08
GOL	0.70
LYS	1.08
MIT	0.93
NUC	0.71
PLA	1.36
POX	0.90

The results coincide precisely with the overall performance from the 10-fold cross validation presented in Table 5. The four classes with the lowest predictive performance as measured by the MCC value (CYT, CSK, GOL, NUC) all had the lowest class-unique odds ratios. From these results, one can conclude that n -grams in these four classes had the lowest percentage of unique n -grams compared to any other class in the data, suggesting that n -grams in these organelles are more likely to be in common with n -grams in other organelles. Therefore the proteins in these organelles will be difficult to predict. This is particularly likely with sequences targeted for the nucleus, which depend on short nuclear localization signals (NLSs) in the sequence for proper targeting to occur. These NLSs can vary significantly between species, and using a 7-gram model is likely discovering many of these NLSs; however, because nuclear sequences have a high representation in the dataset among many species, many candidate NLSs will be discovered, thereby leading to very few unique n -grams in the NUC class, and leading to significant over-prediction. The low class-unique odds ratio for CYT and NUC can be explained by the fact that the largest number of multi-localized sequences are those co-localized into both the CYT and NUC, which occupy 50% of all of the multi-localized sequences. During training, these sequences will be fed into both models for the CYT and NUC classes. It is also interesting to note that LYS and POX, which are the two smallest classes in the data (occupying only 1.4% of the entire dataset), had high numbers of unique n -grams, which contributed to the solid performance on these small classes.

Relatively high class-unique odds ratios can possibly suggest that the proteins in these locations are highly specific and distinctive compared to those proteins localized elsewhere, and could explain the superior performance of the classification of these locations despite having the smallest representation in the training dataset.

4.5.4 Evaluation of the confidence score

A probabilistic confidence measure is an important part of any predictive tool, as it puts a measure of credibility on the output of the classifier. Table 8 demonstrates the utility of our confidence score (CS, ranges from 0-100) in judging the final prediction for each sequence. We found that a score of 90 or better was given to 37.5% of the dataset with an overall accuracy of 99.8% in this range. About 86% of the dataset had a confidence score of 30 or higher. Though the accuracy of sequences scoring in the 30-40 range was only 70.1%, the cumulative accuracy of all sequences scoring 30 or higher was 96.2%. We found that the overall accuracy of the classifier proportionally scaled very well across the entire range of confidence scores.

Table 8 – Benchmarking the performance of ngLOC (7-gram) against its confidence score

Confidence Score:	0	10	20	30	40	50	60	70	80	90
% of Dataset:	0.0	2.4	11.8	6.1	4.4	4.5	5.8	9.3	18.1	37.5
% Overall Accuracy:	0.0	56.2	41.4	70.1	88.3	93.0	97.0	98.1	99.2	99.8
Cumulative % of Data:	100.0	100.0	97.6	85.7	79.6	75.2	70.7	64.9	55.6	37.5
Cumulative % Overall Accuracy:	88.8	88.8	89.6	96.2	98.3	98.8	99.2	99.4	99.6	99.8

This table shows how the confidence score associated with each prediction relates to the overall accuracy. The higher the score, the more likely the prediction is to be the correct one. For example, all sequences scoring 90 or better had an accuracy of 99.8%. About 80% of the dataset was scored 40 or higher with a cumulative accuracy of 98.3%.

In Table 8, we presented the performance of ngLOC under the restriction that the correct localization for a given sequence was predicted as the top most probable class. To understand how close ngLOC was on misclassifications, we expanded our true positive measure by considering correct predictions within the top four most probable classes. As shown in Table 9, for single-localized sequences, the overall accuracy jumped from 88.8% to 94.5% when the correct prediction is considered within the top three most probable classes. While this improved accuracy has no meaning for single-localized sequences, it indicates that the majority of misclassifications were missed by a narrow margin. For multi-localized sequences, the classifier predicted both correct localizations as the top two most probable classes 59.7% of the time; however, the classifier predicted both correct localizations within the top three or four classes with accuracies of 73.8% and 83.2%, respectively. We also considered the accuracy of only those sequences localized into both the cytoplasm and nucleus (CYT-NUC), because they represent 51.6% of our set of sequences with two localizations. As expected, the accuracy increased, with at least one correct localization predicted within the top three with an accuracy of 99.5%, and both localizations predicted at an accuracy of 96.3% in the top four most probable classes. The high performance for sequences localized to both CYT and NUC is partly attributed to the fact that this combination of organelles has the largest representation out of all multi-localized sequences in the dataset (1,120 out of 2,169).

Table 9 - Rank of correct class single- and multi-localized sequences using a 7-gram model

Rank of Correct Class:	1	2	3	4
Single Loc. Only:	88.8	92.2	94.5	96.3
CYT-NUC, 1 Correct:	88.2	96.1	99.5	100.0
CYT-NUC, Both Correct:		66.5	82.9	96.3
All Multi-Loc., 1 Correct:	81.9	92.0	96.1	97.4

All Multi-Loc., Both Correct: **59.7** 73.8 83.2

This table shows the percent of the data that had the correct localization predicted within the top r most probable classes, where r is the rank of the correct class. The items in bold represent the overall accuracy of ngLOC on those sequences specified.

4.5.5 Evaluation of the multi-localized confidence score (MLCS)

It is known that a significant number of sequences in eukaryotic proteomes are localized to multiple subcellular locations; a predominant fraction of such sequences shuttle between or localize to both the cytoplasm and nucleus. To differentiate single-localized sequences from those that are multi-localized, a multi-localized confidence score (MLCS) was developed. The MLCS was evaluated on the entire dataset, and the accuracy was considered on multi-localized sequences over different MLCS thresholds. For accuracy assessment in this test, a prediction is considered to be a true positive if both correct localizations are the top two most probable classes – the most stringent requirement possible. As shown in Table 10, 76% of the multi-localized sequences scored an MLCS of 40 or higher, while 81% of the single-localized sequences have MLCS scores under 40. More than 20% of multi-localized sequences received a score of 90 or better, versus only 0.2% of single-localized sequences in this range. Multi-localized sequences in this range had both localizations correctly predicted 98.7% of the time.

Table 10 - Evaluation of MLCS against single- and multi-localized sequences

MLCS:	0	10	20	30	40	50	60	70	80	90
% of Single-Loc. Data:	25.9	21.2	12.6	21.1	13.6	3.1	1.2	0.6	0.4	0.2
Cumulative %, Single-Loc. Data:	100.0	74.1	52.9	40.3	19.2	5.6	2.4	1.2	0.6	0.2
% of Multi-Loc. Data:	1.7	2.1	2.3	17.9	26.2	7.8	6.2	5.3	10.0	20.5
Overall Acc., Multi-Seq. Only:	36.1	45.7	46.9	20.3	34.5	63.3	83.7	86.2	94.4	98.7
Cumulative %, Multi-Loc. Data:	100.0	98.3	96.2	94.0	76.0	49.8	42.0	35.8	30.5	20.5

Cumulative Acc., Multi-Seq. Only: 59.7 60.1 60.4 60.7 70.3 89.1 93.9 95.6 97.3 98.7

This table shows the percentage of the dataset that resulted in different ranges of the MLCS, as well as the overall accuracy and cumulative accuracy of multi-localized sequences in that range.

These results are very promising considering that multi-localized sequences comprise less than 10% of the entire dataset. In general, the higher the MLCS, the more likely the sequence is to not only be multi-localized, but also to have both correct classes as the top two predictions. Table 11 shows examples of the MLCS and confidence scores output by ngLOC for a few multi-localized sequences.

Table 11 - Examples of prediction for multi-localized sequences

Name	Correct	MLCS	CYT	END	GOL	CSK	LYS	MIT	NUC	PLA	EXC	POX
TAU_MACMU	CYT/PLA	98.2	49.1	0.2	0.1	0.1	0.0	0.3	0.6	49.2	0.3	0.1
CTNB1_MOUSE	CYT/NUC	85.1	49.8	0.1	0.0	0.0	0.0	0.1	42.2	7.5	0.2	0.0
3BHS2_RAT	END/MIT	97.9	0.4	48.9	0.2	0.1	0.0	49.1	0.3	0.4	0.4	0.1
SIA4A_CHICK	GOL/EXC	85.0	2.4	1.8	42.4	0.6	0.0	1.8	2.5	4.6	43.7	0.2
GGH_HUMAN	LYS/EXC	69.1	4.4	3.1	2.1	2.0	33.7	3.2	5.9	5.4	39.9	0.3

This table presents examples of multi-localized sequences predicted with a high MLCS value. The "name" column represents Swiss-Prot entry names. The "correct" column shows both organelles in which the sequence is localized into. The remaining columns show the confidence score for each possible localization.

4.5.6 Comparison against other methods

We evaluated the performance of ngLOC by comparing it with that of existing methods. Comparisons were made in three ways: (i) by using the ngLOC dataset to train and test other methods, (ii) by testing ngLOC on another dataset, and (iii) by training *and* testing ngLOC on another dataset.

For the first test, ngLOC was compared against two existing methods – PSORT (Nakai and Horton, 1999) and pTARGET (Guda and Subramaniam, 2005). Both of these methods were selected because they are widely used by the research community, can predict 10 or more subcellular locations, and are freely available for offline analysis. For

uniformity, a random selection of 80% of the ngLOC dataset was selected for training, and testing was done on the remaining 20%. The overall accuracies of PSORT, pTARGET, and ngLOC are 72%, 83%, and 89%, respectively. The MCC values from each classifier were used as the comparative measure, as it is the most balanced measure of performance for each individual class. Figure 9 compares the MCC values on each of the 10 classes for all three methods. The ngLOC classifier showed a respectable improvement across all locations over PSORT and pTARGET, with the exception of pTARGET's accuracy on NUC, which had a slightly higher MCC than ngLOC. In particular, ngLOC showed a significant improvement in all of the classes that had the smallest representation in the dataset (CSK, END, GOL, LYS, and POX) which are typically difficult to predict.

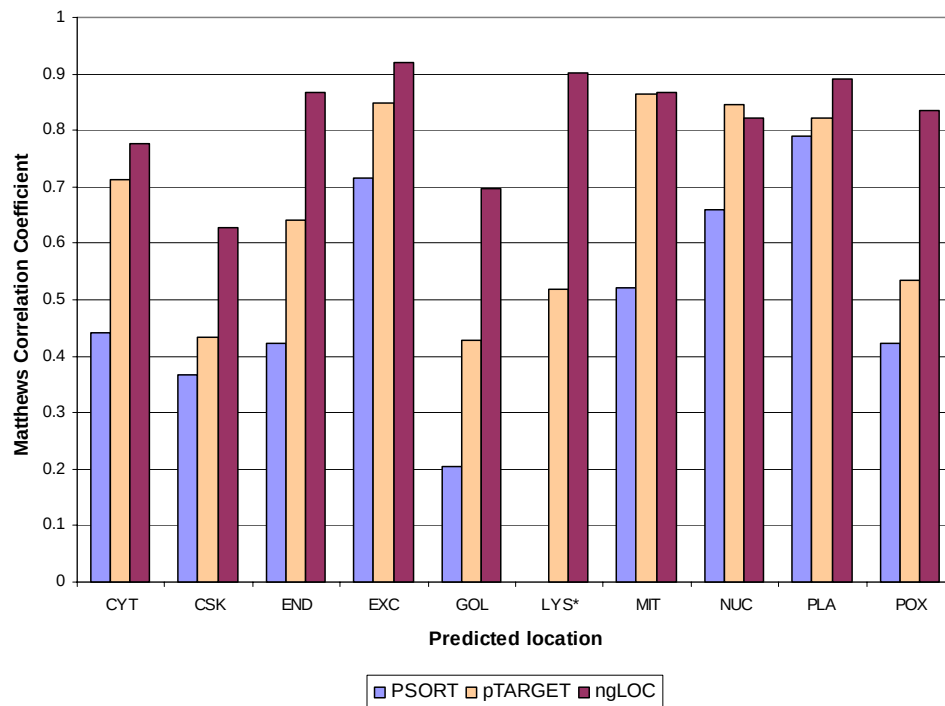


Figure 9 – Comparison of predictions from three methods on the ngLOC dataset

*LYS location was omitted from PSORT predictions, as PSORT predicts this class as part of the vesicular secretory pathway.

For the next comparative test, we found a similar dataset that has been used by the research community – PLOC (Park and Kanehisa, 2003). The PLOC method is an SVM classification method whose feature space is constructed from various facets of amino acid composition. (See section 3 for more details on the PLOC method.) The primary differences between the ngLOC and PLOC dataset is the version of the Swiss-Prot repository from which the sequences were acquired, the level of sequence identity assumed in the dataset, and the multi-localized annotation in the ngLOC dataset. Sequences with up to 80% identity were allowed in the PLOC dataset, while all sequences with less than 100% identity were allowed in the ngLOC dataset. Sequences from their dataset that are localized into the chloroplast and vacuole were disregarded, as ngLOC does not consider plant sequences. We built both a 6-gram and a 7-gram model using the entire ngLOC dataset, and used their dataset for testing purposes. The ngLOC method had overall accuracies of 88.04% and 85.64%, respectively, both of which compared favorably with the 78.2% overall accuracy reported by PLOC. It is important to note that the optimal value of n in ngLOC is dependent on the amount of redundancy in the data being tested. A 6-gram model performed better than a 7-gram model, which confirms the lower redundancy in the PLOC dataset compared to the ngLOC dataset. We observed that there were some predictions with a CS ≥ 90 , yet were misclassified by ngLOC. All sequences predicted with this level of confidence that were misclassified by ngLOC were due to incorrect annotation, likely due to the PLOC dataset being outdated. Table 12 shows some example predictions. Each one was verified in the latest Swiss-Prot entry to match the ngLOC prediction.

Table 12 - Examples of outdated annotations in the PLOC dataset discovered by ngLOC

SP ID	PLOC	Correct	ngLOC	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
-------	------	---------	-------	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Anotation			Predict										
P40541	CYT	NUC	NUC	1.2	0.2	0.5	0.7	0.3	0.0	0.4	96.1	0.5	0.0
Q03562	PLA	CYT	CYT	95.5	0.3	0.5	1.0	0.3	0.0	0.7	0.9	0.7	0.1
P32522	PLA	MIT	MIT	0.8	0.1	0.3	0.9	0.1	0.2	95.5	1.4	0.8	0.0
P33287	CSK	CYT	CYT	94.9	0.2	0.7	0.9	0.2	0.0	0.7	0.9	1.3	0.1
P53730	PLA	END	END	1.1	0.2	94.5	0.8	0.2	0.0	0.8	1.0	1.4	0.1

This table shows some examples of sequences in the PLOC dataset that had outdated annotations, that were discovered by ngLOC.

We also found instances where some of the predictions misclassified by ngLOC were actually multi-localized and should have been considered correct as well (see Table 13).

Table 13 - Examples of multi-localized sequences in the PLOC dataset discovered by ngLOC

PLOC			ngLOC											
SP ID	Anotation	Correct	Predict	MLCS	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
P97840	CYT	CYT/EXC	EXC	90.0	45.0	0.5	1.0	45.4	0.5	0.0	1.4	3.5	2.7	0.1
P70583	CYT	CYT/NUC	NUC	74.2	36.3	0.2	0.5	1.1	0.2	0.0	15.5	45.1	1.0	0.1
P01095	CYT	CYT/NUC	NUC	87.8	43.9	0.8	1.6	3.2	0.9	0.0	2.3	44.1	3.0	0.2

Our performance results are without correcting any annotations in the PLOC dataset. We believe that updated annotations in the PLOC dataset, as well as updated that label multi-localized sequences, would further improve the accuracy of ngLOC on the PLOC dataset.

For the final comparative test, ngLOC was modified to predict 12 distinct classes, thereby allowing the complete PLOC dataset (with original annotations and all 12 localizations) to be used for both training and testing on our method, using a 10-fold cross validation for performance analysis. On a 6-gram model, the overall accuracy was 82.6%, which again compared favorably with PLOC's accuracy of 78.2%. We found numerous misclassifications that had a correct second-highest prediction. Table 14 shows some examples of these predictions.

Table 14 - Example misclassifications of ngLOC method when training and testing on PLOC dataset

SP ID	PLOC	ngLOC												
	Annotation	Predict	CHL	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX	VAC
P81649	LYS	EXC	2.7	1.4	3.4	3.3	37.7	3.4	32.5	4.1	1.5	3.4	3.3	3.4
P21954	MIT	CYT	15.6	37.7	1.8	1.9	1.7	1.4	1.3	31.0	2.5	2.2	1.5	1.4
P51242	CHL	MIT	34.3	4.1	1.9	1.9	2.0	1.9	2.3	37.3	5.5	4.2	2.0	2.7
P05123	CYT	MIT	5.5	16.9	4.4	3.8	3.9	3.9	5.8	39.4	2.8	6.0	3.7	3.9
P30412	CYT	END	14.7	22.5	2.1	27.0	9.3	1.6	2.5	8.6	3.3	4.7	1.6	2.1

This table shows examples of sequences the ngLOC method misclassified, using a 6-gram model trained on the PLOC dataset, using all 12 possible localization classes, as annotated in the PLOC data. For each sequence, the top two predictions are in bold. This shows how the confidence score can be used to consider possible alternatives when the confidence score is low, as the correct prediction was the second highest.

In fact, out of 12 possible classifications, ngLOC predicted the correct localization to be within the top two most probable classes 88.7% of the time. It is interesting to note that even in this test, sequences were found that were misclassified according to PLOC annotations, yet the prediction by ngLOC was consistent with the latest release of Swiss-Prot (Swiss-Prot:P40541, Swiss-Prot:P33287). We also noted instances where the sequence is multi-localized, and ngLOC predicted the location that was not annotated in the PLOC dataset (e.g., Swiss-Prot:P40630, Swiss-Prot:P42859). These results underscore the robustness of our method and usefulness of its confidence score, as we were able to (i) identify outdated annotations in the PLOC dataset, (ii) identify potential multi-localized proteins in data not annotated accordingly, and (iii) consider alternate localizations beside the predicted class when the confidence score is low, suggested by the high accuracy when considering the top two classifications.

4.5.7 Evaluating ngLOC-X for proteome-wide predictions

We extended the core ngLOC method to allow classification of proteins from a single species. We call this method ngLOC-X, which is based on the model depicted in Equation (4.12). Assessing the performance of ngLOC-X proved challenging, as only a

small percentage of each proteome has subcellular localizations annotated by experimental means, and therefore it is impossible to infer an exact accuracy measurement on proteome-wide predictions. However, subsets of these proteomes are represented in the ngLOC dataset, and thus performance analysis can be inferred from these subsets. We chose two species for performing extensive analysis – mouse (3,596 represented sequences out of 23,744), and fruitfly (753 represented sequences out of 9,997). (Human had the largest set, with 5,945 represented sequences; we did not test this subset due to the size of the data that would need to be removed from the core ngLOC data.) For each species, we extracted the represented protein sequences from the ngLOC dataset and trained ngLOC on the remaining data. After training, we ran a 10-fold cross validation on the extracted data, comparing the performance results between the standard ngLOC model against ngLOC-X. For this test, we examined the predictions of only single-localized sequences, resulting in 3,214 sequences from mouse and 683 sequences from fruitfly for analysis. The standard ngLOC model achieved overall accuracies of 93.5% and 79.5% for mouse and fruitfly, respectively. For ngLOC-X, the overall accuracy stayed the same for mouse, and increased to 80.5% for fruitfly. The average sensitivity (often reported as normalized overall accuracy) improved as well, increasing from 86.9% to 87.5% in mouse, and from 72.6% to 74.0% in fruitfly. While the gain in the overall accuracy and sensitivity are not significant, we noted a significant increase in the number of sequences predicted with high confidence. For mouse, ngLOC predicted 39.1% of the data with a CS > 90 at 99.8% accuracy, whereas ngLOC-X predicted 52.9% of the data in the same range at the same accuracy. Fruitfly exhibited the same effect, with ngLOC predicting 28.1% of the data with a CS > 70 at 99.0% accuracy, whereas ngLOC-X predicted 38.7% of the data in the same range at 99.2%

accuracy. We are sure this is an artifact of adjusting the n -gram probabilities to reflect the proteome being predicted. Nevertheless, this test showed us that incorporating the proteome for species X in the model as required for ngLOC- X did not have a negative effect on the performance compared to the standard ngLOC model, while improving the coverage of the proteome predicted with high confidence.

We wanted to determine how the predictions would be affected when ngLOC- X was trained on the proteome of one species, and tested on a different species. When testing the mouse sequences on ngLOC- X trained for fruitfly, the overall accuracy and normalized accuracy again stayed the same. However, when testing fruitfly on ngLOC- X trained for mouse, the overall accuracy dropped from 80.5% to 79.2%, which was slightly worse than the standard ngLOC model. These tests showed us that a species with high representation in the training data will not result in any improvement in overall accuracy by tuning the model for a specific proteome, however a species with low representation will yield the greatest benefit when the model parameters are tuned specifically for that species.

Our next test was to examine the instances in these proteome subsets where ngLOC and ngLOC- X generated different predictions. For the mouse data, we found 62 sequences out of the 3,214 single-localized sequences predicted that resulted in different predictions between the two methods. The standard ngLOC method had 15 of these sequences predicted correctly, whereas ngLOC- X had 16. For the fruitfly predictions, there were 38 sequences out of the 683 sequences with different predictions. Of these, ngLOC had 10 instances that were predicted correctly, whereas ngLOC- X had 17 correct predictions.

While most of these improvements demonstrated by ngLOC-X are statistically insignificant, fruitfly showed a relatively greater improvement from the ngLOC-X method than mouse did. We also discovered in both cases, that almost all sequences with different predictions between the two methods were instances predicted with a low confidence score (e.g., a CS value less than 40.) These results are explained by understanding that low-confidence predictions are more likely for sequences from a species that does not have a high representation of an evolutionarily close species in the training data. The ngLOC dataset has a higher number of proteins from species closely related to mouse (the mammalian proteins) than to fruitfly. This is confirmed by the overall accuracy reported from ngLOC for mouse and fruitfly, which was 93.5% versus 79.5%, respectively, and is also confirmed by the fact that 90.8% of the mouse data was predicted with a $CS \geq 40$, whereas fruitfly only had 66.6% of the data predicted in the same CS range. Moreover, we believe that ngLOC-X will have the most benefit on the predictions from a species with low representation in the training data. This is confirmed by these observations: first, there was a noticeable increase in the overall and normalized accuracy between ngLOC and ngLOC-X on fruitfly, whereas mouse did not benefit from ngLOC-X; second, our cross-species test showed that testing mouse predictions on ngLOC-X trained for fruitfly did not affect the accuracy, whereas fruitfly showed slightly worse performance than the standard ngLOC method when tested on ngLOC-X trained for a mouse. Based on these results, it is evident that ngLOC-X will show improvement in the accuracy of low-confidence predictions over ngLOC. If the sequences from a species being predicted have a high representation of evolutionarily closer species in the training data (such as mouse), then ngLOC-X has little value in the final predictive accuracy. In either case, ngLOC-X never resulted in a decrease in performance

compared to ngLOC, and resulted a significant increase in high confidence predictions, and hence is the method of choice for proteome-wide prediction of subcellular localizations.

Our final test was to compare the location-wise predictions between ngLOC and ngLOC-X on the entire proteome for mouse and fruitfly. For this test, we trained both methods using the entire ngLOC dataset, and then applied each method on the entire GO-annotated proteome data obtained. Table 15 shows the percentage of sequences localized into each possible class.

Table 15 – Comparison of location-wise prediction percentages for mouse and fruitfly

Location	Mouse (<i>M. musculus</i>)		Fruitfly (<i>D. melanogaster</i>)	
	ngLOC	ngLOC-X	ngLOC	ngLOC-X
% CYT	15.86	16.32	13.35	14.60
% CSK	0.88	2.10	0.37	1.29
% END	2.36	3.37	1.76	3.04
% EXC	11.6	12.26	12.50	13.10
% GOL	1.27	2.09	0.97	1.60
% LYS	0.46	0.98	0.24	0.67
% MIT	3.07	4.77	3.46	5.37
% NUC	33.22	30.13	43.90	39.17
% PLA	30.93	27.42	23.23	20.71
% POX	0.33	0.58	0.21	0.44

The prediction for each sequence is determined by observing the most probable class predicted, and assigning that class as the prediction. In this test, all predictions are considered, meaning, no confidence score (CS) threshold is assumed, nor are multi-localized sequences determined. Mouse had 56.8% of the 23,744 predictions for ngLOC generated with a CS $\geq$ 40, compared to 58.1% for ngLOC-X. Fruitfly had 26.3% of the 9,997 predictions for ngLOC generated in the same range, compared to 35% for ngLOC-X. Again, we observed a more substantial increase in coverage for ngLOC-X in the

predictions for the fruitfly proteome, a species with low representation, whereas mouse showed little increase in coverage for the same range. There were 2555 out of 23744 (10.76%) different predictions between ngLOC and ngLOC-X for mouse, and 1126 out of 9997 (12.02%) different predictions for fruitfly. This test showed us that when considering predictions on a proteome level, even a highly represented species such as mouse will result in many predictions of low confidence, and thus can potentially benefit from ngLOC-X as well. We can only offer educated speculation regarding the results, as accurate annotation is not available. However, the proteome-wide predictions obtained by ngLOC-X are closer to what we expect than those obtained by ngLOC. For example, in our previous work which used a completely different method (Guda, et al., 2004), we estimated that 6.3% of the proteome of the fruitfly and 4.6% of the proteome of the mouse is localized in the mitochondria. Our 5.4% and 4.8% predicted with ngLOC-X for fruitfly and mouse, respectively, compared favorably with our former results, and showed significant improvement for mitochondrial estimates over ngLOC in both cases. All of our comparative tests of ngLOC versus ngLOC-X showed that ngLOC-X was a valuable addition to the core ngLOC method.

4.5.8 Estimation of subcellular proteomes of eight eukaryotic species

We have used ngLOC-X to estimate the subcellular proteomes of eight different eukaryotic species. With the exception of yeast, proteomes of eukaryotic model organisms have a significant portion of hypothetical proteins (about 25-40%). To avoid predictions on hypothetical proteins, we generate predictions on a subset of the proteome containing at least one annotated GO concept, i.e., those proteins that have been experimentally validated, or closely related to proteins with experimental validation at

some level. We then use these predictions to generate estimates of the subcellular proteome for each species.

To generate the complete results, we trained ngLOC-X using the entire ngLOC dataset. Predictions were generated for the GO-annotated subset of sequences for each proteome. We selected a confidence score threshold that allows inclusion of all predictions except those of very low confidence. One reason that we did this was that ngLOC predicts only 10 subcellular locations. However, there are other relatively minor organelles in eukaryotic cells that proteins may localize into. (For example, ngLOC does not predict sequences targeted for the vacuole. Though this organelle is nearly non-existent in higher eukaryotic cells, it is significant in yeast cells.) These sequences will likely result in a very low confidence score, as they have no representation in the training data. The other reason why we selected a confidence score threshold was that sequences that have a low homology measure with respect to any other sequence in the ngLOC training data will be hard to classify, and will also result in a low confidence score. For these two reasons, we chose a confidence score threshold (*CStresh*) of 15 as the cutoff value to aid in eliminating these sequences from the proteome estimation. With this threshold, ngLOC covered an impressive range of 94.52-99.82% of the tested proteomes (Table 16). The proteome estimations are based on the percentage of sequences predicted with a confidence score of greater than or equal to *CStresh*. We chose an MLCS threshold (*MLCStresh*) of 60 to estimate the percentage of the proteome that is multi-localized. According to Table 10, in a 10-fold cross validation test, 42% of the multi-localized sequences in ngLOC were predicted with an MLCS of greater than or equal to 60 at an accuracy of 93.9%, whereas only 2.4% of single-localized sequences were incorrectly predicted as multi-localized at this threshold. This is a conservative threshold

chosen to emphasize higher accuracy on multi-localized sequences without over-prediction. We also report the percentage of the proteome multi-localized into both the cytoplasm and nucleus (CYT-NUC), because more than half of the multi-localized sequences in the ngLOC training dataset are localized between these two organelles. Table 16 shows the complete results. Table 17 shows the corresponding chart containing numeric estimates of the fractions in Table 16.

Table 16 – Estimation of the subcellular proteomes of eight eukaryotic organisms

	Yeast	Worm	Fruitfly	Mosquito	Zebrafish	Chicken	Mouse	Human	
	<i>S.cerevisiae</i>	<i>C.elegans</i>	<i>D.melano.</i>	<i>A.gambiae</i>	<i>D.rerio</i>	<i>G.gallus</i>	<i>M.musculus</i>	<i>H.sapiens</i>	RANGE
Proteome:	5799	22400	13649	15145	13803	5394	33043	38149	
GO annotated:	5486	12357	9997	8847	10106	4363	23744	24638	
% ngLOC Coverage:	97.48	94.92	96.73	97.94	98.64	99.82	94.79	94.52	94.79 – 99.82
Proteome Estimated:	5653	21262	13203	14833	13616	5384	31320	36059	
% CYT	15.22	14.80	12.74	14.43	15.01	13.66	13.44	14.14	12.74 - 15.22
% CSK	1.07	1.19	1.05	1.11	1.31	1.24	1.50	1.48	1.05 - 1.50
% END	2.71	3.47	2.85	3.25	3.34	2.53	2.99	3.04	2.53 - 3.47
% EXC	8.88	12.60	12.26	14.28	9.91	12.65	11.52	11.71	8.88 - 14.28
% GOL	1.48	1.31	1.40	1.07	1.68	1.47	1.52	1.56	1.07 - 1.68
% LYS	0.11	0.58	0.55	0.53	0.65	0.44	0.59	0.67	0.11 - 0.67
% MIT	9.55	5.84	4.86	5.52	4.72	4.16	4.24	4.80	4.16 - 9.55
% NUC	33.53	29.75	37.38	29.50	30.31	28.24	27.35	28.38	27.35 - 37.38
% PLA	16.19	24.41	20.06	21.36	21.66	22.78	27.18	24.08	16.19 - 27.18
% POX	0.54	0.66	0.42	0.48	0.51	0.25	0.44	0.46	0.25 - 0.66
% Single-Localized:	89.29	94.60	93.59	91.53	89.11	87.42	90.77	90.32	
% Multi-Localized:	10.71	5.40	6.41	8.47	10.89	12.58	9.23	9.68	
% CYT-NUC:	6.49	2.36	2.76	3.44	5.40	6.27	4.51	4.74	

This chart presents the location-wise percentages of the proteome predicted to localize into one organelle. (For example, 9.55% of the yeast proteome is localized to the mitochondria only.) These percentages sum to the total size of the proteome estimated to be single-localized. We also present the estimated percentage of the proteome that is localized to multiple organelles. The percentage of the proteome estimated to localize to both the cytoplasm and nucleus is also displayed. The coverage is determined with a CS threshold of 15. Multi-localized sequences are determined with an MLCS threshold of 60.

Table 17 - ngLOC subcellular estimates for eight proteomes (actual numbers)

	Yeast	Worm	Fruitfly	Mosquito	Zebrafish	Chicken	Mouse	Human
	<i>S.cerevisiae</i>	<i>C.elegans</i>	<i>D.melano.</i>	<i>A.gambiae</i>	<i>D.rerio</i>	<i>G.gallus</i>	<i>M.musculus</i>	<i>H.sapiens</i>
Proteome:	5799	22400	13649	15145	13803	5394	33043	38149
GO annotated:	5486	12357	9997	8847	10106	4363	23744	24638
% ngLOC Coverage:	97.48	94.92	96.73	97.94	98.64	99.82	94.79	94.52
Proteome Est:	5653	21262	13203	14833	13616	5384	31320	36059

CYT	860	3147	1682	2140	2043	736	4210	5097
END	153	738	377	483	455	136	937	1095
GOL	84	279	184	159	228	79	477	564
CSK	60	252	139	164	179	67	469	533
LYS	6	123	72	79	89	23	184	243
MIT	540	1242	642	818	643	224	1329	1730
NUC	1895	6325	4936	4376	4128	1521	8566	10235
PLA	915	5190	2649	3169	2949	1226	8511	8682
EXC	502	2679	1619	2118	1349	681	3609	4224
POX	31	140	56	72	70	14	138	167
Single-Localized:	5047	20114	12356	13577	12133	4707	28428	32569
Multi-Localized:	606	1147	847	1257	1483	677	2892	3490
CYT-NUC:	367	502	365	510	735	338	1411	1708

Overall, the fractions of subcellular proteomes scaled consistently across the different species, as shown in the last column of Table 16. We observed that the percentage of sequences localized into the endoplasmic reticulum (END), golgi apparatus (GOL), and perixosome (POX) tend to remain relatively consistent across species, with average percentages of 3.0%, 1.44%, and 0.5%, respectively. In contrast, the fractions of the subcellular proteomes with relatively large percentages (CYT, MIT, NUC, PLA and EXC) varied widely across different species. This variation is expected, because as multicellular eukaryotes evolved with higher complexity, consolidation of specific cellular functions to defined organelles took place, resulting in the sequestering of corresponding proteins to these organelles. As a result, more variation is observed in the proteome sizes of larger organelles. Nevertheless, the fraction of subcellular proteomes reported for mouse and human are very similar, which is expected due to their close evolutionary distance. The size of the yeast mitochondrial proteome estimate in this study (9.55%) agrees with those previously reported (~10%) by computational methods (Guda, et al., 2004; Marcotte, et al., 2000), and closely matches the experimental estimates reported (13%) (Kumar, et al., 2002). Similarly, about 1500 nucleus-encoded mitochondrial proteins have been estimated in the human mitochondria (Lopez, et al.,

2000; Taylor, et al., 2003) and our estimate of 4.8% corresponds to 1730 proteins (refer to Table 16 and Table 17), suggesting that ngLOC-X estimates are on par with those obtained by other computational and experimental approaches.

Some of the organelles indicate a trend related to the evolutionary complexity of the species being predicted. The fraction of proteomes localized into the cytoskeleton (CSK) and golgi (GOL) seem to show an increasing trend with the evolutionary complexity of the species, whereas mitochondria (MIT) and nucleus (NUC) indicate a slight decreasing trend. For the other organelles, such trends are not noticeable. Nevertheless, we would like to point out that the proteomes compared in this study are not evolutionarily equidistant, which makes it difficult to infer trends in the evolution of organellar proteomes.

Table 18 shows the prediction percentages for all single- and multi-localized sequences in the human proteome. The boxed areas in the table represent the percentages of single-localized data, as presented in Table 16. The remaining areas in the table represent multi-localized percentages. The sum of the non-boxed cells in Table 18 will result in the percent multi-localized value in Table 16. Though sequences localized to both the cytoplasm and nucleus occupy a significant portion of the multi-localized subcellular proteome, we found that approximately one-third of the sequences localized into the cytoplasm were predicted to localize into other organelles as well. This is probably because the cytoplasm is the default location for protein synthesis as well as the hub of cellular core metabolism. Similarly, almost 1% of the proteome consisted of secreted proteins that were also localized into the plasma membrane.

Table 18 – A matrix showing estimated fractions of subcellular proteomes on the human proteome

Location	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	14.14									
CSK	0.64	1.48	0.01							
END	0.10		3.04							
EXC	0.22	0.01	0.04	11.71						
GOL	0.29	0.03	0.31	0.17	1.56					
LYS	0.02			0.03	< 0.01	0.67				
MIT	0.31		0.07	0.02	< 0.01		4.80			
NUC	4.74	0.07	0.09	0.12	0.01		0.09	28.38		
PLA	0.77	0.02	0.14	0.94	0.09	0.00	0.03	0.19	24.08	
POX	0.05			< 0.01			0.03			0.46

This chart shows the percentages of the proteome estimated to localize over 10 different organelles. The boxed cells represent the percentage of sequences predicted to single-localize, all other cells represent multi-localized sequences. The values are based on a *CStresh* of 15 and *MLCStresh* of 60.

We compared our estimates with those generated by the PSLT method (Scott, et al., 2004). Our estimates of the human subcellular proteome largely agree with those reported by PSLT, with the exception of a difference in the number of multi-localized sequences (16.0% compared to 9.7% reported by ngLOC-X), which is likely due to our conservative choice for *MLCStresh* ≥ 60 . (In comparison, an *MLCStresh* ≥ 50 resulted in 13.4% of the predictions being multi-localized.) We also show a significant difference for those proteins targeted for the plasma membrane (17.1% compared to 24.1% reported by ngLOC-X). This may be significant, as our predictions are based on 24,638 sequences from the human proteome, compared to PSLT's predictions on 9,793 sequences. Moreover, proteins localized to the plasma membrane have a large coverage in the ngLOC dataset. These reasons suggest that our estimates are certainly plausible. Please refer to section A.4 in the Appendix for the complete prediction matrices generated for all eight eukaryotic species.

4.5.9 Biological significance of discriminatory *n*-grams

It is well known that functional domain regions of proteins are highly-conserved because they define a vital part of the overall functionality of the protein. From our previous

studies, we observed that about 74% of the functional domains are localized exclusively to only one of the 10 subcellular locations (Guda and Subramaniam, 2005). Hence, we wondered if we could observe any relationship between discriminatory n -grams and their occurrence in the domain regions in a protein. To perform this test, we downloaded domain definitions from the Interpro database (Mulder, et al., 2007). Only domains definitions that were at least as long as the n -gram length used in the ngLOC model were considered. We mapped these domain definitions onto the single-localized proteins in the ngLOC dataset. This resulted in 15,109 protein sequences that had some portion of its sequence mapped to a functional domain. Overall, 75.5% of the n -grams in these sequences were mapped to a domain. (We say that an n -gram is mapped to a domain only if the entire n -gram falls within the bounds of a domain.) Different localization classes had different coverage of n -grams in domain regions, ranging from 53.7% (NUC) to 86.8% (LYS). See Table 19 for the complete results of this analysis.

Table 19 - Analysis of discriminatory n -grams in domain regions

Location	Expected	Observed
CYT	77.5	96.9
CSK	65.9	85.3
END	83.4	96.0
EXC	80.8	89.5
GOL	73.1	97.5
LYS	86.9	96.7
MIT	86.1	97.8
NUC	53.7	80.2
PLA	80.0	92.1
POX	85.0	97.4
TOTAL	75.5	91.4

This table shows the comparison between the expected percentage of n -grams that will be found in a domain region over all sequences in each class, against the observed percentage of discriminatory n -grams that were found in those same domain regions. Though 75.5% of all n -grams were found in a domain region, we observed 91.4% of all discriminatory n -grams to be located in these same domain regions. These results were calculated over 15,109 protein sequences in the ngLOC dataset that had at least one Interpro domain definition mapping available.

In this study, an n -gram is said to be highly-discriminatory if its occurrence in a protein sequence is highly correlated with a specific localization. We consider a very conservative, strict definition of a discriminatory n -gram as an n -gram that occurs at least 5 times over all sequences, but in only one localization class in the ngLOC dataset. Based on this definition, we found that only 15.0% of all n -grams were highly-discriminatory. However, 91.4% of all discriminatory n -grams occurred entirely within a domain region, suggesting that the discriminatory n -grams indeed originate from the domain regions. It should be noted that the number of discriminatory n -grams found in domain regions vary among different subcellular classes (ranging from 80.2% – 97.5%). Nevertheless, all of these occurrences are statistically significant compared to their expected values, as shown in Table 19.

4.6 Discussion

4.6.1 ngLOC method development

A multinomial Naïve Bayes model is a simplistic, yet effective model when used in conjunction with the n -gram model for representing proteins. The n -gram model is able to capture sequence homology, while allowing for differences due to insertion, deletion, and/or mutation. This model effectively shrinks the protein sequence space, thereby allowing a higher degree of redundancy between proteins of different classes that could not be had by considering the entire protein sequence. It should be noted that the optimal value of n chosen is highly dependent on two factors: (i) the number of sequences in the training data, and (ii) the measure of sequence similarity in the training data. Generally, both large datasets and datasets with high sequence similarity will need longer n -grams for effective classification, though larger values of n will likely overfit the training data.

Additionally, this has an affect on the confidence score. If the dataset is large and highly similar, we found that short n -grams lead to probabilities that are all relatively close in value, which results in confidence scores all falling within a very tight range. The reasoning for this is due to the total number of n -grams in Equation (4.7) being proportionally large with respect to the size of the dataset. For example, when using a 2-gram model on our dataset, the scores for the entire dataset all ranged between 8.41-11.72, while when using a 7-gram model, the range is 0.0-99.21. Though the scores were in a tight range for the 2-gram, we observed the same relationship between relative score value and overall accuracy. It would be easy to rescale the scores for performance analysis to fall within similar ranges across all models.

From the protein feature space point of view, a different sized n -gram will map protein surface features differently. We believe that the high performance shown by 6-8 gram models (Figure 7) is due to the fact that these n -gram peptides are ideal for mapping the secondary structure space of proteins. Secondary structure elements are vital for attaining a proper fold of a protein, and consequently are vital for its function. Hence, these secondary structures are distinctly conserved across proteins with different functions and from different subcellular locations.

4.6.2 Comparison with other methods

Many recent methods, including PLOC, have been based on SVMs (Hoglund, et al., 2006; Park and Kanehisa, 2003; Yu, et al., 2006). As successful as some of these models have been, we determined that SVMs were not suitable for our needs. First, we plan to explore the most discriminatory n -grams in proteins between different subcellular organelles. With ngLOC, it will be easy to extract n -grams of interest from the model, as

the relation between each n -gram and the integer identifier generated for use by the classifier is symmetric. However, with SVM-based methods, the kernel in the SVM projects the features of the data to a higher dimensional space to increase the likelihood of making the data linearly separable. While one might discover excellent SVM parameters for a particular classification problem, it will be difficult to understand how the translated feature space is discriminating between classes. Second, as we have illustrated, a probabilistic measure ought to be considered a crucial part of any predictive model. Therefore, we determined that a pure probabilistic model was desired. Deriving this measure is a difficult feat for SVMs due to their non-probabilistic output. Any attempt to derive such a measure with SVMs can be done only by creating another layer of classification to simulate a probability measure from the output of the SVMs. The risk of the simulated probability distribution overfitting the data used to generate the distribution is a known artifact with these methods (Drish, 2001). The pure probabilistic confidence measure derived directly from the probabilities calculated from a method such as ngLOC will have a more consistent, scalable probability measure.

4.6.3 Estimation of subcellular proteoms

Our model, ngLOC, was enhanced to allow the dynamic adjustment of the model parameters specific to a proteome being estimated. This model, termed ngLOC-X, is useful for predicting the subcellular localization of the proteome of any species. Our proteome prediction results showed that although a single model can be used on a variety of species, better results can be had if the model is tuned for a specific species being considered (Table 15). If a single model is being used across numerous species, it is very important to include a broad spectrum of data across all species. Unfortunately, this is

not possible due to the imbalanced nature of protein sequence data in the public domain. Our model, ngLOC-X, essentially extends the core ngLOC model by introducing a bias toward a single species being predicted. The accuracy and coverage of our model across species will continue to burgeon as the proteomes of new eukaryotic species become available. The eukaryotic species selected in this study represent a broad spectrum in the eukaryotic superkingdom (not including plants). Despite this, corresponding fractions of each subcellular proteome fall within a reasonable range across species (Table 16). This suggests that the proteome size corresponding to the core functionality of an organelle remains unchanged across species, while the observed variation in size allows for functionalities required by specific species for their adaptation. This hypothesis can be tested by studying the organellar proteomes at the domain level, and we aim to continue this work in the future.

4.6.4 Discriminatory *n*-grams and functional domains

It is known that targeting signals such as KDEL/HDEL (for endoplasmic reticulum) and SKL (for peroxisomes) play a distinct role in transporting a protein to its destination in the cell. Nevertheless, ngLOC does not require prior knowledge regarding such signal peptides, nor does it explicitly consider such information in the prediction process.

Despite this, ngLOC is able to perform better than methods that explicitly use such information because each discriminatory *n*-gram is analogous to such signals. To support this argument, we demonstrated that 91.4% of all discriminatory *n*-grams originate from the domain regions of proteins (see Table 19), which define the core function of a protein. The observations suggest that ngLOC predictions are based on functionally-significant regions (domains) of the protein sequences, which are represented by *n*-grams

covering the entire sequence space. In contrast, methods relying on target signals generally scan only the N-terminal or C-terminal regions of protein sequences where such signals are located. It is likely that if the targeting signals are shorter than the n -gram, the discriminatory n -grams represent both the signal as well as its neighborhood (which is often very important for transport). Similarly, if protein transport requires motifs that are longer than the n -gram, such motifs would be represented by multiple and mostly contiguous n -grams. Therefore, ngLOC need not have prior knowledge of specific targeting signals, as it is likely that analogous signals (discriminatory n -grams) are inherently identified *de novo* and used in establishing the localization prediction of the protein. Because of this plasticity, the ngLOC method has the ability to perform well on a number of locations, and hence is highly suitable for proteome-wide prediction of subcellular localization.

4.7 Conclusions

In this post-genomic era, there is a crucial need for computational methods that can classify newly discovered proteins using information derived only from the primary sequence. Methods that predict the subcellular localization of a protein are an important part of meeting this need. In our study, we have developed the ngLOC method, a Bayesian classifier that can predict the subcellular localization of a protein with superior performance against other methods of similar scope.

Because ngLOC is a probabilistic method, we were able to generate an extremely useful probabilistic confidence score that places a measure of credibility on each prediction. We have shown how this measure was used to determine the most likely localization for new proteins and possible annotation errors on known proteins. From

this score, we also were able to develop a confidence measure to aid in determining multi-localized proteins as well, which is an important need in this area, because a significant part of the proteome is known to localize into multiple compartments. These scores developed in this study are sound and useful for predicting sequences localized to both single or multiple locations with high accuracy.

We extended the core ngLOC method, called ngLOC-X, and showed how it improved coverage for proteome-wide predictions over a single species by performing a comparative analysis of the results from both methods. We applied ngLOC-X to estimate ten distinct subcellular proteomes for eight eukaryotic model organisms. To our knowledge, this study presents the first estimate of ten distinct organelles on eight eukaryotic species with our coverage.

As with most computational models, the accuracy of ngLOC is completely dependent on the quality and coverage of the dataset used to train the model. While many methods are unable to use large datasets due to computational limitations, ngLOC does not have these limitations. Clearly, modern day genomics will continue to produce increasing amounts of experimentally determined data. The simplicity of the ngLOC model will enable it to easily incorporate these new data as they become available, thereby increasing the accuracy and coverage of ngLOC in the future. ngLOC can play a significant role in this field, when used in conjunction with experimental methods, to help meet the needs of the research community.

5 ngLOC-bact – Supervised learning for prokaryotic proteome estimation

Background

Computational methods for predicting the subcellular localization of bacterial proteins play a crucial role in the ongoing efforts to annotate the function of these proteins and to suggest potential drug targets. These methods, used in combination with other experimental and computational methods, can play an important role in biomedical research by annotating the proteomes of a wide variety of bacterial species.

Results

We use the ngLOC method, an n-gram based Bayesian classifier that predicts the subcellular localization of a protein based on a new curated dataset of experimentally determined proteins. Subcellular localization was predicted with an overall accuracy of 89.7% and 89.3% for Gram-negative and Gram-positive bacteria protein sequences, respectively. Through the use of a confidence score threshold, we improve the precision to 96.6% while covering 84.4% of Gram-negative bacterial data, and 96.0% while covering 87.9% of Gram-positive data. We use this method to estimate the subcellular proteomes of ten Gram-negative species and five Gram-positive species, covering an average of 74.7% and 80.6% of the proteome for Gram-negative and Gram-positive sequences respectively.

Conclusions

The current method is useful for large-scale analysis and annotation of the subcellular proteomes of bacterial species. We demonstrate that our method has the best proteome coverage, and that the accuracy of our method is superior to comparable methods.

NOTE: This work represents a completed manuscript that has yet to be submitted.

5.1 Background

The previous chapter gave an exhaustive analysis and evaluation to the ngLOC method(King and Guda, 2007), including how it was applied to the estimation of the proteome for a wide range of eukaryotic species. This chapter will focus on the application of the ngLOC method toward the estimation of bacterial proteomes. As with eukaryotic species, accurate prediction of bacterial subcellular proteomes can improve our understanding of defined cellular processes at various subcellular levels. However, having the ability to predict subcellular localization of protein in bacteria has special interest in the research community, partly because such methods can aid in suggesting plausible candidates for drug targeting in bacterial pathogen research, which is important because immunogenic proteins targeted to the cell surface make ideal drug targets.

Gram-negative bacterial cells are primarily composed of the cytoplasm, an inner membrane surrounding the cytoplasm, and a cell envelope consisting of an outer membrane and an area between the membranes known as the periplasm. Gram-positive bacteria have a slightly different structure, in that they have a peptidoglycan cell wall surrounding the inner/cytoplasmic membrane instead of a defined outer membrane and periplasm. Bacterial proteins are synthesized in the cytoplasm of the cell. These proteins remain in the cytoplasm, or are targeted to one or more possible locations in the cell

through various transport systems and signal peptides in the sequence. Proteins that are transported to the extracellular space are of particular interest since they are ideal candidates for drug targeting. Although many of these transport systems are well understood, the total range of possible targeting mechanisms has yet to be characterized (Gardy and Brinkman, 2006; Holland, 2004). Regardless, the signaling and transport mechanisms for the protein, along with most of the structural and functional attributes for the protein, lie in the primary structure of the protein – the sequence of amino acids. Computational methods for prediction of subcellular localization from sequence capitalize on this knowledge.

Numerous methods have been developed for prediction of subcellular localization of bacterial proteins (Bhasin, et al., 2005; Chou and Shen, 2006; Gardy and Brinkman, 2006; Gardy, et al., 2005; Gardy, et al., 2003; Lu, et al., 2004; Nakai and Horton, 1999; Shen and Chou, 2007; Yu, et al., 2004). Some methods explicitly use prior knowledge of known transport and signaling mechanisms; some rely solely on machine-learned classifiers to implicitly represent this knowledge, while others use a combination of both to generate the prediction for a protein. The PSORT-I method (Nakai and Horton, 1999), one of the first methods available for subcellular localization prediction, was a rule-based system that used amino-acid composition, known motifs, targeting signals and other structural information to make predictions for four Gram-negative and three Gram-positive subcellular locations. This work has been improved with the release of the PSORTb method (Gardy, et al., 2005; Gardy, et al., 2003), which expanded the number of subcellular locations predicted as well as the coverage of predicted proteins. The PSORTb approach is similar to the original PSORT I method, in that both apply prior knowledge of various features known to correlate to distinct localizations. The predictive

performance of the PSORTb method was improved by the use of modern classification algorithms, such as support vector machines (SVM) and through integration of each individual component output via a Bayesian network to produce a final score. SVM-based methods are common in this field of research, with methods that focus on bacterial protein localization differing only in the feature space considered as the input to the classifier (Bhasin, et al., 2005; Chou and Shen, 2006; Gardy, et al., 2005; Shen and Chou, 2007; Yu, et al., 2004). A few methods that extract keywords from the annotations associated with each sequence have appeared in recent years. The Proteome Analyst is one such method that can be used on bacterial proteins. The method works by BLASTing the query sequence against the Swiss-Prot database, selecting a set of the best homologous sequences, and extracting text associated with the homologous sequences (Altschul, et al., 1990; Boeckmann, et al., 2003; Lu, et al., 2004). For further information on bacterial protein subcellular localization prediction, we recommend that the reader refer to an excellent review by Gardy and Brinkman (Gardy and Brinkman, 2006). For a general review of many of the current methods and challenges in protein localization prediction, we recommend the reviews by Dönnies and Höglund (Donnes and Hoglund, 2004) and Sprenger et al (Sprenger, et al., 2006).

In this study, we address the task of predicting the subcellular localization of bacterial proteins using information that can be derived from the sequence of the protein alone. Our work is based on previous work completed on the ngLOC method (King and Guda, 2007), which was developed to annotate the subcellular localization of eukaryotic proteins. The ngLOC method is an n -gram based Bayesian method based on sequence homology, and operates by observation of fixed length subsequences of length n (i.e. n -grams) over different subcellular localization classes. The method was designed to

address many of the known limitations of existing methods that make them prohibitive for proteome-wide predictions.

5.2 *The ngLOC-bact method*

This section introduces aspects of the ngLOC method that are specific for this particular study. For a thorough exposition on the ngLOC method, see Chapter 4, Section 4.1.

5.2.1 ngLOC-bact dataset

The dataset used for this task is a set of protein sequences taken from the Swiss-Prot database, release 54 (Boeckmann, et al., 2003), which contains experimentally determined annotations for subcellular localization. We applied the following filters to obtain high-quality data for training and testing purposes: only bacterial (not archaeal) sequences were considered; sequences with predicted and ambiguous localizations were removed, including those annotated with the terms 'probable', 'potential', 'likely', or 'by similarity'; sequences shorter than 10 residues in length were removed. There are very few bacterial proteins in the Swiss-Prot database that have been experimentally determined to be multi-localized in the cell, and therefore we removed such sequences (see the Discussion section for related information on this topic). We separated proteins from Gram-negative (i.e. have an outer membrane encasing an inner membrane, with a very thin cell wall) and Gram-positive (i.e. an inner membrane encased in a thick cell wall) species, resulting in 7,229 and 2,814 sequences, respectively. Finally, we used the cd-hit sequence clustering software (Li and Godzik, 2006) to reduce the maximum sequence identity to 98%, resulting in a final dataset of 6,558 Gram-negative sequences

and 2,447 Gram-positive sequences. Table 20 shows the location-wise distribution of the resulting datasets.

Table 20 – Gram-negative and Gram-positive training data

Localization	Code	Gram-Negative Sequences	Gram-Negative % of Data*	Gram-Positive Sequences	Gram-Positive % of Data
Cytoplasm	CYT	4,139	63.0%	1,776	72.6%
Extracellular	EXC	263	4.0%	292	11.9%
Inner Membrane	IN	1,397	21.3%	347	14.2%
Outer Membrane	OUT	344	5.2%	NA	NA
Periplasm	PER	415	6.3%	NA	NA
Cell Wall	WAL	NA	NA	32	1.3%
Total		6,558		2,447	

This table shows the distribution of proteins in the training data for both Gram-negative and Gram-positive bacteria over each subcellular location. NA – Not applicable.

5.2.2 Proteome datasets

The proteomes and corresponding PSORTb v2.0 predictions were downloaded from the PSORTdb online database of subcellular localization predictions for bacteria (Gardy, et al., 2005; Rey, et al., 2005). We randomly chose ten Gram-negative organisms and five Gram-positive organisms for subcellular proteome estimation purposes. The Gram-negative examples include *Anaeromyxobacter dehalogenans*, *Campylobacter jejuni*, *Caulobacter crescentus*, *Escherichia coli*, *Geobacter sulfurreducens*, *Haemophilus influenzae*, *Helicobacter pylori*, *Neisseria gonorrhoeae*, *Rickettsia conorii*, and *Thiobacillus denitrificans*. The Gram-positive examples include *Bacillus anthracis*, *Clostridium acetobutylicum*, *Lactobacillus acidophilus*, *Mycobacterium tuberculosis*, and *Streptococcus pneumoniae*.

5.2.3 ngLOC-bact – The ngLOC method for bacterial localization prediction

The ngLOC method is a Bayesian classification method for predicting protein subcellular localization as described in King and Guda (King and Guda, 2007). Our method uses n -gram peptides derived solely from the primary structure of a protein to explore the search space of proteins. It is suitable for proteome-wide predictions, and is also capable of inferring multi-localized proteins, namely those localized to more than one subcellular location. We modified the original method to predict distinct subcellular localization classes over Gram-negative and Gram-positive bacterial proteins, and to emphasize a balance between high precision across all subcellular localization classes, versus high coverage across prokaryotic proteomes. We refer the reader to the research of King and Guda (King and Guda, 2007) for a detailed analysis of the method.

5.3 **Results**

The evaluation criteria used in this analysis is similar to the analysis presented on the ngLOC method; however, a more thorough analysis was performed in light of the highly unbalanced nature of the data over fewer classes. Standard performance measures over each subcellular location are reported, including sensitivity (recall), precision, specificity, and Matthews correlation coefficient (MCC). The latter provides a measure of performance for a single predicted class; the MCC value is 1 for perfect predictions on that class, 0 for random assignments, and less than 0 if predictions are worse than random (Matthews, 1975).

We apply a standard validation technique known as ‘leave-one-out’ validation (also known as ‘jack-knife’), in which one sequence is removed from the training data, the predictive model is trained using the remainder of the data, and the model is then

tested on the single sequence that was removed. This process is repeated for the entire dataset, one sequence at a time.

5.3.1 Performance metrics in the midst of unbalanced data

For evaluation of the overall classifier performance, we report overall accuracy as the fraction of the data tests that were correctly classified. However, due to the highly unbalanced nature of the data over each localization class, overall accuracy is not a useful measure upon which to rely exclusively. For such unbalanced datasets, a good solution that has been used in information retrieval is to report *macro-averaged* class measures, which average the individual class measures over all classes (Yang and Liu, 1999). We also report a macro-averaged F_1 measure (denoted $Mac-F_1$), where the F_1 measure for a given class combines sensitivity and precision with equal weighting to produce a balanced measure (van Rijsbergen, 1979). The equation is given as follows:

$$F_{1_j} = \frac{2 \times Sens_j \times Prec_j}{Sens_j + Prec_j}$$

$$Mac-F_1 = \frac{\sum_{j=1}^{|C|} F_{1_j}}{|C|}$$

5.3.2 The Receiver Operating Characteristic (ROC) Curve

We report a receiver operating characteristic (ROC) curves for each class as a graphical means of observing the discriminatory ability of the classifier. ROC curves have been increasingly adopted in the machine learning and data mining community as a more

rigorous means of comparing classifiers, in part because researchers are realizing that simple accuracy measures are often a poor metric for measuring classifier performance (Ling, et al., 2003; Provost, et al., 1998). Moreover, it is known that resulting ROC curves plotting the classifier performance are independent of class distributions, which is an ideal consideration for our classification problem due to our highly unbalanced dataset. ROC curves typically plot the true positive rate (TPR, which is equivalent to sensitivity) against the false positive rate (FPR, which is equivalent to $1.0 - \text{specificity}$). Individual points on the graph are plotted on the basis of selected scoring thresholds output by the classifier. The curve begins at the bottom-left corner of the plot area and rise to the top-right corner. A classifier that performs no better than random will result in a plot that is close to the diagonal. The closer a curve comes to the top-left corner, the better the classifier will be able to discriminate between positive and negative instances. We generate a single ROC curve for a classifier by computing the macro-average of the fraction of true positives and false positives over each class observed at distinct CS thresholds, denoted as a Mac-ROC curve. These curves help us determine the ability for the classifier to discriminate between true predictions and false predictions over the entire range of confidence scores for each class. We compute the area under the Mac-ROC curve as another single measure to evaluate the overall predictive performance of the classifier. The range of the AUC measure will be between 0.0 and 1.0, where 1.0 is a perfect classifier, and a 0.5 is a classifier that is performing no better than random, and less than 0.5 is worse than random guessing. This last measure is particularly useful to determine how well the confidence score generated by the ngLOC method discriminates between correct and incorrect predictions.

5.3.3 Evaluation of different size n -grams

Our first test consisted of determining an appropriate n -gram length for constructing the feature space. The optimal value of n chosen will depend on a variety of factors. The size of the training data and the measure of similarity of the data have a clear influence on n , because as the percentage of sequence identity observed in the training data increases, the value of n required to discriminate between sequences belonging to different classes also increases. Values of n that are too large, however, will lack an ability to generalize beyond the training data because n -grams observed in the test sequence are unlikely to be observed in the training data. Likewise, a value of n that is too short will be unable to effectively learn discriminatory features in the training data. These problems are more pronounced when learning model parameters from highly unbalanced classes. We perform a separate test for each dataset to ensure that we select an optimal value of n for each classifier. Table 21 shows the performance results for both datasets.

Table 21 - Performance of different n -gram models

n -gram	<i>Gram-Negative</i>			<i>Gram-Positive</i>		
	<i>Overall</i>			<i>Overall</i>		
	<i>Accuracy</i>	<i>Mac-F1</i>	<i>AUC</i>	<i>Accuracy</i>	<i>Mac-F1</i>	<i>AUC</i>
1	82.5%	0.658	0.673	87.4%	0.715	0.770
2	84.5%	0.715	0.746	88.7%	0.738	0.772
3	88.5%	0.788	0.829	91.1%	0.769	0.807
4	90.3%	0.819	0.882	92.1%	0.694	0.780
5	89.7%	0.795	0.904	90.8%	0.751	0.756
6	89.8%	0.808	0.898	89.3%	0.732	0.895
7	87.3%	0.799	0.862	87.7%	0.706	0.898
8	84.5%	0.771	0.847	86.7%	0.682	0.901

This table shows overall accuracy, macro-averaged F_1 , and the area under the ROC curve (AUC) performance metrics over varying n -gram lengths on both the Gram-negative and Gram-positive datasets.

Notice that on both datasets, the 4-gram model has the highest overall accuracy. Based on the results for the Gram-negative data, one may consider using either the 4-gram, 5-gram or 6-gram models, because these models have relatively close performance

measures. The ROC curve in Figure 10 shows that the 5-gram and 6-gram models offer the most discrimination between true and false predictions.

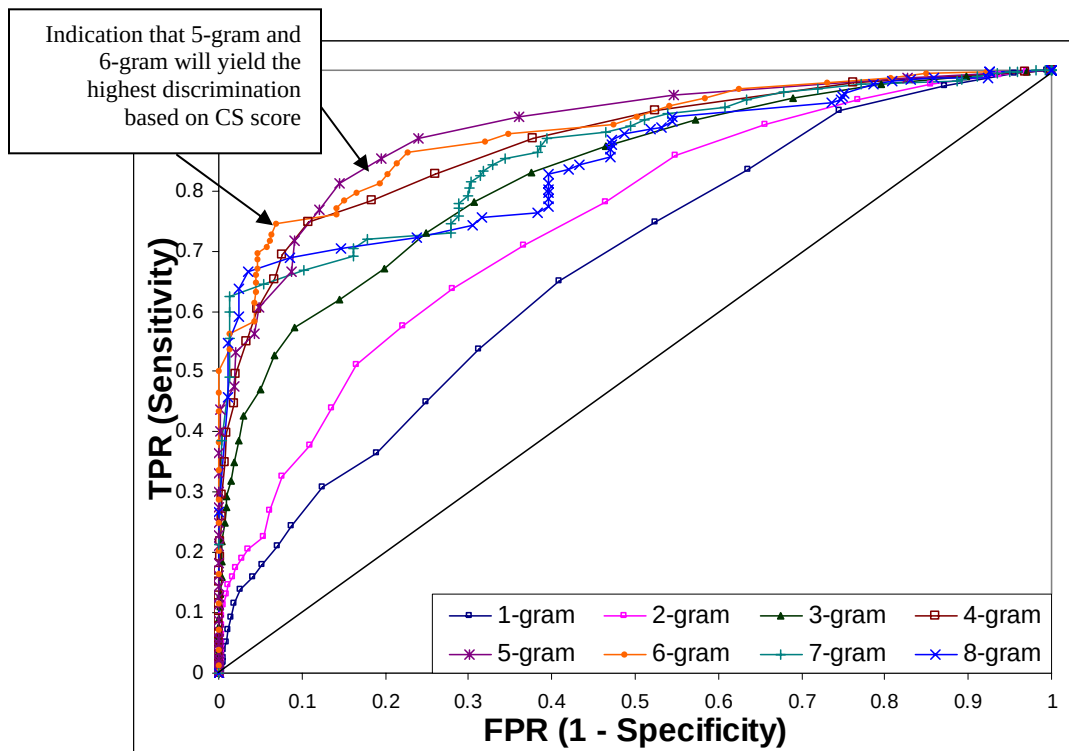


Figure 10 - ROC Curves for [1-8]-gram models on Gram-negative data

This figure depicts the ROC curves plotted, where the sensitivity and specificity are based on macro-averaged calculations observed for distinct CS thresholds across each class. We point out that the 5-gram and 6-gram are likely to have the best discriminatory ability based on CS.

To illustrate, we use the confidence score (CS) output by the ngLOC method for each prediction to enable us to select only those predictions of high confidence. This has the effect of increasing the precision by lowering false positive predictions. If we select a CS threshold (*CSthresh*) that allows only 70% coverage (where ‘coverage’ is defined as the percentage of the data that is meeting or exceeding the specified *CSthresh*), we obtain excellent macro-averaged precisions of 98.6%, 99.0% and 99.5% for a 4-gram, 5-gram and 6-gram model, respectively. However, we lose predictions for 30% of the

data. A more liberal threshold that allows coverage of 95% of the data results in macro-averaged precisions of 89.9%, 94.9% and 95.5%, respectively, coinciding with the resulting ROC curves. (Please see Table 22 for complete data.)

Table 22 - Coverage vs. precision for gram-negative data

<i>n</i> -gram	Coverage	Micro-Averaged Precision	Macro-Averaged Precision
4-gram	50%	99.7%	99.5%
	70%	99.2%	98.6%
	95%	92.2%	89.9%
	100%	90.3%	85.9%
5-gram	50%	100.0%	100.0%
	70%	99.3%	99.0%
	95%	91.8%	94.9%
	100%	89.7%	92.2%
6-gram	50%	100.0%	100.0%
	70%	99.7%	99.5%
	95%	91.8%	95.5%
	100%	89.8%	93.0%

This table shows how limiting the coverage of predictions generated by ngLOC through selecting various CS thresholds dramatically improves the precision. The coverage of predictions generated is increased by decreasing the CS threshold setting, thereby allowing lower confidence predictions to be generated. Both micro-averaged and macro-averaged precisions are shown in order to accurately convey performance across all classes. For 100% coverage, micro-averaged precision is equal to overall accuracy for the classifier.

While the curve in Figure 10 does suggest that certain *CStresh* values could be chosen to allow a slightly higher precision on a 5-gram model, the *mac-F₁* metric suggests that the 6-gram model should have slightly better performance (see Table 21). We are interested in maximizing the number of correct predictions, as opposed to maximizing overall accuracy. Therefore, we chose the 6-gram model for the Gram-negative dataset.

Analyzing the results for the Gram-positive data shows impressive overall performance measures for the 3-gram and 4-gram model, at 91.1% and 92.1%, respectively (see Table 21). However, it is interesting that the 4-gram model also results

in one of the lowest $mac-F_1$ values, at only 0.694. Moreover, if we were to choose either of these models, we would lose significant sensitivity on the smallest classes, compared to higher n -gram models. For example, selecting a threshold that allows a generous 90% coverage results in macro-averaged precision values of 86.8% and 71.9% on a 3-gram and 4-gram model, compared with 92.4% and 95.2% on the 5-gram and 6-gram models, respectively (data not shown). The poor performance of the 4-gram model is a result of an absence of predictions for cell wall at this threshold. Moreover, for those predictions achieving a CS below our threshold, only 9% of all sequences localized to the cell wall were predicted correctly with the 4-gram model. We can conclude that the 3-gram and 4-gram models lack an ideal $CStresh$ value that would achieve high coverage while reducing the false positive rate (FPR) across all classes. The ROC curves for the Gram-positive data (see Figure 11 below) and corresponding AUC (see Table 21) show that the 6-gram, 7-gram and 8-gram models all perform reasonably well, suggesting that a $CStresh$ value can be chosen that can effectively discriminate between true-positives and false-positives for these models. However, the 7-gram and 8-gram models have comparatively poor $mac-F_1$ values compared to the 6-gram model, suggesting that they will not perform well on the smallest classes. These reasons suggest that the 6-gram model should be the model of choice for the Gram-positive data.

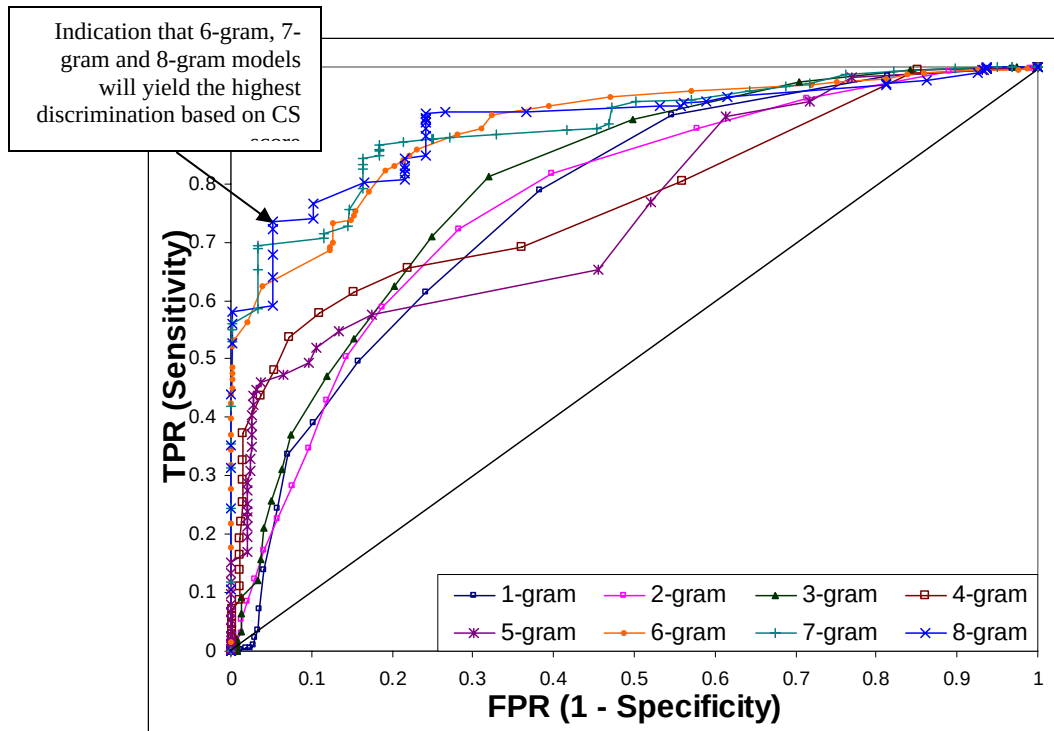


Figure 11 - ROC curves for [1-8]-gram models on gram-positive data

This figure depicts the ROC curves plotted for gram-positive data, where the sensitivity and specificity are based on macro-averaged calculations observed for distinct CS thresholds across each class. We point out that the 6-gram, 7-gram and 8-gram are likely to have the best discriminatory ability based on CS, and according to Table 21, they all have reasonable AUC calculations. However, the overall accuracy and Mac-F1 scores of the 7-gram and 8-gram model are significantly lower than the 6-gram, suggesting that the 6-gram is the best overall candidate model to use.

5.3.4 Class-wise analysis of the 6-gram model

For our next test, we used the 6-gram model to further analyze the performance on individual classes. We ran both datasets separately, with and without specifying a confidence score threshold (*CSthresh*) (see Table 23). We chose a *CSthresh* that would result in a macro-averaged precision of 98% on the Gram-negative data and 95% on the Gram-positive data. While these are rather restrictive selections, we deemed this to be important in order to use only high-confidence predictions. (We chose a slightly lower threshold for the Gram-positive data because of the extremely small proportion of data localized to the cell wall).

Table 23 - Jack-knife performance of ngLOC on training data using a 6-gram model

GN Localization	ngLOC				ngLOC (with CStresh)			
	<i>Precision</i>	<i>Sensitivity</i>	<i>Specificity</i>	<i>MCC</i>	<i>Precision</i>	<i>Sensitivity</i>	<i>Specificity</i>	<i>MCC</i>
Cytoplasmic	88.7%	99.2%	78.5%	0.82	96.1%	95.6%	89.8%	0.85
Extracellular	94.0%	59.7%	99.8%	0.74	98.6%	53.6%	100.0%	0.72
Inner Membrane	90.8%	83.0%	97.7%	0.83	98.1%	60.1%	99.7%	0.72
Outer Membrane	98.7%	68.0%	100.0%	0.81	99.0%	59.0%	100.0%	0.75
Periplasmic	92.5%	56.1%	99.7%	0.71	98.1%	50.4%	99.9%	0.69
Micro-averaged Results	89.8%	89.8%	97.5%		96.7%	81.6%	99.2%	
Macro-averaged Results	93.0%	73.2%	95.1%		98.1%	63.7%	97.9%	
GP Localization								
Cytoplasmic	88.8%	99.3%	66.8%	0.75	95.9%	96.7%	82.8%	0.81
Extracellular	89.9%	63.7%	99.0%	0.73	97.7%	57.9%	99.8%	0.73
Cell Membrane	94.1%	64.8%	99.3%	0.75	97.1%	49.0%	99.7%	0.66
Cell Wall	78.6%	34.4%	99.9%	0.52	90.9%	31.3%	100.0%	0.53
Micro-averaged Results	89.3%	89.3%	96.4%		96.1%	84.4%	98.7%	
Macro-averaged Results	87.8%	65.6%	91.3%		95.4%	58.7%	95.6%	

This table shows the performance of ngLOC on both the Gram-positive (GP) and Gram-negative (GN) bacterial datasets using a 6-gram model, with and without a CS threshold. We report both micro-averaged and macro-averaged results.

Table 23 shows the effect of selecting a proper CS threshold, which produces a remarkable improvement in the precision measures across all classes. The only drawback to the use of a scoring threshold in this manner is the reduction of the number of sequences for which predictions are generated. Predictions that fall below the specified CS threshold are considered to be false-negatives, and likewise affect the overall accuracy (reported as micro-averaged sensitivity in Table 23). This value decreased from 89.8% to 81.6% in the results for Gram-negative data, and from 89.3% to 84.4% for Gram-positive data. Figure 12 shows the ROC curve for each individual class for the Gram-negative data. The bold-line curve is the macro-averaged ROC curve (mac-ROC) over all classes, which is identical to the 6-gram curve displayed in Figure 10.

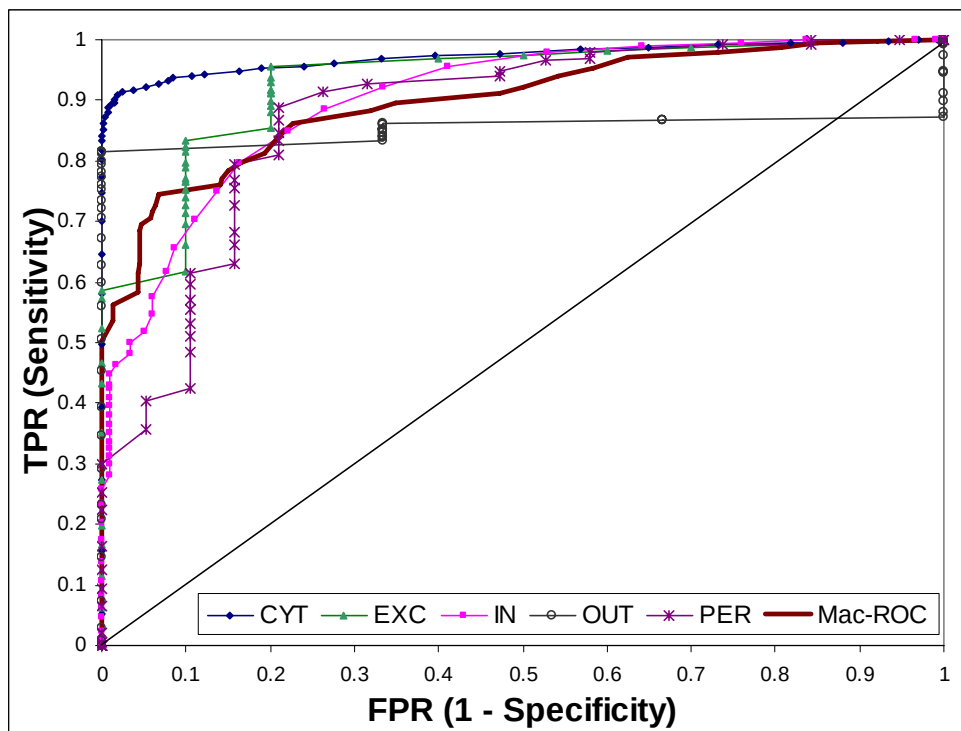


Figure 12 - ROC curves for 6-gram model on Gram-negative data

This figure depicts the ROC curves plotted against a 6-gram model. The Mac-ROC plot is the average ROC curve over each of the individual curves. (CYT = cytoplasmic, EXC = extracellular/secreted, IN = inner/cytoplasmic membrane, OUT = outer membrane, PER = periplasmic, Mac-ROC = macro-averaged ROC curve)

Note that this curve is distinct and distant from the curve for cytoplasm, which is the most prominent class in the data and which shows the highest discriminatory performance. This illustrates the importance of using macro-averaged values to indicate performance in the context of unbalanced data. Without averaging, the mac-ROC curve would be very close to the curve for cytoplasmic sequences, giving a false sense that the classifier is near perfect for all classes. We noticed that the curve for the outer membrane data looks peculiar; however, we also observed remarkably high precision on these sequences without a CS threshold, at 98.7%. This is an anomaly that may happen when there are very few false predictions generated with relatively higher confidence scores.

See the Discussion section for more information pertaining to these and related topics surrounding ROC curves.

5.3.5 Comparison with other methods

For comparison purposes, we used a test dataset of Gram-negative bacteria protein sequences that was used by Chou and Shen (Chou and Shen, 2006) for comparing their method (called Gneg-PLoc) against PSORTb. This dataset, denoted as *GP-PSORTb*, contained 1114 sequences of which 945 sequences were overlapping with our training dataset for Gram-negative bacteria. The results are reported in Table 24.

Table 24 - Comparison of ngLOC against other methods using *GP-PSORTb* test dataset

		<i>PSORTb</i>	<i>Gneg-PLoc</i>	<i>ngLOC</i>	<i>ngLOC</i>	<i>ngLOC(UO)</i>	<i>ngLOC(UQ)</i>
	<i>Sequences</i>	<i>Sensitivity</i>	<i>Sensitivity*</i>	<i>Sensitivity</i>	<i>Specificity</i>	<i>Sensitivity</i>	<i>Specificity</i>
Cytoplasm	140(39)	83.6%	93.6%	100%	99.6%	100%	96.9%
Extracellular	74(18)	31.1%	90.5%	98.7%	100.0%	94.4%	100.0%
Inner Membrane	687(74)	83.0%	97.5%	99.7%	98.8%	97.3%	98.9%
Outer Membrane	97(21)	82.5%	87.6%	99.0%	99.4%	95.2%	100.0%
Periplasmic	116(17)	81.0%	94.8%	95.7%	100.0%	94.1%	100.0%
Overall		79.4%	95.4%	99.2%		97.0%	
Accuracy							

This table reports the individual sensitivity measures (TP / (TP + FN)) for each class resulting from the original PSORTb v1.1 method, Gneg-PLoc method, and ngLOC. Specificity is also indicated for the results from the ngLOC method. The *Sequences* column reports the number of sequences in each class. The number in parentheses is the number of sequences that are not in the ngLOC training data. The *ngLOC* column shows the results from the ngLOC method. The *ngLOC(UO)* column shows the performance of ngLOC on the 169 unique sequences in the test data that did not exist in the training data. We show the overall accuracy resulting from each method at the bottom of the sensitivity columns.

*(The PSORTb and Gneg-PLoc results are taken from the results reported by Chou and Shen (Chou and Shen, 2006)).

The *ngLOC* column indicates the performance of the ngLOC method on the entire *GP-PSORTb* test dataset, and the *ngLOC(UO)* column indicates the performance on the 169 non-overlapping sequences that were unique to the *GP-PSORTb* dataset. We report these results separately to demonstrate that our method also performs well on the portion of the

test data that does not overlap with our training data. Specificity is also reported only for the ngLOC method. Finally, not all of the methods that were compared make use of a scoring threshold to improve precision; therefore we report the results without specifying a confidence score threshold.

While the results indicate that all methods outperform PSORTb v1.1, the ngLOC method outperforms the other results presented. (We were unable to obtain results for PSORTb v2.0.) The results presented in the *ngLOC* column represent an unfair comparison due to the substantial number of sequences that exist in both the training data and test data. The *ngLOC(UO)* test results offer a more legitimate performance validation because the test eliminates any redundant sequences between the training and testing datasets. Though the overall sensitivity for the non-overlapping sequences (*ngLOC(UO)*) drop slightly from 99.2% to 97.0%, these results still indicate that the method is performing remarkably well.

We also use a separate dataset of test sequences, denoted *GP-test*, that was assembled by Chou and Shen (Chou and Shen, 2006) for the purpose of independently testing their own method. This dataset consisted of 637 sequences, of which 512 overlapped with sequences in the ngLOC training data for Gram-negative bacteria. We performed an analysis identical to the previous test and present our results in Table 25.

Table 25 - Performance of ngLOC on *GP-test* dataset

	<i>Sequences</i>	<i>Gneg-PLoc*</i>	<i>ngLOC</i>	<i>ngLOC(UO)</i>
Cytoplasm	210(62)	177/210 = 84.3%	205/210 = 97.6%	61/62 = 98.4%
Extracellular	20(6)	13/20 = 65.0%	20/20 = 100%	6/6 = 100%
Inner Membrane	345(49)	325/345 = 94.2%	345/345 = 100%	49/49 = 100%
Outer Membrane	13(1)	10/13 = 76.9%	13/13 = 100%	1/1 = 100%
Periplasmic	49(7)	43/49=87.8%	48/49 = 98%	6/7 = 85.7%
Overall Accuracy		89.1%	99.1%	123/125=98.4%

This table reports the individual sensitivity measures (TP / (TP + FN)) for each class resulting from predictions generated using the independent test dataset assembled by Chou and Shen (Chou and

Shen, 2006), denoted as GP-test. The Sequences column indicates the number of sequences assigned to each localization class. The number in parenthesis represents the number of sequences in the test data that are not in the training data. The ngLOC column reports the results of the ngLOC method on the same test dataset. The ngLOC(UO) column reports the results of the ngLOC method on the sequences in GP-test that are not in the training data.

*(The Gneg-PLoc results are taken from the results reported by Chou and Shen (Chou and Shen, 2006))

We used ngLOC to generate predictions for all sequences in the *GP-test* dataset. As with the previous test, we separated out the 125 sequences that did not overlap with any sequence in the ngLOC training data. Again, our method performed extremely well, resulting in an overall accuracy of 99.1% versus 89.1% resulting from the Gneg-PLoc method. While we are certain that the high performance results are partially an artifact of the substantial number of sequences (512) that exist in both the training and test data, we were pleased with our 98.4% overall accuracy on the 125 sequences in the *GP-test* test dataset that were not in the ngLOC training data. These results suggest that the ngLOC model has an ability to discriminate between subcellular localizations on bacterial protein sequences that have yet to be seen by the method, with remarkable predictive performance that surpasses most existing methods today.

5.3.6 Generation of proteome-wide predictions

For our final test, we used the ngLOC method to generate predictions for the entire proteome of ten Gram-negative and five Gram-positive organisms. To generate proteome-wide localization predictions, we trained ngLOC-X (King and Guda, 2007) (see Section 4.2) using the Gram-negative training data (or Gram-positive, respectively), using a 6-gram model, and a confidence score threshold (*CStresh*) value of 25 (29 for Gram-positive) that allowed inclusion of all predictions except those of very low confidence. Both ngLOC and PSORTb v2.0 methods use a scoring threshold to reduce

false positives, and therefore predictions are not generated for the entire proteome. There are two reasons why using the CS threshold is important: first, it reduces the chances of generating predictions for sequences targeted to subcellular locations that are not covered by our method. For example, there are other minor subcellular locations to which a protein may be targeted in a Gram-negative cell, such as the flagellum or the fimbrium. Our method does not handle these locations due to the miniscule amount of experimentally determined localization data available for these locations. Second, sequences containing very low homology with respect to any other sequence in the training data will be hard to classify, and will likely result in a low CS. The proteome estimates shown in Table 26 and Table 27 are based on the proportion of sequences predicted with a CS of greater than or equal to *CStresh*.

Table 26 - Estimation of the subcellular proteome of ten Gram-negative organisms

<i>Species</i>	<i>Proteome</i>	ngLOC	PSORTb v2.0			ngLOC					PSORTb v2.0				
		% coverage	% coverage	% overlap	% agreement	% CYT	% EXC	% IN	% OUT	% PER	% CYT	% EXC	% IN	% OUT	% PER
<i>A. dehalogenans</i>	4346	79.8	57.7	48.3	78.3	76.18	0.46	21.94	0.61	0.81	62.36	0.8	30.3	3.07	3.47
<i>C. crescentus</i>	3737	76.8	53.9	43.1	81.3	71.99	0.63	24.49	0.77	1.08	61.67	0.7	28.95	5.21	3.48
<i>C. jejuni</i>	1628	70.0	55.0	41.6	83.3	62.25	1.23	32.92	1.67	1.93	57.43	0.67	32.18	6.26	3.46
<i>E. coli</i>	5323	71.5	56.3	43.1	83.2	48.11	3.39	35.71	5.59	7.2	60.19	1	29.78	4.2	4.83
<i>G. sulfurreducens</i>	3445	74.9	65.4	50.7	79.0	67.13	0.85	29.19	1.12	1.71	66.06	1.06	28.35	2.13	2.4
<i>H. influenzae</i>	1791	72.6	59.4	46.5	87.1	58.08	1.85	34.85	2.69	2.54	64.85	0.66	27.44	3.01	4.04
<i>H. pylori</i>	1575	67.2	54.6	39.6	84.3	61.38	1.7	33.62	1.23	2.08	61.4	1.4	27.67	8.02	1.51
<i>N. gonorrhoeae</i>	2002	76.0	52.1	42.1	81.4	63.91	1.97	27.55	2.5	1.97	68.55	0.19	25.22	3.45	2.59
<i>R. conorii</i>	1374	77.2	42.7	34.5	81.2	59.28	1.51	31.1	2.36	1.98	60.48	0.34	34.24	2.73	2.21
<i>T. denitrificans</i>	2827	81.3	59.4	49.6	80.8	69.63	0.48	26.15	0.83	1.83	63.75	0.36	29.35	3.27	3.27
Average		74.7	55.7	43.9	82	63.79	1.41	29.75	1.94	2.31	62.67	0.72	29.35	4.14	3.13

This table presents the location-wise percentages of the proteome predicted to localize into one of the five distinct organelles for ten Gram-negative organisms. The proteomes and corresponding PSORTb v2.0 predictions were downloaded from the PSORTdb online database of subcellular localization predictions for bacteria (Rey, et al., 2005). We compared our predictions against PSORTb v2.0 for the subset of predictions for which both methods generated predictions. The % *overlap* column indicates the fraction of the proteome in which both methods had a prediction. The % *agreement* column indicates the percent of the overlapping data that had identical predictions from both methods. The ngLOC predictions were generated with a 6-gram model, with a confidence score threshold of 25. CYT, cytoplasm; EXC, extracellular/secreted; IN, inner membrane; OUT, outer membrane; PER, periplasm.

Table 27 - Estimation of the subcellular proteome of five Gram-positive organisms

<i>Species</i>	<i>Proteome</i>	ngLOC	PSORTb v2.0			ngLOC				PSORTb v2.0			
		% coverage	% coverage	% overlap	% agreement	% CYT	% EXC	% IN	% WAL	% CYT	% EXC	% IN	% WAL
<i>M. tuberculosis</i>	4179	77.5	69.3	53.1	71.9	64.29	10.10	21.66	3.95	72.11	7.29	20.28	0.31
<i>S. pneumoniae</i>	2088	86.2	77.8	66.6	69.7	66.72	8.00	23.06	2.22	65.83	7.76	25.25	1.17
<i>B. anthracis</i>	5311	79.3	74.0	58.2	69.1	53.81	10.26	29.46	6.46	62.95	6.36	29.27	1.42
<i>C. acetobutylicum</i>	3848	76.2	70.8	54.7	71.9	60.80	9.34	24.81	5.04	65.31	3.78	29.77	1.14
<i>L. acidophilus</i>	1861	83.8	70.8	59.9	71.7	61.06	10.20	25.08	3.66	62.75	4.48	29.74	3.03
Average		80.6	72.5	58.5	70.8	61.34	9.58	24.81	4.27	65.79	5.93	26.86	1.42

This table presents the location-wise percentages of the proteome predicted to localize into one of the four distinct organelles for five Gram-positive organisms. The proteomes and corresponding PSORTb v2.0 predictions were downloaded from the PSORTdb online database of subcellular localization predictions for bacteria (Rey, et al., 2005). We compared our predictions against PSORTb v2.0 for the subset of predictions for which both methods generated predictions. The % *overlap* column indicates the fraction of the proteome in which both methods had a prediction. The % *agreement* column indicates the percent of the overlapping data that had identical predictions from both methods. The ngLOC predictions were generated with a 6-gram model, with a confidence score threshold of 30. CYT, cytoplasm; EXC, extracellular/secreted; IN, inner membrane; WAL, cell wall.

A large portion of the bacterial proteomes lack experimentally determined localizations, and therefore no real accuracy measure can be calculated at the proteome level. Instead, to establish a measure of credibility of our predictions, we report the fraction of the proteome covered by ngLOC-X and PSORTb v2.0, the fraction of the proteome for which predictions from both methods are available, and the fraction of this overlapping data that have identical predictions. Table 26 shows the results for the Gram-negative organisms, and Table 27 shows the results for the Gram-positive organisms.

We observed that ngLOC-X had a substantially larger coverage across all Gram-negative proteomes, with an average of 74.7% coverage, versus an average of 55.7% coverage by PSORTb. This represents a 34.1% increase in coverage by ngLOC-X, covering an additional 19% of the Gram-negative bacterial proteomes on average. Additionally, we are pleased to report that ngLOC-X predictions covered about 79% of the PSORTb predictions, where 82% of these have identical localizations predicted. The results for the Gram-positive bacterial proteomes are similar, with ngLOC-X resulting in an average coverage of 80.6% of the proteome, versus an average coverage of 72.5% by PSORTb, representing an 11.1% increase in coverage by ngLOC-X, covering an additional 8.1% of the proteome. The ngLOC-X predictions covered about 80.7% of the PSORTb predictions, where 70.8% of those have identical localization prediction.

The fractions of subcellular proteomes scaled consistently across most Gram-negative species for both methods (Table 26). The average subcellular proteome estimates are strikingly similar, with the largest difference observed between the estimates for the outer membrane, with ngLOC-X reporting 1.94% and PSORTb reporting 4.14%, respectively. However, there were significant departures for a few

species. The largest anomaly is the ngLOC-X estimate for *E. coli*, with only 48.1% of its proteome estimated to localize into the cytoplasm – a significantly lower estimate than average (63.8%), and also substantially lower than the estimate from PSORTb (60.2%). However, we also observed that 83.2% of the overlapping predictions for this species between these two methods are in agreement. One possible reason for this anomaly is that our training data contains a disproportionately higher number of *E. coli* sequences for non-cytoplasmic locations, resulting in over-prediction of the non-cytoplasmic locations, and vice-versa. The dominance of *E. coli* protein sequences in our training data is likely due to the role of *E. coli* as an established model organism, and to the successful experimental characterization of the subcellular localization of its proteome, compared to any other bacterium. The other significant anomaly was the substantially larger estimate for the outer membrane proteome by PSORTb for *C. crescentus*, *C. jejuni*, and *H. pylori* compared to the ngLOC-X estimate. For the rest of the organelles, both methods were relatively consistent across all locations.

The predicted proportions across different subcellular localizations of Gram-positive bacteria are also highly similar between ngLOC-X and PSORTb, though they are slightly more disproportionate than the Gram-negative predictions (Table 27). The two primary differences are in the subcellular proteome estimates for cell wall and extracellular space. The ngLOC-X method estimated an average of 4.27% of the proteome to localize to cell wall, versus an average of 1.42% estimated from PSORTb. The estimates for extracellular space were also larger from ngLOC-X, which predicted an average of 9.58% of the proteome to localize here, versus an average of 5.93% estimated from PSORTb. The estimates for the cytoplasm and the cytoplasmic membrane were more proportionate between the methods. The overall estimates for *S. pneumoniae* were

strikingly similar between the methods and also offered the most coverage and overlap of predictions between the methods.

5.4 Discussion

The ngLOC method uses the naïve Bayes classification method to model the probability that any given protein sequence will be targeted to a specific localization. As with any Bayesian method, the performance of ngLOC is partially dependent on the development of a good estimate for the prior probability distribution of an arbitrary sequence being localized to each localization class. Our method uses the observed distribution in the training data to estimate the parameters of the prior distribution. This accentuates the importance of ensuring that our training data is drawn in a way that is representative of the true underlying distribution of the data. Our training data were derived from the subset of sequences in the Swiss-Prot repository that have experimentally determined localizations annotated for bacterial sequences. Observing the class-wise distribution of our training data in Table 20 reveals the highly unbalanced nature of our data. This is expected, because the true distribution of proteins localized to each subcellular location is also unbalanced. Yet, we noticed that the data used by the PSORTb method are balanced (Gardy, et al., 2005; Gardy, et al., 2003), which is not representative of the natural, underlying distribution across subcellular classes. As shown in Table 26 and Table 27, the prediction coverage of the proteome by ngLOC-X for all species was significantly larger than that by PSORTb. We attribute these successful results to using a robust training dataset with a location-wise distribution that is closely representative of the true underlying proportions in a typical bacterial cell.

One significant benefit of the ngLOC method is its foundation on a probabilistic model. We believe that there is a significant lack of methods that generate probabilities or confidence scores with each prediction. It is valuable to know "how true" or "how false" a true or false prediction may be. Our method and the PSORTb method are among the very few methods available that generate a score associated with each prediction. By using a scoring threshold, the false positive rate can be significantly contained, thereby improving the precision of the classifier (Table 23).

We demonstrated how an ROC curve could be used as a visual graphical indicator to measure the ability for a classifier such as ngLOC to distinguish between correct and incorrect predictions. It is important to note that an ROC curve does not say anything directly about the overall accuracy of the classifier – this information is inferred. At face value, an ROC curve shows the ability of a classifier to rank positive instances relative to negative instances in the dataset according to a score generated for each instance (Fawcett, 2006). It is typically plotted only for binary classification problems, whereas in our classification task, we are working with 4 or 5 distinct classes, depending on the classification problem being addressed. We generate a single ROC curve, denoted as *Mac-ROC*, by macro-averaging the fraction of true positives and false positives over each class observed at distinct CS thresholds (Figure 12). We have not observed any other methods that have presented a multi-class ROC curve and analysis in this manner.

For the comparative test against other methods, the initial aim was to compare the ngLOC results with those of the PSORTb v2.0 method, a popular method for bacterial protein subcellular localization prediction (Gardy, et al., 2005). However, numerous high-confidence predictions were observed from ngLOC that disagreed with the annotation in the PSORTb training data. The ngLOC predictions for these proteins were

consistent with the annotation in Swiss-Prot. This problem was particularly noticeable with multi-localized proteins, which is partly due to the way that multi-localized proteins are defined in the literature. Although the training data used by PSORTb was originally taken from the Swiss-Prot database, their data was further subject to manual verification based on literature review (Gardy, et al., 2003). This process resulted in discrepancies in the interpretation of multi-localized proteins. For instance, the curators of the PSORTb dataset consider some membrane proteins to be multi-localized if a significant part of a protein (i.e. a domain) lies outside of the membrane (Brinkman, 2007). In contrast, the curators of Swiss-Prot database, which is the basis for our training data, annotate such proteins as ‘integral membrane proteins,’ since they are targeted to membranes. We concur with the annotation as defined by Swiss-Prot because we intend to use and predict localization information at the protein level, not at the domain level (as interpreted by PSORTb). Similar observations have been fully documented by Chou and Shen (Chou and Shen, 2006) and thus we do not delve into the topic here. However, due to the numerous inconsistencies in the annotation between our data and that of PSORTb, we determined that a comparative test could not be performed on the PSORTb dataset. The ngLOC method has the ability to use multi-localized proteins for training and generating multi-localized predictions (King and Guda, 2007). Nevertheless, there are far fewer bacterial proteins in Swiss-Prot annotated as multi-localized, thus limiting their use. Therefore, we did not attempt to generate any multi-class predictions in this study.

We showed a significant overlap in our proteome-wide predictions between the predictions generated by our method and the PSORTb method. We are pleased to see that the majority of the predictions were in agreement. However, we found several low-confidence predictions generated by ngLOC-X for which the PSORTb method generated

a high confidence prediction, and vice versa. This demonstrates the value of using such methods in tandem, in conjunction with experimental methods, in order to reduce the likelihood of false positives when understanding the localization of proteins. However, we also stress that functional attributes of protein sequences are annotated in a dynamic process; thus, some disagreements between the methods were likely due to differences in the time frame that the training data was acquired and in the methods used to assemble the training data.

5.5 Conclusions

Computational methods for predicting protein subcellular localization are vital for functional annotation of proteomes *en masse*. In our study, we curated a new dataset with subcellular annotation for Gram-negative and Gram-positive sequences. Using this new dataset, we extended the ngLOC method (King and Guda, 2007), a Bayesian classifier that can predict the subcellular localization of a protein. We demonstrated the performance of the ngLOC method through leave-one-out cross validation and direct comparison with similar methods, further validating its usefulness and superior performance. This method shows an overall accuracy of 89.7% and 89.3% for Gram-negative and Gram-positive bacteria protein sequences, respectively. As a probabilistic method, the method can generate a confidence score (CS) that places a measure of credibility on each prediction. We showed how this measure can be used to improve the precision of the method, as well as be used to generate a receiver operating characteristic (ROC) curve for individual classes and a single macro-averaged ROC curve covering all classes. We discussed the importance of micro-averaging and macro-averaging performance results in the context of unbalanced datasets such as those used in this study.

We used this method to annotate the subcellular proteomes of ten Gram-negative and five Gram-positive species that resulted in an impressive coverage of 74.7% and 80.6% of the proteomes in these species, respectively. To our knowledge, this study is the first to offer such a wide coverage for the estimation of bacterial subcellular proteomes.

6 Semi-Supervised Learning for Classification of Protein Sequence Data

Protein sequence data continue to become available at an exponential rate. Annotation of functional and structural attributes of these data lags far behind, with only a small fraction of the data understood and labeled by experimental methods. Classification methods that are based on semi-supervised learning can increase the overall accuracy of classifying partly labeled data in many domains, but very few methods exist that have shown their effect on protein sequence classification. We show how proven methods from text classification can be applied to protein sequence data, as we consider both existing and novel extensions to the basic methods, and demonstrate restrictions and differences that must be considered. We demonstrate comparative results against the transductive support vector machine, and show superior results on the most difficult classification problems. Our results show that large repositories of unlabeled protein sequence data can indeed be used to improve predictive performance, particularly in situations where there are fewer labeled protein sequences available, and/or the data are highly unbalanced in nature.

This chapter contains the work published in 2008 in Scientific Programming (King and Guda, 2008).

6.1 Introduction

Supervised methods for classification induce predictive models from labeled training data. Each instance in the training data is represented as a vector of feature values that

represent the instance, and a categorical assignment of the instance determined prior to model inference. Ideally, these methods aim to induce a model that can generalize over the entire domain space. This paradigm is no exception in supervised protein classification, where a model is inferred through analysis of training data consisting of proteins labeled with high-level, experimentally-determined annotation. The majority of publicly available protein data contain instances, represented by strings of letters, which identify the chain of amino acids that make up the protein. The string is determined over an alphabet of 20 characters, where each character represents a unique amino acid – a representation commonly known as the primary structure of a protein. There is rarely a problem finding protein data in their primary structure form, due to the exponential rate that these data are made available in public repositories, which far exceeds the rate of any other structural representation of proteins available. Fortunately, supervised learning methods for protein sequence classification are currently being researched and developed in earnest. However, methods to experimentally determine the function and/or structure of a protein are costly and time-consuming, resulting in large repositories of protein sequences with unknown function (i.e., unlabeled data). There is a considerable lack of methods that can make use of the comparatively large amount of unlabeled sequence data. Thus, this represents an ideal domain to explore *semi-supervised learning* methods, which are employed in situations where there are both labeled and unlabeled data available, and the amount of unlabeled data often excessively exceeds the amount of labeled data available (Chapelle, et al., 2006)

This chapter considers how semi-supervised learning can be used to induce a better protein classification model than supervised learning, primarily in situations where the target classes of interest have a relatively small amount of labeled protein data, and/or

there is low sequence similarity among labeled protein sequences. The primary set of target labels for empirical analysis will be drawn from the set of the most common subcellular localizations that a protein can localize into in order to perform its function. Our work presented is founded on our prior work completed on a supervised classification method for subcellular localization, called ngLOC (King and Guda, 2007). The semi-supervised extensions applied are founded on work completed by Nigam and colleagues, where semi-supervised learning methods based on expectation maximization (EM) techniques were applied to the problem of document classification (Nigam, et al., 2006; Nigam, et al., 2000). For comparative analysis, we present empirical results from application of transductive support vector machines (TSVMs), which also apply unlabeled data for improved model inference (Joachims, 1999). For our work, we adopt a dataset of proteins labeled with subcellular localization, though this work can be extended to any protein classification problem where entire sequences are labeled with some type of functional annotation.

This chapter is organized as follows. Section 6.2 will focus on the problem of learning from unlabeled protein sequence data in this context, and present the modifications to the ngLOC model to incorporate unobserved data, showing how it is theoretically possible to yield a better model than one learned from labeled data alone. A presentation of the EM algorithm is included that specifically presents the extensions to the ngLOC algorithm for semi-supervised learning. Section 6.3 presents a brief overview of the transductive SVM that is used for comparative analysis. Section 6.4 presents empirical results from our work, as well as comparative results from transductive SVMs. Section 6.5 offers a discussion regarding the practicalities of this work, and discusses differences between our work and other related work.

6.2 *Semi-supervised protein classification through EM*

In Section 4.1, it was shown how a mixture of multinomial models in a supervised, naïve Bayes framework can be used to classify proteins over 10 distinct subcellular localizations (King and Guda, 2007). A formal, yet brief, analysis of this model will be presented here, in order to effectively lay the foundation required for the semi-supervised extensions to be presented.

6.2.1 ngLOC - The Bayesian supervised model

Formally, we let $\mathcal{D}$ represent the protein space – the set of all possible proteins. The labeled data will be represented by D_L , and the unlabeled data will be represented by D_U , and $|D_L| \ll |D_U|$. D is the union of D_L and D_U . Let $\mathcal{C}$ denote the set of possible classes representing the discrete annotations that each protein may be annotated with, where $\mathcal{C} = \{c_j \mid j = 1, \dots, |\mathcal{C}|\}$. For our work, our target classes are represented by a set of major subcellular localizations that each protein may be localized to upon synthesis. Let $\mathcal{X}$ represent the feature space – the space of features that could be used to represent any instance in $\mathcal{D}$. The function that maps instances in $\mathcal{D}$ to features over $\mathcal{X}$ is assumed to be neither injective nor surjective; that is, there may be distinct instances in $\mathcal{D}$ that map to the same feature vector, *and* there may be feature vectors over $\mathcal{X}$ for which no instance in $\mathcal{D}$ could possibly exist in nature. The only assumption is that each unique instance in $\mathcal{D}$ maps to one feature vector over $\mathcal{X}$. Let $\mathbf{x}_i$ represent the vector of feature values representing instance $d_i \in D$, and y_i represent the class that d_i has been assigned to. Then, $D_L = \{(\mathbf{x}_i, y_i) \mid i = 1, \dots, |D_L|\}$ and $D_U = \{\mathbf{x}_i \mid i = 1, \dots, |D_U|\}$. We use the notation $D_{L,j}$ to represent the subset of the labeled data that belongs to class c_j . The notation $d_i \in D$ and $\mathbf{x}_i$

$\in D$ are both used in this research to refer to the same i^{th} instance in D ; the representation is implied in the notation.

In general, given any protein sequence d_i , the aim is to develop a probabilistic model of $P(c_j | d_i)$ for each class c_j . The classifier h will label each instance d_i according to the class model resulting in the highest probability. Equation (5.1) shows this in probabilistic terms, and shows how the well-known Bayes rule is used to derive an estimate for this probability:

$$h(d_i) = \arg \max_{c_j} P(c_j | d_i) = \arg \max_{c_j} \frac{P(d_i | c_j) P(c_j)}{P(d_i)} = \arg \max_{c_j} P(d_i | c_j) P(c_j) \quad (5.1)$$

The probability of $P(d_i)$ in the denominator is dropped, as it is independent of the class. Therefore, obtaining a model for classification involves obtaining models for the likelihood $P(d_i | c_j)$ and the prior probability $P(c_j)$.

We assume a known form for the probabilistic model prior to learning, with the parameters of the model for the likelihood denoted as ϕ_j , and the parameter for the class prior denoted as π . We let θ denote the entire set of all parameters for the model, $\theta = \{ \pi, \phi_j | j=1..|C| \}$. We assume the form of the model to be a generative, probabilistic mixture model, where a protein sequence is produced by selecting a generating component according to the prior class probabilities parameterized by π . The selected mixture component generates a protein sequence according to its own set of parameters ϕ_j for class c_j . We assume a one-to-one relationship between the components in the mixture model and the classes in which data may be assigned to (McCallum and Nigam, 1998).

The probabilistic form of the model will be understood through development of the feature space we use to represent protein sequences. The feature space $\mathcal{X}$ is developed in light of the significant work that has been accomplished in the field of

document classification. In a typical document classification model, $\mathcal{X}$ is constructed by considering all possible words that may appear throughout the entire set of documents. Here, we consider subsequences of a protein of fixed-length n as the equivalent of words in a document. In literature, these protein subsequences have been commonly called n -grams, n -mers, n -peptides, or simply words or subsequences of length n (Fofanov, 2004; Ganapathiraju, et al., 2002). Here, the term n -gram will be adopted.

Let Σ represent the set of all possible amino acids, $|\Sigma| = 20$. Let d_i be a sequence in D having k residues in length, where $d_i = (s_1 s_2 \dots s_k)$ and each $s_i \in \Sigma$. Letting d_i^n be the n -gram representation of protein sequence $d_i \in \mathcal{D}$, we have:

$$d_i^n = ((s_1 \dots s_n), (s_2 \dots s_{n+1}), \dots, (s_{k-n+1} \dots s_k))$$

where each $(s_j, \dots, s_{j+n-1})$ represents the n -gram in d_i starting at position j and continuing for n residues. Overlapping subsequences are used to capture as much information as possible about the dependencies between individual residues within the entire sequence. In protein classification tasks using the n -gram model, $\mathcal{X}$ is constructed by considering all possible n -grams that may occur in proteins. In an n -gram model, the size of the feature space grows exponentially with n , as $|\mathcal{X}| = |\Sigma|^n$. Fortunately, for large values of n (namely, $n > 5$), only a fraction of the theoretically possible n -grams actually occur in nature due to the evolutionary selection process, as a delicate mixture of various amino acid combinations are required to sustain a fold that allows the protein to carry out its designated function. Thus, we use a one-to-one mapping between unique n -grams and the set of integers to be used as indices. This requires memory allocation only for n -grams that occur in the training data, thereby allowing exploration of large values of n .

Let w_t represent the t^{th} n -gram in $\mathcal{X}$, and let random variable X_t indicate the event that n -gram w_t occurred N_{it} times in sequence d_i . We assume that the probability of such

an event occurrence is independent of the position in the sequence, and is independent of any other n -gram event occurrence (commonly called the naïve Bayes assumption).

Under these assumptions, the joint distribution of $X_1, X_2, \dots, X_{|\mathcal{X}|}$ follows a multinomial distribution for each class c_j , with parameters $\varphi_{1j} \dots \varphi_{|\mathcal{X}|j}$. Each class c_j has its own model, and thus has its own set of parameters, denoted $\boldsymbol{\varphi}_j$. (i.e. $\boldsymbol{\varphi} = \{ \boldsymbol{\varphi}_j \mid j = 1 \dots |\mathcal{C}| \} = \{ \varphi_{ij} \mid i = 1 \dots |\mathcal{X}|, j = 1 \dots |\mathcal{C}| \}$) Each sequence d_i is then represented as a vector of the number of occurrences of each n -gram w_t that occurred in the sequence. It can be shown through an easy derivation that a sequence will be classified according to the derivation shown in Figure 13.

$$\begin{aligned}
h(d_i) &= \arg \max_{c_j} P(c_j \mid d_i, \boldsymbol{\theta}) \\
&= \arg \max_{c_j} P(c_j \mid \boldsymbol{\pi}) (d_i \mid c_j, \boldsymbol{\varphi}_j) \\
&= \arg \max_{c_j} P(Y = c_j \mid \boldsymbol{\pi}) P(\mathbf{X} = \mathbf{x}_i \mid Y = c_j, \boldsymbol{\varphi}_j) \\
&= \arg \max_{c_j} P(Y = c_j \mid \boldsymbol{\pi}) \left(\prod_{t=1}^{|\mathcal{X}|} P(X_t = N_{it} \mid Y = c_j, \boldsymbol{\varphi}_j) \right) \\
&= \arg \max_{c_j} \pi_j \left(\prod_{t=1}^{|\mathcal{X}|} \varphi_{tj}^{N_{it}} \right)
\end{aligned}$$

Figure 13 - Derivation of naïve Bayes classifier

6.2.2 Parameter estimation

Let $\hat{\boldsymbol{\theta}}$ represent the estimate for the true model parameters $\boldsymbol{\theta}$. Typical naïve Bayes text classifiers determine parameter estimates through application of maximum likelihood (ML) methods or Bayesian maximum a posteriori (MAP) methods (Mitchell, 1997;

Nigam, et al., 2000). To estimate the class prior, we use a ML estimate (though a MAP estimate is acceptable, the difference between the two estimates is negligible due to the size of the data). The class prior estimates are calculated based on the proportion of the labeled training data belonging to that class:

$$\hat{\pi}_j = P(Y = c_j) = \frac{|D_{L,j}|}{\sum_k |D_{L,k}|}$$

The estimate of $\hat{\phi}_{ij}$, the probability of n -gram w_t occurring in class c_j , is determined empirically through observation of actual occurrences of w_t in the training data. This gives the maximized probability estimate based on the observed data. The ML estimate for probability of n -gram w_t occurring in class c_j is computed as the number of times w_t occurs in class c_j divided by the total number of n -grams in c_j . However, this can result in a probability of zero for some n -grams, as not all n -grams occur in all classes. The common solution that we adopt is to use a MAP estimate, which incorporates a prior distribution over the parameter space by adding a phantom occurrence of every n -gram in every class. This approach is commonly known as Laplace smoothing (Nigam, et al., 2000; Peng and Schuurmans, 2003). The estimate is given by:

$$\begin{aligned} \hat{\phi}_{ij} &= \frac{1 + \sum_{d_i \in D_{L,j}} N_{it}}{|\mathcal{X}| + \sum_{d_i \in D_{L,j}} (\text{Length}(d_i) - n + 1)} \\ &= \frac{1 + (\text{Count}(w_t) \text{ in class } c_j)}{|\mathcal{X}| + (\text{Total } n\text{-grams in class } c_j)} \end{aligned}$$

6.2.3 Incorporating unlabeled data through EM

The expectation maximization (EM) algorithm is used in situations where ML or MAP parameter estimates of a probabilistic model are needed, yet the model consists of unobserved or latent variables (Dempster, et al., 1977). The latent variables can represent missing data or missing values of existing data. In our case, the latent variables represent the missing labels in our unlabeled data D_U . We present a generalized EM approach applied to our problem. We follow a derivation similar to that presented in numerous resources covering EM in machine learning literature (Chapelle, et al., 2006; Mitchell, 1997; Nigam, et al., 2000; Seeger, 2001). EM has been shown to perform well in many real-world scenarios, including various protein-related classification tasks, discovery of molecular pathways (Segal, 2003), mining interaction data (Deng, et al., 2002), and prediction of protein families (Bejerano and Yona, 2001), among many other related tasks.

Learning can be viewed as attempting to maximize an objective function. Our objective is to find the most probable parameter estimates for our probabilistic model, given the evidence of the observed data $D = D_L \cup D_U$, formally stated as $\hat{\theta} = \arg \max_{\theta} P(\theta | D)$. Using the MAP method to find the parameter estimates, we assume that a prior distribution over θ exists, and then apply Bayes rule to this expression, yielding

$$\begin{aligned}\hat{\theta} &= \arg \max_{\theta} P(\theta | D) \\ &= \arg \max_{\theta} P(\theta) P(D | \theta)\end{aligned}$$

It would be desirable to simply maximize the function through determination of the partial derivative with respect to θ , which necessitates taking the log of both sides. However, our model contains latent variables, which makes straightforward

maximization infeasible. EM solves this problem through maximization of the expected value of the latent data by finding a locally optimal set of parameters in an iterative, hill-climbing fashion.

We first derive a form for the complete joint log-likelihood of the observed and latent data as $l_c(\boldsymbol{\theta}; D) = \sum_{d_i \in D} \log P(d_i | \boldsymbol{\theta})$. We let random variable $\mathbf{X}$ represent the occurrence of the feature vector $\mathbf{x}_i$ for each given data instance d_i , and random variable Y represents the target class. Y is latent for the unlabeled data only. In semi-supervised learning, our complete log-likelihood function is defined as:

$$\begin{aligned} l_c(\boldsymbol{\theta}; D) &= \log P(\boldsymbol{\theta}) + \sum_{d_i \in D_L} \log P(\mathbf{X} = \mathbf{x}_i, Y = y_i | \boldsymbol{\theta}) \\ &\quad + \sum_{d_i \in D_U} \log \sum_{c_j \in \mathcal{C}} P(\mathbf{X} = \mathbf{x}_i, Y = c_j | \boldsymbol{\theta}) \end{aligned}$$

We need to compute the expected value of the complete log-likelihood function with respect to the latent variables. We know from elementary statistics that the expectation of a sum is equal to the sum of an expectation. Thus, for clarity, we will deal with the expectation of a single instance. To further simplify the derivation, we will assume that Y is unobserved for all data. The fact that Y is labeled for a small subset of our data will only improve the parameter estimates, and does not harm our derivation in any way:

$$\begin{aligned} E_Y(l_c(\boldsymbol{\theta}; \mathbf{x}_i)) &= E_Y(\log P(\mathbf{X} = \mathbf{x}_i | \boldsymbol{\theta})) \\ &= E_Y\left(\log \sum_{c'_j \in \mathcal{C}} P(\mathbf{X} = \mathbf{x}_i, Y = c'_j | \boldsymbol{\theta})\right) \\ &= E_Y\left(\log \sum_{c'_j \in \mathcal{C}} P(Y = c'_j, \boldsymbol{\theta}) P(\mathbf{X} = \mathbf{x}_i, Y = c'_j | \boldsymbol{\theta})\right) \\ &= E_Y\left(\log E_Y(P(\mathbf{X} = \mathbf{x}_i, Y, \boldsymbol{\theta}))\right) \\ &\leq E_Y\left(E_Y \log(P(\mathbf{X} = \mathbf{x}_i | Y, \boldsymbol{\theta}))\right) \end{aligned}$$

We observe that we have a log of a sum, which makes computing any partial derivative for maximizing purposes intractable. We can apply Jensen's inequality, which gives us a lower-bound on the expectation. (This is part of the reason why local maxima are obtained for the parameter estimates during the maximization step.) More importantly, we clearly see the necessity of the iterative, divide-and-conquer approach of EM due to the recurrence of the expectation. Each E-step computes the expected values of the class assignments based on the current parameter estimates, and each M-step uses the expected values to compute better parameter estimates. This continues until the observed change in the complete log-likelihood of the data between successive iterations falls below a predetermined threshold.

We introduce the notation $\hat{\mathbf{y}}$ to represent the estimated class assignments for D , where $\hat{\mathbf{y}}_i$ represents the expected class assignments for instance $\mathbf{x}_i$. By holding the current expected values constant, we can calculate new parameter estimates using supervised learning methods. The complete log-likelihood is restated, indicating the dependence on the estimated class assignments:

$$l_c(\boldsymbol{\theta}; D, \hat{\mathbf{y}}) = \log P(\boldsymbol{\theta}) + \sum_{d_i \in D_L} \log P(\mathbf{X} = \mathbf{x}_i, Y = y_i; \boldsymbol{\theta}) \\ + \sum_{d_i \in D_U} \sum_{c_j \in \mathcal{C}} \log P(\mathbf{X} = \mathbf{x}_i, Y = c_j; \boldsymbol{\theta}, \hat{\mathbf{y}}_i)$$

We use the notation $\hat{\mathbf{y}}_i^{(k)}$ to denote the vector of new expected class assignment estimates computed for instance $\mathbf{x}_i$ after the k^{th} iteration of EM. The vector $\hat{\mathbf{y}}_i^{(k)}$ has $j = 1..|\mathcal{C}|$ entries, where each entry represents $P(Y = c_j | \mathbf{X} = \mathbf{x}_i; \hat{\boldsymbol{\theta}})$, the estimated probability that instance $\mathbf{x}_i$ belongs to class c_j , given the instance and the current parameter estimates. We let $\hat{\boldsymbol{\theta}}^{(k)}$ denote the current parameter estimates after performing the k^{th} iteration of EM. Our two steps precisely consist of:

E-step: Compute $\hat{\mathbf{y}}_i^{(k)} = \left(\forall c_j \in \mathcal{C}, P(Y = c_j | X = \mathbf{x}_i; \hat{\boldsymbol{\theta}}^{(k-1)}) \right), \forall \mathbf{x}_i \in D_U$

M-step: Compute $\hat{\theta}^{(k+1)} = \arg \max_{\theta} P(\theta | D; \hat{y}^{(k)})$

Notice that we compute the expected class assignments of the unlabeled data only. For the labeled data, the vector $\hat{y}_i^{(k)}$ is a vector of zeros except for its labeled class entry, in which it is a one. This can only improve the estimates derived in the M-step, because we know the true class assignments for a portion of our data. Figure 14 displays the basic EM algorithm.

Basic-EM(D_L, D_U, δ)

Input: D_L – Set of labeled protein sequence data
 D_U – Set of unlabeled protein sequence data

Output: Classifier h with parameter estimates $\hat{\theta}$

1. Train initial naive Bayes classifier using D_L , deriving MAP parameters estimates to find $\hat{\theta}^{(0)} = \arg \max_{\theta} P(\theta | D_L)$
2. $k = 0$
3. **REPEAT**
 - a. **E-step** – Use classifier parameterized by $\hat{\theta}^{(k)}$ to determine new expected class assignments $\hat{y}^{(k)}$ for unlabeled data.
 - b. **M-Step** – Use D_L , current expected class assignments $\hat{y}^{(k)}$ for D_U and current parameters $\hat{\theta}^{(k)}$ to compute new parameter estimates $\hat{\theta}^{(k+1)}$
 - c. $k = k + 1$

UNTIL $l_c(\hat{\theta}^{(k)}; D, \hat{y}^{(k)}) - l_c(\hat{\theta}^{(k-1)}; D, \hat{y}^{(k-1)}) < \delta$

4. Return parameter estimates

Figure 14 - Basic EM Algorithm

6.2.4 Extending the EM algorithm

We describe three extensions applied to the basic EM algorithm that improved classification performance under most circumstances. The enhancements are designed to overcome the limitations brought about by the stated assumptions of our model, to incorporate a priori biological knowledge into the classifier, and to select better instances from the unlabeled data to improve predictive results.

Weighted EM using EM- λ

The first extension is called EM- λ (Nigam, et al., 2000). It adds a new parameter λ to control the extent to which the expected class assignments of the unlabeled data adjusts the new parameter estimates. The λ parameter weights the unlabeled data by a factor of λ when computing new parameter estimates, where ($0 < \lambda \leq 1$). When $\lambda = 1$, each unlabeled protein sequence will have the same effect as labeled sequences, which is the equivalent to the basic EM algorithm. The resulting complete log-likelihood is modified as follows:

$$l_c(\boldsymbol{\theta}; D, \hat{\mathbf{y}}) = \log P(\boldsymbol{\theta}) + \sum_{d_i \in D_L} \log P(\mathbf{X} = \mathbf{x}_i, Y = y_i; \boldsymbol{\theta}) \\ + \lambda \left(\sum_{d_i \in D_U} \sum_{c_j \in \mathcal{C}} \log P(\mathbf{X} = \mathbf{x}_i, Y = c_j; \boldsymbol{\theta}, \hat{\mathbf{y}}_i) \right)$$

This option may be useful in situations where the amount of unlabeled data significantly dominates the labeled data to the point of skewing the reasonable initial estimates formed by the initial step in the EM algorithm.

Multi-EM – Multiple Mixture Components per Class

The second enhancement we made to the basic EM algorithm is called Multi-EM, and is based on relaxing the assumption that there is a one-to-one correspondence between the components in the mixture model and the classes (Nigam, et al., 2006; Nigam, et al., 2000). There is biological justification for relaxing this assumption, particularly when developing a model for subcellular localization. When a protein sequence is synthesized, it is targeted for one or more organelles where it will carry out its function. However, a few organelles in the cell are made of several distinct subcompartments, or suborganelles.

For example, the mitochondria has an outer membrane that encloses the structure inside the cell. Below this, there is an inner membrane, which separates the matrix region of the organelle from the intermembrane space. These represent four distinct suborganelles of mitochondria, and proteins that localize to the mitochondria actually localize into one of these suborganelles. Unfortunately, most protein sequences that have subcellular localization annotated are done so only at the organelle level, due to the difficulty of experimentally determining localization of a protein at this level of granularity.

However, we can use this biological knowledge and enhance our mixture model to assume that each component represents a suborganelle, and then organize the component structure in the model a priori to group together components at the level of the organelle. In other words, we can assume there are some number of generating components belonging to each class, which relaxes the restricted one-to-one correspondence and assumes a many-to-one relationship between components and classes.

We introduce a new latent variable, denoted Z , into our model for each data member, which represents the suborganelle in which each protein sequence is localized. Note that this is unobserved for both labeled and unlabeled data. Once again, we can use the EM algorithm to estimate the parameters over latent variable Z using the same method as the basic EM algorithm, with the exception that the expected subclass assignments are going to be generated for D_L and D_U during each iteration – whereas the basic EM algorithm generated expected class assignments only for the unlabeled data D_U . The primary difference is in the interpretation of the probability of the target class. Because we know a priori the relationship between components and target classes, the probability of a given class will simply be the sum of the probabilities of each of the components belonging to that class.

Let $\mathcal{Z}$ represent the total set of components (i.e., suborganelles) in the mixture model, and $z_a \in \mathcal{Z}$ will represent one distinct component in the model. We will use the notation $\mathcal{Z}[c_j]$ to represent the subset of components that are assigned a priori to class c_j . We still use $\boldsymbol{\theta}$ to refer to the parameter space, but now $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\varphi}_a \mid a = 1 \dots |\mathcal{Z}|\} = \{\varphi_{ta} \mid t = 1 \dots |\mathcal{X}|, a = 1 \dots |\mathcal{Z}|\}$. Under these modifications, the probability of class c_j , given a sequence, is defined as the sum of the probabilities of all components that were assigned to the class. Simplifying the derivation for classification shown in Figure 13, we assign a class to a new protein sequence as follows:

$$h(d_i) = \arg \max_{c_j} \sum_{z_a \in \mathcal{Z}[c_j]} \left(\pi_a \prod_{t=1}^{|\mathcal{X}|} \varphi_{ta}^{N_{it}} \right)$$

The modifications required to the EM algorithm and underlying model to support this extension are minor, provided that the a priori assignments of components to individual class assignments are maintained throughout. The algorithm is presented in Figure 15. For the labeled data, we restrict the expected component membership assignments to only those components belonging to the class in which d_i has been assigned to, and always renormalize over only these classes. We let $\hat{\mathbf{z}}$ represent the vector of expected component assignments computed during each iteration of EM, where each entry in the vector represents the probability that a given sequence belongs to that component. To calculate class membership for class c_j , we simply sum the vector entries representing the components assigned to class c_j . The rest of the classification and EM algorithm proceeds in the same manner.

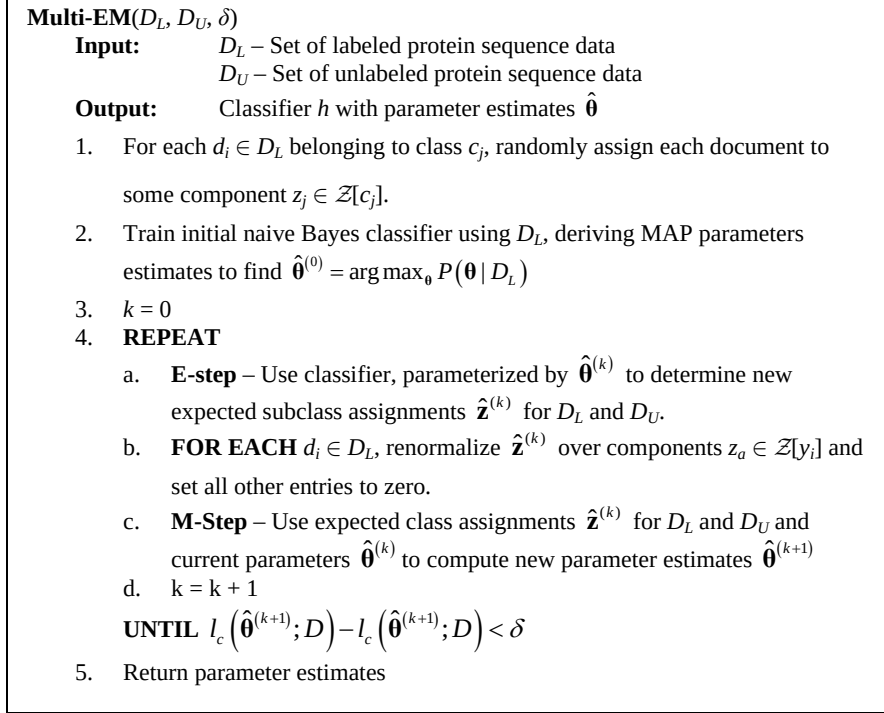


Figure 15 - Algorithm for Multi-EM

EM-CS – Improved Selection of Unlabeled Data

Our third, novel extension to the basic EM algorithm is designed to intelligently select better examples from the unlabeled dataset to use for determining better parameter estimates. A smaller set of unlabeled data will allow for improved running time of the EM algorithm. Moreover, carefully chosen examples from D_U may yield an improvement in predictive performance. Most protein classifiers are designed to predict a limited number of classes over all of the possible functional annotations that could be considered. However, the reality is that the unlabeled protein data D_U will likely contain sequences that are not related to any of the classes that are being predicted. For example, if we are developing a binary classifier to predict whether a sequence will localize to the nucleus or cytoplasm, our training dataset D_L will contain only sequences targeted to

those organelles. However, it must be expected that data in D_U is going to contain sequences targeted to other organelles in the cell. It would be beneficial if we could select sequences from the unlabeled data that have a higher likelihood of being organized to one of the organelles of interest, and only include those sequences in the unlabeled data.

We implemented an extension to the basic EM algorithm called EM-CS, which uses our confidence score (CS) to select the proteins from the unlabeled data that achieve a CS score $\geq CStresh$. This subset of proteins is then further reduced by selecting the number of instances in each class to be in accordance with the current estimates for π – the class prior parameters. These selected proteins are then used during the M-step of the algorithm to determine better parameter estimates. Sequences are allowed to be added to the selected subset during each iteration. This will allow the improved parameter estimates determined in each E-step to iteratively expand the n -gram feature space. This is in contrast to the basic EM algorithm, which incorporates all sequences in D_U from the very first iteration, thereby incorporating many features that are unlikely to be observed because the features are significant for data belonging to other classes. By assuming a confidence score threshold, we increase the likelihood that only sequences associated with one of the classes of interest are included in the model. If we can selectively add sequences that are related to one of the classes of interest, then it should increase the likelihood that discriminatory information about n -grams in the test data will be determined. The algorithm is given in Figure 16.

EM-CS($D_L, D_U, CStresh, \delta$)

Input: D_L – Set of labeled protein sequence data
 D_U – Set of unlabeled protein sequence data
 $CStresh$ – Minimum confidence score allowed on D_U sequences

Output: Classifier h with parameter estimates $\hat{\theta}$

1. Train initial naive Bayes classifier using D_L , deriving MAP parameters estimates to find $\hat{\theta}^{(0)} = \arg \max_{\theta} P(\theta | D_L)$
2. $k = 0$
3. **REPEAT**
 - a. **E-step** – Use classifier, parameterized by $\hat{\theta}^{(k)}$, to determine new expected class assignments $\hat{y}^{(k)}$ for D_U
 - b. **FOR EACH** $d_i \in D_U$, flag sequences whose confidence score $\geq CStresh$. Select sequences in proportion to the current class probabilities $P(c_j | \hat{\theta}^{(k)})$
 - c. **M-Step** – Use D_L , only flagged sequences in D_U and their current expected class assignments $\hat{y}^{(k)}$, and current parameters $\hat{\theta}^{(k)}$ to compute new parameter estimates $\hat{\theta}^{(k+1)}$
 - d. $k = k + 1$
- UNTIL** $l_c(\hat{\theta}^{(k)}; D) - l_c(\hat{\theta}^{(k-1)}; D) < \delta$
4. Return parameter estimates

Figure 16 - EM-CS algorithm

6.3 Transductive SVMs as a comparative method

The transductive support vector machine (TSVM) will represent the method used for comparison against the EM-based methods presented in this study. As noted previously, there are very few applications in bioinformatics that use semi-supervised and transductive inference methods. However, the inductive support vector machine (SVM) is perhaps the most widely used classification model in this field. We present a very brief introduction to the SVM and TSVM methods here to give the interested reader an understanding of the methods used for comparison purposes. We encourage the interested reader to consult the vast number of available resources on SVM and kernel methods for further information (Vapnik, 1995; Vapnik, 1998)

The support vector machine (SVM) is a discriminative, supervised classification method that aims to find a hyperplane $h(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ that maximally separates positive and negative instances of the input data, known as the maximum-margin hyperplane (Cortes and Vapnik, 1995). (Quite often, the data are projected into a higher dimensional space through applying the kernel trick to the input vector (Vapnik, 1995), but this is beyond the scope of this work.) The data instances that are closest to the hyperplane limit the size of the margin, and are known as the support vectors. Letting $\mathbf{x}_i$ represent the vector of feature values for instance d_i , and y_i represent the class that d_i belongs to, and $y_i \in \{-1, 1\}$, the SVM learning method involves solving an optimization problem under a specified set of constraints:

$$\begin{aligned} \text{Minimize : } & \frac{1}{2} \|\mathbf{w}^2\| + C \left(\sum_i \xi_i \right) \\ \text{Subject to : } & y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad (1 \leq i \leq |D_L|) \end{aligned} \quad (5.2)$$

The standard SVM algorithm is referred to as a hard-margin SVM, and requires the data to be linearly separable. Alternatively, soft margin SVMs are frequently used when data may not be linearly separable, in which case slack variables ξ_i are introduced, as presented in equation (5.2). These slack variables measure the degree that instance $\mathbf{x}_i$ is misclassified and allow a maximally separating hyperplane to be found even when there are instances that are misclassified.

The transductive SVM attempts to improve the predictive performance of SVMs by using unlabeled data (Joachims, 1999; Vapnik, 1995). As is the case with inductive SVMs, a maximal separating hyperplane is learned from labeled data, but extra constraints are introduced to ensure that the hyperplane is maximally separating unlabeled data as well. Letting $\mathbf{x}_j^*$ denote the j^{th} instance in unlabeled dataset D_U , the aim is now to find a labeling y_j^* for each datum $\mathbf{x}_j^*$ in the unlabeled dataset, and hyperplane

parameters $\mathbf{w}$ and b such the hyperplane separates the labeled and unlabeled data with maximum margin. Slack variables are introduced for the unlabeled data as they were for labeled data in (5.2) in the event that the unlabeled data are non-separable. The optimization is now formulated as:

$$\begin{aligned} \text{Minimize: } & \frac{1}{2} \|\mathbf{w}^2\| + C \left(\sum_i \xi_i \right) + C^* \left(\sum_j \xi_j^* \right) \\ \text{Subject to: } & y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad (1 \leq i \leq |D_L|) \\ & y_j^* (\mathbf{w} \cdot \mathbf{x}_j^* + b) \geq 1 - \xi_j^* \quad (1 \leq j \leq |D_U|) \end{aligned}$$

The parameters C and C^* are user-defined parameters set before learning commences. The C parameter allows trading off margin size against misclassified data, and the C^* parameter penalizes unlabeled data that is inside the margin between the support vectors.

6.4 Results

In this section, we present our results showing how unlabeled data can be used to improve performance over models inferred using labeled data alone. Our tests are designed to replicate many real-world situations that occur in protein classification, including limited labeled data available and unbalanced classes. Results will indicate that the smaller the amount of labeled data observed initially, the more dramatic the effect that unlabeled data can have on improving the performance of the classifier. We use the problem of subcellular localization prediction as the platform in which to test the method. The dataset used for the various tests, the test configurations, and the measures used to critically assess the performance of the classification are all presented here.

For comparative purposes, we present the results of using a transductive SVM implementation called UniverSVM, which integrates multi-class training and prediction, incorporates cross validation in generating predictive results, and is optimized for large

scale datasets (Collobert, et al., 2006). We chose a linear kernel because it outperformed the other kernels on these tests. We varied the C and C^* parameters over a wide range of selections, and chose the best results for comparison purposes. We modified the original code to include reporting of our standard performance measures over multiple classes.

6.4.1 Datasets

We use varying subsets of possible subcellular localizations as the set of target classes that the data are to be classified as. Our datasets are configured specifically to address common challenges in datasets used for protein classification. Though we use subcellular localization as our target class labels, this work can be applied to any protein classification problem where the entire sequence is to be labeled.

For this work, we started with the same dataset that was used in our ngLOC method (King and Guda, 2007). The set of protein sequences were taken from the Swiss-Prot database, release 50.0 (May 2006), that contains experimentally determined annotations on subcellular localizations (Boeckmann, et al., 2003). The following filters were applied to obtain high-quality data for training and testing purposes: (i) only eukaryotic, non-plant sequences were considered, (ii) sequences with predicted or ambiguous localizations were removed, (iii) sequences shorter than 10 residues in length were removed, (iv) all redundant sequences were removed, and (v) sequences known to localize in multiple locations were manually checked and sorted to avoid errors caused by automated keyword-based sorting.

To reduce the similarity of sequences in the dataset, we used the cd-hit clustering software (Li and Godzik, 2006) to reduce the dataset to a maximum sequence identity of 60%. Using this clustered dataset, we created three separate datasets. Our tests

sometimes use randomly selected fractions of these datasets to compare how differing amounts of labeled data affect the predictive performance of the classifier. It is understood that sequence similarity is certainly much less than 60% in these fractions selected, though the similarity was not directly measured. Our three primary dataset configurations are presented in Table 28.

Table 28 – Labeled datasets of proteins over subcellular localizations

Organelle	Code	D_L-2A	D_L-2B	D_L-3
Extracellular	EXC	2,745		
Golgi Apparatus	GOL			378
Lysosome	LYS			80
Mitochondria	MIT		1,155	
Nuclear	NUC		3,814	
Plasma Membrane	PLA	2,596		
Perioxosomes	POX			126
TOTAL:		5,341	4,969	584

Our simplest test dataset, D_L-2A, is a balanced, two-class subset of sequences annotated to be localized to either the extracellular space (EXT) or to the plasma membrane (PLA). These two classes are chosen because they both have close prior class distributions in the original dataset. From a biological perspective, they represent two organelles of particular interest in the biomedical research community, as proteins that are targeted to these regions are interesting for drug targeting purposes. The second dataset, D_L-2B, is designed to evaluate the Multi-EM algorithm. These two classes are chosen because they represent organelles in the cell that are known to have multiple suborganelles, and they have an unbalanced distribution. The third dataset, D_L-3, was designed to test the classification of sequences where the target classes represent a small fraction of all possible classes that the data may be annotated with. For this set, we

created a three-class dataset of sequences localized into the golgi apparatus (GOL), lysosomes (LYS), and perixosomes (POX). These classes are unbalanced, and represent some of the smallest organelles in the cell. (As presented in the ngLOC method, these three classes combined represent only 2.3% of the entire original dataset (King and Guda, 2007).) Thus, the annotated data available for these organelles is relatively limited. Location-wise distribution of each of these datasets is shown in Table 28.

Our unlabeled dataset D_U was taken from a set of 76,997 eukaryotic, non-plant protein sequences downloaded from the Swiss-Prot database (Boeckmann, et al., 2003). We filtered out sequences that did not belong to eukaryotic, non-plant species prior to classification to reduce the likelihood of using data that belonged to classes outside of our domain. For example, plant cells have proteins localized to the chloroplast and vacuole, which are organelles that typically do not exist in eukaryotic animal cells. We removed all sequences in D_U that already exist in the set of labeled data before beginning each test.

6.4.2 Performance measures

We evaluated the performance of sequences labeled with single classes through calculation of precision, recall (sensitivity), and the F_1 measure. The recall for class c_j , denoted s_j , is defined as the fraction of proteins belonging to class c_j that were predicted correctly. The precision for class c_j , denoted p_j , is defined as the fraction of proteins predicted to belong to class c_j that were correct predictions. We define the following:

- TP_j – Number of correctly predicted sequences belonging to class c_j
- FN_j – Number of incorrectly predicted sequences belonging to class c_j
- TN_j – Number of correctly predicted sequences not belonging to class c_j
- FP_j – Number of incorrectly predicted sequences not belonging to class c_j

In these terms, we define s_j and p_j as:

$$s_j = \frac{TP_j}{TP_j + FN_j} \quad p_j = \frac{TP_j}{TP_j + FP_j}$$

The F_1 measure is a per-class measurement that combines recall (sensitivity) and precision on the specified class in the calculation. The F_1 measure for class c_j is defined as:

$$F_{1j} = \frac{2s_j p_j}{s_j + p_j}$$

For determining overall classifier performance, we report micro-averaged and macro-averaged F_1 , denoted $micF_1$ and $macF_1$ respectively. These measures are used widely in evaluation of methods for text classification, particularly in research on unbalanced data (Yang, 1999). They give a single overall measure of classifier performance, particularly in the midst of unbalanced, multi-class data. With micro-averaged scores, the measures are produced by totaling the appropriate counts with respect to each class, and then calculating the measure. In this way, an equal weight is given to every protein instance, regardless of class. Micro-averaged scores tend to be dominated by the performance of the large classes in the data. In comparison, macro-averaged scores are calculated on a per-class basis first, and then the measure is averaged across all classes to produce the final measure. Here, an equal weight is given to every class, regardless of how often instances in any specific class occur in the data. Macro-averaged scores are dominated by the performance of the classifier on rare classes (Yang and Liu, 1999). Because we are particularly interested in improving predictions on the smaller represented classes, we focus mostly on macro-averaged F_1 . It provides the most honest measure of performance across all classes without artificially inflating the performance measure due to over-predictive behavior of the classifier on the large classes. However, we report micro-averaged F_1 for completeness. It can be easily shown

that micro-averaged F_1 is equivalent to the total accuracy for the classifier. However, because the majority of our data are unbalanced, we believe that micro-averaged measures, such as $micF_1$, commonly used in a wide array of research on classification, fail to reveal the performance under this situation.

All performance measures are based on a standard 10-fold cross validation, where the dataset is broken up into 10 randomly selected disjoint partitions. The model is trained with nine parts and tested with the remaining part. This is repeated over all 10 possible combinations.

6.4.3 Empirical results

We present the results of different experiments conducted using our datasets that show how unlabeled data can be used to improve predictive performance over a variety of scenarios encountered in protein classification. We will state the value of n for the n -gram length chosen for each test. It is important to choose n in such a way that the classifier is able to generalize beyond the training data. If you have access to large amounts of labeled data, then larger values of n (typically $n \geq 5$) will yield better results on new data yet to be seen. If n is too large, then the feature space will consist of unique n -grams that were observed in the training data, thereby overfitting the training data. If n is too small, then the feature space will be too restrictive for learning discriminative features in the data (King and Guda, 2007). Because we reduce the similarity to a maximum threshold of 60% sequence identity in our data, and because we randomly select small subsets of the dataset, the actual level of similarity in most tests will be much less than 60%, and thus smaller values of n (usually ranging from 4 to 6) will be used. The optimal value of n is chosen based on 10-fold cross validation using the ngLOC

method without unlabeled data. All subsequence tests with unlabeled data are based on a 10-fold cross validation unless otherwise stated.

Basic-EM Results

Our first test was designed to confirm the usefulness of the basic EM approach. We tested the D_L -2A dataset over different sizes of labeled data, using a fixed-size set of 25,000 unlabeled data sequences selected at random from D_U . An n -gram size of 6 was chosen for this test. Figure 17 shows a graph of the macro-averaged F_1 measure with and without unlabeled data being used. Table 29 reports both the macro-averaged F_1 ($macF_1$) and micro-averaged F_1 ($micF_1$) measures generated on different fractions of the D_L -2A dataset that result with and without 25,000 unlabeled data sequences.

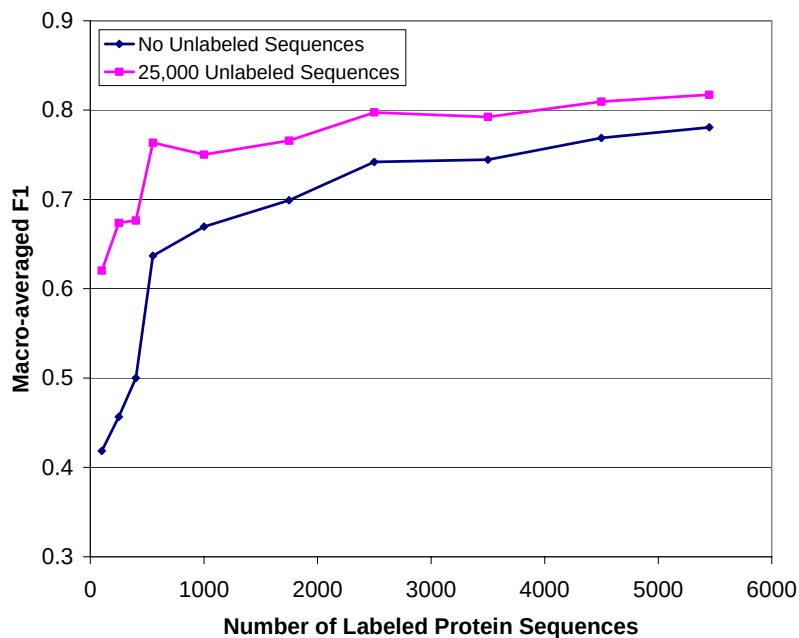


Figure 17 - Basic EM Algorithm on 2-class dataset

Table 29 - Performance improvement on 2-class data.

# Labeled Seqs	<i>No UL Data</i>		<i>25,000 UL Seqs</i>	
	<i>macF₁</i>	<i>micF₁</i>	<i>macF₁</i>	<i>micF₁</i>
100	0.418	0.449	0.620	0.622
250	0.457	0.474	0.674	0.677
400	0.500	0.517	0.676	0.676
550	0.637	0.649	0.764	0.764
1000	0.670	0.680	0.750	0.752
1750	0.699	0.717	0.766	0.772
2500	0.742	0.748	0.797	0.799
3500	0.744	0.752	0.792	0.795
4500	0.769	0.775	0.810	0.812
5450	0.781	0.787	0.817	0.820

This chart displays how different fractions of the unlabeled data show remarkable improvement as a result of applying the basic EM algorithm with 25,000 unlabeled sequences.

The EM algorithm used with the unlabeled data results in significantly improved $macF_1$ and $micF_1$ measures for all fractions of the labeled data used. As expected, the improvement is much more dramatic when only a small fraction of the unlabeled data are used. When there are only 100 labeled sequences, $macF_1$ improves from 0.418 to 0.62 and $micF_1$ improves from 0.449 to 0.622, resulting in a remarkable 48% and 38% improvement in the measures, respectively. However, even when all 5,450 labeled sequences are used, we still observe an improvement in the $macF_1$ measure, increasing by 4.6% from 0.781 to 0.817. Additionally, $micF_1$ improved 4.2% from 0.787 to 0.82. While these performance improvements are small, it shows that this approach is useful even when the amount of labeled data is relatively large. This behavior suggests that as the number of labeled sequences increase in the model, there are fewer discriminatory n -grams to be discovered in the data, and may justify how the improvement in predictive performance decreases proportionally to the amount of unlabeled data used.

EM- λ Results

It is possible to overwhelm the classifier with too much unlabeled data, resulting in skewed parameter estimates for the model which becomes detrimental to predictive performance (Nigam, et al., 2000). We chose a random selection of 250 sequences from the D_L-2A, and using an n -gram length of 4, we applied the basic EM algorithm while varying the fraction of unlabeled data. Table 30 reports the results, and shows how both $macF_1$ and $micF_1$ begin to deteriorate when the unlabeled data becomes increasingly large. For this particular test, we observed the best performance when applying 25,000 unlabeled sequences. At this point, we observed a $macF_1$ measure of 0.674, a 47% improvement from 0.457, that was achieved without using any unlabeled data. Using more unlabeled data caused the performance to decrease, dropping by more than 4% from the best observed measure for this test. This shows that it is possible to overwhelm the model parameter estimates with too many expected predictions from the unlabeled data. It is for this reason that the EM- λ modification was made, to reduce the effect that the unlabeled data will have on determining the parameter estimates.

Table 30 - Effect of unlabeled data on classification of 250 sequences from the D_L-2A dataset.

<i># Unlabeled</i>	<i>macF₁</i>	<i>micF₁</i>
0	0.457	0.474
2,000	0.602	0.602
4,000	0.633	0.633
6,000	0.637	0.637
8,000	0.645	0.645
12,000	0.649	0.649
16,000	0.653	0.653
20,000	0.669	0.668
25,000	0.674	0.677
37,500	0.650	0.653
50,000	0.644	0.649

The numbers in bold indicate the best results observed for this test.

For our next test, we used the D_L-3 dataset, which consisted of the most rare classes in the original dataset obtained from the ngLOC method (King and Guda, 2007). These data consisted of 584 sequences total, of which only 80 are classified as being targeted to LYS, representing only 13.7% of the data. The largest class in this dataset consists of those sequences targeted to GOL, which represents over 64% of the data. The remainder of the data are targeted to POX. This dataset exhibits numerous challenging characteristics common in real-world data classification problems: the amount of labeled data is extraordinarily small compared to other localization classes, and the data are highly unbalanced across all classes. A primary aim in this research was to improve predictive performance on rare classes that are also unbalanced, and these tests are designed to address these challenges.

We observed in our previous test how too much unlabeled data can have an adverse effect in situations where the amount of labeled data is a small fraction of the unlabeled data. We tested the EM- λ algorithm (Nigam, et al., 2000) with the entire D_L-3 dataset over different amounts of unlabeled data and different values of λ . We used an n -gram length of 4 for this test. Table 31 shows the results of this test. The baseline results for this test with no unlabeled data are $macF_1 = 0.561$ and $micF_1 = 0.728$.

Table 31 – The performance results for the D_L-3 dataset across differing values of the λ parameter in the EM- λ algorithm and differing amounts of unlabeled data.

λ	2,000 UL Sequences		4,000 UL Sequences		8,000 UL Sequences		25,000 UL Sequences		50,000 UL Sequences	
	$macF_1$	$micF_1$	$macF_1$	$micF_1$	$macF_1$	$micF_1$	$macF_1$	$micF_1$	$macF_1$	$micF_1$
0.01	0.581	0.735	0.592	0.738	0.598	0.741	0.635	0.745	0.677	0.752
0.025	0.598	0.741	0.592	0.736	0.608	0.736	0.674	0.738	0.657	0.700
0.05	0.591	0.736	0.606	0.735	0.647	0.740	0.659	0.700	0.623	0.654
0.075	0.602	0.735	0.633	0.743	0.672	0.747	0.627	0.661	0.613	0.640
0.1	0.606	0.735	0.631	0.733	0.669	0.729	0.627	0.658	0.610	0.637
0.25	0.653	0.741	0.674	0.729	0.637	0.673	0.618	0.647	0.622	0.652

0.5	0.673	0.733	0.649	0.692	0.625	0.656	0.622	0.656	0.633	0.668
0.75	0.663	0.719	0.643	0.683	0.629	0.659	0.626	0.663	0.643	0.682
1	0.656	0.711	0.643	0.682	0.637	0.673	0.631	0.670	0.656	0.697

The numbers in bold indicate the best test results for a given quantity of unlabeled data. (Baseline results with no unlabeled data are $macF_1 = 0.561$ and $micF_1 = 0.728$.)

The most important observation to make from this test is with regard to the effect that the λ parameter has over different amounts of unlabeled data. Through selection of appropriate values of λ , the unlabeled data can be controlled and prevented from dominating the parameter estimates. The best results were obtained when using 50,000 unlabeled sequences, and choosing $\lambda = 0.01$. This has an effect of reducing the ratio of unlabeled-to-labeled sequences to be roughly one unlabeled instance to every one labeled instance. While this may seem ideal, this should not be interpreted as a rule. (Refer to the discussion section for more details.)

We again notice that the $macF_1$ measure improved in every single test executed, indicating that the performance on the smallest classes in this dataset improved. Though there were numerous tests where $micF_1$ decreased below the 0.728 measure established without any unlabeled data, this did not happen on the best performing tests. (Regardless, we note that this is not necessarily a bad thing, because even though predicting every instance as GOL would yield a reasonable $micF_1$ measure of 0.647, this same classification would result with a $macF_1$ of .262, stressing the importance of observing $macF_1$ on unbalanced datasets.) We are more interested in improving the sensitivity on the smaller classes, ideally without a decrease in total accuracy, which was indeed accomplished. The best results yielded a more than 20% improvement in $macF_1$, increasing from 0.561 to 0.677, while showing a more than 3% improvement in $micF_1$, increasing from 0.728 to 0.752. This showed that using more unlabeled data was able to significantly improve predictive performance of most rare classes in the data, as long as

we were able to control the dominating effect that the unlabeled data had through the appropriate selection of λ .

Multi-EM Results

For our next test, we used the D_L-2B class dataset to evaluate the performance of the multi-component mixture model (Figure 15). We chose this dataset for this test, because we know that both the nucleus (NUC) and mitochondria (MIT) organelles in the cell are comprised of distinct suborganelles. As discussed, we would expect a multi-component mixture model to be able to model this better. For this test, we use n -gram values of both 4 and 5 for comparative purposes, and evaluate different numbers of components selected per class. We use a randomly selected set of set of 50,000 unlabeled data sequences Table 32 shows the results of this test.

Table 32 - Results for the Multi-EM algorithm.

# Labeled Seqs:		250		750		1250	
	<i>numComp</i>	<i>macF₁</i>	<i>micF₁</i>	<i>macF₁</i>	<i>micF₁</i>	<i>macF₁</i>	<i>micF₁</i>
4-gram		<i>0.4871</i>	<i>0.7406</i>	<i>0.5959</i>	<i>0.7940</i>	<i>0.5679</i>	<i>0.8065</i>
	1	0.6967	0.7669	0.7309	0.7966	0.7459	0.8195
	2	0.6972	0.7707	0.7417	0.8057	0.7463	0.8211
	3	0.7007	0.7744	0.7411	0.8070	0.7475	0.8236
	4	0.6972	0.7707	0.7402	0.8070	0.7462	0.8236
5-gram		<i>0.5386</i>	<i>0.7594</i>	<i>0.5639</i>	<i>0.7849</i>	<i>0.5326</i>	<i>0.8033</i>
	1	0.6625	0.7256	0.7011	0.7640	0.7250	0.8033
	2	0.6543	0.7368	0.7192	0.7979	0.7094	0.8154
	3	0.6568	0.7519	0.7210	0.8162	0.6945	0.8244
	4	0.6693	0.7744	0.6947	0.8083	0.6689	0.8211

This table presents the results from using varying amounts of multiple components per class in the mixture model (denoted *numComp*) on the D_L-2B dataset, using both a 4-gram and 5-gram feature space. We tested across different number of labeled sequences, using 50,000 unlabeled sequences in the test. The numbers in bold indicate the best performing results for a given number of labeled sequences. The results in italics represent the baseline results without unlabeled data.

All tests conducted yielded significantly better results than not using any unlabeled data, particularly with the $macF_1$ measure, indicating that the smallest classes in the data experienced remarkable improvement in sensitivity. The best performing tests were those that used more than one component per class in almost every case. The only exception was the $macF_1$ measure observed on the set of 1,250 labeled sequences using the 5-gram model, where the single component model surprisingly resulted in a better $macF_1$ measure (0.725) than the multi-component model. We ran further tests using larger amounts of labeled data, and we continued to observe a better $macF_1$ measure on the single component model. The 4-gram model did not have this problem on larger size labeled datasets tests. In all tests, the best $micF_1$ was always observed for the multi-component model.

These tests showed us that the multi-component model does indeed capture the underlying distribution of discriminating n -grams at the subcomponent level better than using a single component per class. This was much more evident with the 4-gram model, likely because the 4-gram model does a better job at being able to generalize beyond the observed training data. This is confirmed by comparing the performance measures without any unlabeled data between the 4-gram and 5-gram. Introducing multiple components per class into the mixture model introduces an extra risk of reducing generalization on new data not yet observed. For this reason, the smallest possible n -gram that can effectively classify the data should be used.

EM-CS Results

For our next set of tests, we evaluated the EM-CS algorithm, which iteratively constructs a subset of unlabeled sequences to be used for estimating model parameters by selecting

sequences at each iteration that meet or exceed a predetermined confidence score threshold. We chose the D_L-3 dataset for this test because it was designed to contain the rarest classes observed across all possible classes that could be observed, and thus it is quite reasonable to assume that the majority of the data in the unlabeled dataset are not associated with any of these three classes.

For this test, we chose a 4-gram model. We define *CStresh* to be the minimum allowed confidence score (CS) that a sequence must have to be considered in the model. For test purposes, we first generated predictions without using any unlabeled data on all the data in D_L-3 . We observed a CS range between 33.5 and 47.8 on all the data. Table 33 reports the percentile of the data falling over different selected CS thresholds in this test.

Table 33 – Percentile of data in D_L-3 falling in different ranges of CS based on cross validation.

CS	% correct	percentile
40	100%	94
38.25	97%	87
37	88%	71
36	85%	50
35	78%	21

The predictions were generated using a 4-gram model without using any unlabeled data. The range of the CS for all predictions falls between 33.5 and 47.8. We report the percentile to indicate the fraction of data in the D_L-3 data that is falling below the specified CS value. For example, 71% of all predictions in the D_L-3 dataset score less than 37.

Next, we evaluated the EM-CS algorithm at these different CS thresholds using a dataset of 60,000 unlabeled sequences. The performance results from the test are shown in Table 34. We report the number of sequences actually used in the model for each

value of $CStresh$ used. We also report the results without using any unlabeled data, and the results of using all the unlabeled data for comparison purposes.

Table 34 – Results for the EM-CS algorithm.

$CStresh$	<i>Unlabeled Sequences</i>	$macF_1$	$micF_1$
	0	0.561	0.728
0	60,000	0.650	0.690
35	6,814	0.662	0.721
36	3,810	0.674	0.735
37	2,388	0.676	0.741
38.25	1,474	0.672	0.752
40	726	0.621	0.740

The value of $CStresh$ was selected at different levels ranging between 35 and 40. The number of unlabeled sequences that were actually selected is reported. The numbers in bold indicate the best results. The numbers in italics indicate the baseline results for a model generated without unlabeled data.

As expected, a large number of sequences are able to be eliminated from the unlabeled dataset. Our lowest $CStresh$ value of 35 had eliminated almost 89% of the data in the unlabeled dataset. This confirmed that an overwhelming majority of this dataset is not likely to belong to these three classes. More importantly, we also observed an improvement in the $macF_1$ and $micF_1$ measurements by eliminating these sequences. The best results (indicated in bold in the table) occurred with a $CStresh$ value of 38.25, which resulted in 97.5% of the unlabeled data being eliminated. In this test, the $macF_1$ measure improved to 0.672 from 0.561, and $micF_1$ improved to 0.752 from 0.728. The scores improved over the results obtained from using the entire unlabeled dataset in all test cases, with a notable exception when $CStresh = 40$, demonstrating how performance can degrade if there are too few sequences selected due to a more restrictive setting of $CStresh$.

6.4.4 Comparison against Transductive SVM

For our final set of tests, we evaluated our method against the transductive SVM (TSVM) method discussed above (Joachims, 1999; Vapnik, 1995). We conducted a test on the D_L -2A dataset to ensure that the TSVM would yield reasonable results on a binary, balanced dataset. Our second test was on the D_L -3 dataset, which represents the most challenging problems for protein sequence classification.

The TSVM performed well on the basic D_L -2A dataset, but did not perform well on the T_L -3 dataset. The results for the D_L -2A data are shown in Table 35 and are presented first. The SVM method performed remarkably well, resulting in a $macF_1$ measure of 0.789. Adding unlabeled data for the TSVM method only marginally improved the results to almost 0.81 by including 12,000 unlabeled sequences. We noticed the predictive performance start to drop with any additional data beyond this. Moreover, we could not consider using more than 15,000 unlabeled sequences because the TSVM would not converge on a solution. The EM-CS algorithm showed a more remarkable improvement from no unlabeled data, increasing the $macF_1$ measure from 0.624 to 0.789 with 60,000 unlabeled sequences, resulting in a 26.4% improvement. However, we point out that the standard inductive SVM method performed the same as our method with unlabeled data, and adding unlabeled data marginally improved the results for the TSVM method. On the best results, the TSVM with 8,000 unlabeled sequences outperformed the EM-CS algorithm by 2%.

Table 35 – Comparing TSVM against the EM-CS algorithm on DL-2A

	TSVM		EM-CS	
<i># UL Sequences</i>	<i>macF₁</i>	<i>micF₁</i>	<i>macF₁</i>	<i>micF₁</i>
0	0.7894	0.7894	0.6244	0.6518

2,000	0.8004	0.8004	0.7479	0.7555
4,000	0.8022	0.8022	0.7614	0.7685
6,000	0.8059	0.8058	0.7538	0.7611
8,000	0.8095	0.8095	0.7552	0.7629
12,000	0.8095	0.8095	0.7681	0.7759
15,000	0.8059	0.8059	0.7723	0.7796
60,000	NA	NA	0.7889	0.7826

The DL-2A dataset represents the balanced, binary class dataset for testing. The results in bold are the best results from each method. NA = Not available.

There is a substantial cost in the computational time required for the TSVM compared to our method. See Table 36 for the results. We note that the computational time required for our method to complete is linear with respect to the size of the unlabeled dataset. SVM methods are known to have a running time between $O(n^2)$ and $O(n^3)$ with respect to the number of examples n in the labeled and unlabeled data, dependent on the value of the C and C^* parameters chosen (Bottou and Lin, 2007). Our empirical results shown in Table 36 clearly confirm these running times. This represents a significant benefit of our method over the TSVM. See the discussion section for more information pertaining to these results.

Table 36 – Running time comparison of TSVM against EM-CS.

<i># UL Sequences</i>	<i>TSVM (secs)</i>	<i>EM-CS (secs)</i>
0	6	19
2,000	555	468
4,000	2,016	1,147
6,000	4,768	1,609
8,000	9,908	3,090
12,000	14,653	5,069
15,000	122,813	9,326

This table shows the results of comparing the actual running time for a 10-fold cross validation on the DL-2A dataset over different amounts of unlabeled data between both methods. The results reported are in actual seconds of CPU time. All tests were conducted on a 3.0 Mhz Intel x64.

The results from our EM-based methods were clearly superior compared to the results from the TSVM on the DL-3 data. We report only the results from the EM- λ method. The results are displayed in Table 37. The best results from the TSVM method were obtained from using 6,000 unlabeled sequences, where the $macF_1$ measure increased from 0.5 without any unlabeled sequences to 0.66 by using unlabeled data, yielding a 32% increase in performance. Additionally, the $micF_1$ (overall accuracy) measure increased from 0.626 to 0.677, an 8.1% increase on the same test. As noted in the previous test, using more unlabeled data resulted in performance degradation. The TSVM failed to converge when using more than 10,000 unlabeled sequences. The best results from the EM- λ method were obtained from using 50,000 unlabeled sequences, where the $macF_1$ measure increased from 0.561 to 0.677, yielding a 21% performance improvement based on this measure. The $micF_1$ measure increased from 0.728 to 0.752, yielding a 3.3% increase in the measure. We note that our methods outperformed the TSVM in all cases, with the $macF_1$ measure showing a marginal 2.7% improvement over TSVM. We were pleased to see a significant 12.7% improvement in the overall accuracy from our method compared to the TSVM. The results of these two tests suggest that the TSVM performs well on simple binary, balanced classification problem; however the EM-based approach is more useful for multi-class problems where the data are unbalanced and availability of data are limited. See the discussion for a more detailed exposition on these methods.

Table 37 – Results of EM- λ against the transductive SVM on the DL-3 dataset.

# UL Sequences	TSVM		EM- λ	
	$macF_1$	$micF_1$	$macF_1$	$micF_1$
0	0.499	0.626	0.561	0.728
2,000	0.641	0.653	0.602	0.735
4,000	0.645	0.661	0.633	0.743

6,000	0.659	0.667	0.645	0.743
8,000	0.654	0.661	0.672	0.747
50,000	NA	NA	0.677	0.752

Results in bold indicate the best results observed for each method. NA = not available.

6.5 Discussion

Our previous work on the ngLOC method (King and Guda, 2007) has shown that when there is plenty of labeled data available, and there is a relatively high similarity threshold that is allowed in the training data, classification performance on new data can be quite impressive. However, in our previous work, as is the case with many other classifiers in this domain, prediction tends to suffer on the classes in the data with significantly low representation. Furthermore, there are significantly more biological concepts that proteins can be classified with that have significantly less data available than subcellular localization. Traditional supervised learning methods require a sufficient amount of labeled training data for inference of a model that has good generalization performance. In this study, we showed remarkable improvement in performance over supervised methods (i.e., naïve Bayes and SVM) when learning from small, multi-class, unbalanced datasets. Moreover, we gained the most substantial improvement in the least represented classes. These are challenges which are likely to be characteristic of a wide range of other protein annotations beyond subcellular localization. This suggests the value of this work for inducing models of protein characteristics that have very little labeled data available, perhaps allowing the inference of classification models on functional aspects of proteins that have yet to be considered.

6.5.1 Comparison Against Text Classification Methods

We applied the naïve Bayes classifier using an n -gram model for classification, where each possible n -gram is the equivalent of a word in a document used in text categorization. The largest distinguishing difference between our work and the feature space commonly found in text classification is the size of the feature space, denoted $|\mathcal{X}|$. Typical feature spaces in text categorization run on the order of 10^3 - 10^5 distinct features. In contrast, the feature space in protein sequence classification is exponential with respect to the value of n chosen, where the feature space is $O(20^n)$. While it is true that the observed feature space becomes sparse for values of $n > 4$, it is still enormous. Even at $n = 4$, there are 160,000 possible features, with almost all of them observed in the training data. At $n = 5$, though there are 3.2 million theoretically possible features, there were roughly 2.2 million features observed. Through careful, efficient memory management, and accounting for only those n -grams that occur in the training data, larger values of n are still very feasible to use. The primary restriction with using large values of n is the fact that it significantly reduces generalization beyond the training data. In general, if there is access to a large set of training data, and/or all data in the domain of interest have a very high level of sequence similarity, then large values of n are needed to represent discriminating n -grams that can effectively distinguish between the classes in the dataset. We use cross-validation to select the most ideal size of n for each dataset used.

The n -gram model represents a powerful, yet simple feature space for protein sequences. This model is able to capture sequence homology, while allowing for differences due to insertion, deletion, and/or mutation. It effectively shrinks the protein sequence space, thereby allowing a higher degree of redundancy between proteins that could not be had by considering the entire protein sequence. Moreover, different sized n -grams of significant length will map to structural features of the protein differently.

Secondary structure elements are vital for attaining a proper fold of a protein, and consequently are vital for its function. Hence, these secondary structures are distinctly conserved across proteins with different functions. Basic amino acid or pair-wise amino acid composition (i.e., a 1-gram or 2-gram model) represent feature spaces frequently used in this field, most likely due to the small dimensionality that is required by many classification methods. Though it has been applied with varying degrees of success, this limited feature space will fail to capture distinctive structural components of the protein. As demonstrated, a naïve Bayes classification method such as ours does not have this restriction. Fixed-length n -grams models were considered for this study, though numerous extensions to the basic n -gram model (e.g., single-gapped n -grams, variable-gapped n -grams) could be considered.

A significant difference with our work compared to other semi-supervised work in text classification is in regard to the domain in which the unlabeled data are taken from. In most semi-supervised work in text classification, it is assumed that the unlabeled data comes from the same source as the labeled data, and that the only difference is that it has not been categorized. Probabilistically speaking, it is assumed that D_U is drawn from the same distribution as D_L , formally stated as $\forall c_j \in \mathcal{C}, P(c_j | D_L) \approx P(c_j | D_U)$. However, many methods do not consider that D_U may be drawn from a different distribution. Even worse, D_U may contain documents belonging to classes entirely outside of the target classes. For example, if documents belonging to different sports from various athletic magazines are being categorized, documents pertaining to financial or business subject matters are not incorporated in D_U . This is a very reasonable assumption for most problems in text categorization; however, it is not reasonable for the majority of semi-supervised protein classification tasks. While

it is true that, biologically speaking, all proteins are synthesized through the same process of DNA replication, transcription, and translation, the mere fact that a protein has been successfully synthesized from a common source says absolutely nothing about its functional role in a species, or where in the cell it needs to localize to perform its function. It is for this reason that the EM-CS enhancement to the basic EM algorithm was added, to increase the likelihood of incorporating sequences that may at least have some association with one of the classes of interest. (The EM-CS algorithm is discussed below.)

6.5.2 Complexity of naïve Bayes and EM

The space requirements can be rather extensive for larger n -grams, and is upper-bounded by $O(|\mathcal{C}|20^n)$. The Multi-EM extension adds an additional multiplicative factor of the number of components selected per class. However, as discussed, for values of $n \geq 5$, the observed feature space becomes very sparse, with only a small fraction of all theoretically possible n -grams observed. Therefore it is essential that a one-to-one mapping of n -grams to distinct indices is maintained, in order to allow the exploration of large n -grams. However, the run time is linear with respect to the size of the labeled and unlabeled data – a very nice benefit of using the naïve Bayes classifier. If there are m_L sequences in D_L , and m_U sequences in D_U , and the average number of EM-iterations before convergence is p , and the average length of a sequence is k amino acids, then the running time for learning is bounded by $O(kp(m_L + m_U))$.

6.5.3 Considerations of EM Enhancements

In the EM- λ algorithm, the λ parameter serves as a weight applied to unlabeled data, in order to control the effect that unlabeled data has in determining parameter estimates. It explicitly violates the assumption that each sequence is generated by one component in the model, by assuming that a sequence can be generated by a fraction of a component. While this assumption does not seem sound, it is a very reasonable in situations where the amount of unlabeled data would render the labeled data ineffective in determining the final parameter estimates. We clearly showed how too much unlabeled data causes the predictive performance to deteriorate (Table 30), and we are quite certain that even more unlabeled data beyond what is shown would not improve the situation. As a final note regarding this algorithm, using EM- λ in conjunction with EM-CS will result in significantly less protein sequences from the unlabeled data selected, and therefore a larger λ parameter will likely be required. Our own observations confirmed this behavior (data not shown).

We point out that in several of our tests of the EM- λ algorithm, we often observed the best results when selecting a value for λ such that the unlabeled data was equally weighted with respect to the labeled data. However, this was not always observed, and should not be used as anything more than a good starting point, and then exploring alternative settings from there. Cross-validation can be used to select the most ideal value of λ for classification of new data.

In general, the problem of controlling the undesired influence of unlabeled data on parameter estimates can be addressed through either the EM- λ or EM-CS algorithms. The EM-CS will offer substantial running time improvement that EM- λ will not have. Moreover, EM-CS does not require relaxing the assumption that each component generates one sequence, unlike EM- λ does, because the unlabeled data are controlled by

removing poor examples of the data, instead of application of the weight parameter to reduce the effect of all unlabeled data. However, if it is determined that a large fraction of the unlabeled data contain instances belonging to the target classes, then EM- λ is the algorithm of choice, otherwise, EM-CS will likely yield superior results, provided an appropriate value of *CStresh* is chosen.

The EM-CS extension offers the most benefit when there is high certainty that the majority of the data in the unlabeled dataset are likely to belong to other classes not in $\mathcal{C}$. We successfully showed predictive performance improvements through the test on the D_L-3 dataset, which contained unbalanced data consisting of proteins localized to three distinct organelles that have a relatively small number of sequences targeted to them over all protein sequences available. Without the EM-CS extension, sequences that will never localize into these three organelles will be incorporated into the model, likely skewing parameter estimates because there are significant quantities of sequences not belonging to any target class.

In the EM-CS algorithm, we witnessed the importance of incorporating the sequences meeting the specified CS threshold in proportion to the current estimates for the class prior distribution (data not shown.) Specifically, this requirement was most useful in situations where data are highly unbalanced across classes. Without this restriction, we observed tests where the class prior distribution would quickly become more unbalanced than it was before beginning EM iterations, resulting with the prior probability becoming increasingly weighted toward the classes that were already the most populated.

We showed how the performance of EM-CS was dependent on the value of *CStresh* chosen. By observing the results in both Table 33 and Table 34, there is a clear

tradeoff between choosing an appropriate value for *CStresh* that maximizes selection of the most probable sequences that belong to one of the target classes, while allowing selection of enough unlabeled data to use in improving parameter estimates. A *CStresh* value that is too high will allow many potentially useful sequences to be prevented from being used in the EM-CS algorithm, while a *CStresh* value that is too low will incorporate sequences that are likely to belong to other classes outside of the set of target classes being modeled. The best selection of *CStresh* is chosen through cross validation based on observation of scores without unlabeled data first, and then choosing a few scores around the score falling at the 50 percentile mark.

A small improvement could be made to the EM-CS algorithm to improve the running time by limiting the number of iterations that the algorithm is allowed to add new sequences to the selected subset. We rarely observed any significant additional sequences added after the first few iterations of the EM algorithm. By eliminating the need to calculate the expected class assignments on the entire unlabeled data on future iterations of the E-step (and thereby eliminating those sequences from the M-step), the computational time required could be significantly improved without degradation in predictive performance.

In the multiple-component mixture model, there are varying techniques to consider selection the best number of components to model each class, including cross-validation, Akaike information criterion (AIC) and Bayesian information criterion (BIC) (Chapelle, et al., 2006; Mitchell, 1997). We used cross validation to evaluate which number of components were performing the best. However, the results we presented are based on a restricted implementation which uses a fixed number of components for every

class. While we believe that there may be benefit to selection of a specific number of components for each class, this was not tested.

6.5.4 Comparison against transductive SVM

A significant problem with SVM-based methods is the computational resources required in training the methods, particularly when considering non-linear kernels in the midst of training with high-dimensional, large-scale datasets. Our results showed how the computational time required to train transductive SVMs increased dramatically with respect to the amount of unlabeled data incorporated (see Table 36). Moreover, the method would often fail to converge if too much unlabeled data were used. Sometimes clustering techniques are applied and selective instances are used to represent each class, in order to reduce the size of the data used.

For high-dimensional data in particular, SVM methods become intractable for all but the smallest datasets. Solutions involve implementation of feature selection methods to reduce the feature space, clustering techniques to reduce the high-dimensionality of the data, or methods to reduce the size of the data. The computational resources required by our methods for training and testing are linear with respect to the dimensionality and amount of data used, which remains quite tractable for very large, high dimensional data.

Many existing SVM-based methods for protein sequence classification have used variants of standard amino acid composition and occurrence of amino acid pairs for feature representation (Garg, et al., 2005; Park and Kanehisa, 2003). These are equivalent to a 1-gram and a 2-gram model in our work. As noted previously, this feature space is very limited, particularly for work that requires discrimination across multiple classes, as it will not capture the discriminatory n -grams required to distinguish between classes.

Despite this, we tested both of these representations for all SVM and TSVM tests conducted. We tried to test 3-gram and higher feature spaces that were used for our methods, but the predictive performance for the SVM methods degraded substantially for the majority of the tests, or the computational time required for the method to converge was unreasonable.

A more significant problem involves the selection of the SVM and TSVM parameters. It is difficult to know whether the parameters you have selected are optimal. Moreover, tuning of the parameters usually involves a manual process of repeated tests until you achieve superior results, which can be an exhaustive task. This is a well known challenge with SVMs (Chapelle, et al., 2002), and represents an ongoing area of research in the machine learning community. Despite this challenge, when acceptable parameters are found, the results can be very good.

Quite often, the data are projected into a higher or different dimensional space through the kernel trick. It is possible that an ideal kernel may be found, and excellent SVM parameters might be determined for a classification problem, but impossible to understand how the translated feature space is discriminating between classes. With our methods, the feature space can be explored to extract features that were significant for classification, or anomalies in the data that were prohibitive to correct classification. These are significant requirements in the biomedical research community.

A probabilistic measure ought to be considered a crucial output of any predictive model, and this is a difficult feat due to their non-probabilistic output. Solutions to this problem for SVM methods usually involve creating another model to simulate a probability distribution from the output of the SVM (Yu, et al., 2006). Alternative techniques sometimes involve implementation of a bootstrap method that create multiple

SVMs using random draws from the training data, and estimating a probability distribution based on the output of these random SVMs. In contrast, probabilistic methods such as ngLOC (King and Guda, 2007) and the work presented in this study allow the output of a probabilistic measure. This is particularly important in the biomedical research, where it usually much more important to attain high precision over maximum overall sensitivity (i.e., recall) for a given classification method.

Though we noted an increase in performance for transductive SVMs compared to inductive SVMs for most of our tests, we noted a decrease in predictive performance for transductive SVMs compared to our semi-supervised method for most tests conducted except for the tests performed on the D_L-2A binary class, balanced dataset. Multiple kernels were tested for these tasks over a wide range of parameters, and the linear kernel always resulted in the best performance.

Despite these challenges with inductive and transductive SVMs, they have been shown to be successful on an extremely wide range of problems over many domains. We encourage the reader to consider TSVMs, particularly if an SVM approach has already been shown to work well. Our results indicated that if the inductive SVM performs well, then there is a high likelihood that incorporating a limited amount of unlabeled data with the TSVM may improve the model. Furthermore, methods are continuing to emerge that directly address many of these limitations mentioned. We were encouraged by the results of the basic D_L-2A data, and believe that TSVMs may be useful for protein sequence classification under certain scenarios. Moreover, observing the plethora of SVM-based methods for sequence classification will yield an equal number of choices for feature representation that researchers have adopted, and therefore, we believe that improved feature selection might have yielded improved performance for TSVMs.

6.6 Conclusion

Methods in protein classification continue to be researched in earnest, primarily focusing on improving classification of proteins from the sequence alone. Existing research on genomic and proteomic methods have yielded a large amount of protein data, of which only a fraction has been experimentally determined. This domain provides an excellent platform for research into semi-supervised learning methods, which have proven useful in domains where there is a large amount of data, and yet only a fraction of it has been labeled.

Many tasks in protein classification deal with challenges that arise due to limited data being available for learning, and unbalanced classes in the data. We presented a collection of EM-based algorithms designed to address these limitations and challenges through incorporation of unlabeled data. Using the n -gram model for the feature space, along with the naïve Bayes classifier, we showed how each of the algorithms can be used to improve classification accuracy over models inferred without unlabeled data. We focused extensively on improving prediction performance for the classes with the smallest representation in the training data, and showed how each algorithm improved the unbalanced predictions, sometimes remarkably well. The improvements in predictive performance were most substantial on the problems where the least amount of labeled data was available.

Limitations of this work were discussed, most of which resulted from the fact that our unlabeled dataset contained instances that belonged to classes that were not in the set of target classes being considered. It was also possible to have too much unlabeled data for adjusting the parameter estimates for the model in situations where there is a minute

amount of labeled data. However, by relaxing stated assumptions of the model, notably the one-to-one correspondence between the classes and underlying components in the mixture model, we demonstrated how these problems can be addressed through the EM- λ algorithm. We proposed a novel, alternative solution by using the EM-CS algorithm, which has the benefit of allowing the stated assumptions of the model to stay intact, while offering an additional improvement in run-time performance not possible with EM- λ .

We conducted a thorough comparison of our semi-supervised learning method against the transductive inference method of the transductive support vector machine (TSVM). We noted situations where the TSVM may or may not yield good classification performance compared to our method, with reasons for the observed performance discussed. Substantial improvement was shown by our method compared to the TSVM on the most difficult classification problems. However, we believe that the method is promising for protein sequence classification under certain limitations, and may result in a complementary classifier that can work in conjunction with our method, as well as other experimental methods in this field.

We have demonstrated the added value of semi-supervised learning in this domain, using specific tests designed to replicate common protein classification scenarios characterized by having only limited, highly unbalanced annotated data available. Our results have clearly demonstrated the value of incorporating a large set of unannotated protein sequences, showing how the predictive performance of the resulting classifier in the majority of our tests improved in many of these common scenarios.

7 Conclusions

This dissertation has focused on the use of supervised and semi-supervised learning for protein sequence classification. The primary methods have essentially been built upon the exorbitant work completed in text classification. The naïve Bayes classification framework was largely adopted in the supervised work. Numerous enhancements have been applied that consider the large feature space encountered in the n-gram model. Other enhancements consider the development of a species-specific predictive model. A semi-supervised learning approach was adopted that was based on the expectation maximization (EM) algorithm. We showed that both supervised and semi-supervised models can be successfully built, able to perform as well as or better than a large number of classifiers that use much more complex models, or require significant information beyond the sequence alone.

7.1 *Future directions*

Unfortunately, we still live in a world where people are still suffering from countless genetic disorders and diseases. We need to better understand relationships between patterns of interest in protein and DNA sequences and the diseases that have been thought to be related to these sequences. In other words, there is still an ever-increasing need to understand the biological functions of genes and proteins in all species. Therefore, proteomic and genomic methods will continue to be developed. Yet, the majority of data analysis methods today are failing to keep up with the exponential rate at which data continues to be made available.

There are many challenges in large-scale, sequence-based classification in both protein and DNA classification methods that are yet to be solved. Experimental methods

continue to abound and generate data, leaving computational biologists left with many ongoing challenges that need to be addressed. The boundaries of existing computational resources continue to be pushed, and we need to tap these resources effectively.

This section will, in very general terms, outline areas for future research. Most of these suggestions are considered based on the extensive work conducted for this dissertation, as well as countless discussions had with colleagues through e-mails, at conferences, and on campus.

7.1.1 Predictive vs. explanatory reasoning

Too many methods today focus on predictive reasoning. Better methods need to be developed for explanatory reasoning. Researchers need to extract knowledge (or information) associated with the underlying structure and distribution of data, in order to better understand the biological process(es) that generated the data in the first place. A lot of methods fail to provide useful information to non-computer scientists for which the tool was intended. This is particularly important for research in bioinformatics, because many of the users are researchers in the biomedical community, and this must be considered.

7.1.2 Subsequences of biological or biomedical interest

Of particular interest is the determination of discriminatory features in the sequence data. In particular, it would be beneficial to answer the question, which subsequences in the data were most useful for classification? (An obvious computational side-benefit of this is the application of feature selection to reduce high-dimensional data.) On the other end, it would also be useful to know what subsequences might be

anomalies, as compared with the general population. This information could suggest possible mutations of interest.

7.1.3 Semi-supervised learning

The work on semi-supervised learning conducted for this dissertation represents only a beginning of the possible directions that can be considered in the future. The work completed using the transductive SVMs as well as the EM-based Bayesian model opened many doors for further exploration.

The work completed on the TSVM was intriguing. It was surprising how well it performed on the smaller n-gram models. It would be interesting to consider refinements that would make the TSVM tractable over a larger, sparser feature space. Additionally, as mentioned in Section 7.1.1 and 7.1.2, there is a need to extract information from the model – a well-known problem with the SVM classification model. Clearly, the SVM inherently discriminates between objects belonging to different classes. However, it is difficult to determine which features are most significant in discriminating between different classes, owed in part to the kernel trick. It is not known if any work has been completed on SVMs or TSVMs that have the ability to extract useful information from their respective models over a biological sequence domain.

It would also be of interest to consider some other semi-supervised methods that have been used successfully in recent years in other domains, and see how they could be applied in this work. One method of interest is the co-training method of Blum and Mitchell (Blum and Mitchell, 1998). The co-training method is a semi-supervised learning method that can be used when two (or more) disjoint feature sets can be used to represent the data. Each feature set correspond to different "views" of the data, and it is

assumed that each view alone provides a sufficient feature set for classification. The main requirement is that each set of features must be conditionally independent, given the class. The success of co-training largely depends on how exhaustive the two independent feature spaces are. Each feature space must minimally be sufficient for weak classification. This requirement would represent a large part of the success of this work.

7.1.4 Ensemble classification methods

Machine learning and data mining have given us an extraordinary number of classification methods. Each of these methods has certain strengths and weaknesses, as well as limitations with regard to their capability to learn the true underlying model of the data. How does one weigh each model in the ensemble? Even one type of classifier can yield an infinite array of models to consider, depending on the training data used, the parameter space, the prior assumptions, or other issues. Is it possible for an ensemble to truly do better than any one model? In this work, it is possible to consider an ensemble over different n -gram spaces, using the same underlying predictive model.

7.1.5 Sample selection bias

Existing protein databases do not represent the entire protein universe, nor is the subset of all possible proteins available in these databases sampled according to the true distribution representing all proteins. The proteins that have the most detailed functional annotation have been researched in depth for a reason. First, it is quite possible that the proteins were easy to determine, due to being readily accessible for *in vivo* or *in vitro* experimentation. However, it is more often the case that many of the proteins deposited

in public databases have resulted from research on model organisms that have been selected for their relative merit in biomedical research or in basic sciences.

To illustrate, an early analysis was done using the Swiss-Prot database, release 51.2. It has 243,975 protein sequences. As discussed, sequences in Swiss-Prot have a high-level of annotation that has been experimentally determined to some degree. There are 10,723 different species represented in this database. However, this database does not contain a representative sample drawn from the true distribution of the protein universe. The top five species, representing 0.047% of the species in Swiss-Prot, occupy 17.8% of the entire data (see Table 38). Among these top five species, only yeast has its entire proteome included, whereas human, the most represented species in the database, has only approximately 43% of its proteome included (assuming an estimate of ~35,000 proteins in the human proteome.)

Table 38 - Most represented species in Swiss-Prot

Number of sequences	Species
15093	<i>Homo sapiens</i> (Human)
12000	<i>Mus musculus</i> (Mouse)
5956	<i>S.cerevisiae</i> (Yeast)
5577	<i>Rattus norvegicus</i> (Rat)
4892	<i>E.coli</i>
43518 (17.8% of Swiss-Prot)	

Many protein classification methods available today work well because they are tuned according to the skewed distribution of data available in public databases. These methods often are advertised as predictors for any protein from any species. The reality is that, though many methods will have reasonable accuracy simply due to the homologous nature of many sequences throughout the eukaryotic kingdom, many methods are likely to fail for predictions on new and dissimilar sequences. The ngLOC

method has taken a preliminary step to address this problem by reporting a confidence score, which implicitly acknowledges the lack of closely related data to give a good prediction.

To illustrate the potential problem, a set of sequences from two species were retrieved. One dataset contained 2,292 sequences from a species in the *Plasmodium* genus (malaria parasite), and another dataset contained 24,638 sequences from the human proteome. Using the ngLOC method, only 3.6% of the predictions from plasmodium (84 sequences) were output with a high confidence score (e.g. CS $\geq$ 70), whereas 39.1% (9,625) of the human predictions had the same high confidence measure. Part of these results are explained by the high representation of higher-order species in the ngLOC dataset (see Table 39). Because the ngLOC dataset was retrieved from the Swiss-Prot database, it was drawn according to a similar distribution. Therefore, as one would expect, human is the highest represented species in this dataset. In fact, from the 24,638 human sequences tested, 5,945 of them were identical to a sequence in the ngLOC dataset. This certainly accounts for over half of the high predictions. The rest of the high predictions are due to the presence of close homologs in the dataset. However, there were no occurrences of Plasmodium proteins in the ngLOC dataset. Furthermore, judging from the low fraction of high-confidence predictions, one can infer that the ngLOC dataset has very few close homologs to Plasmodium. The ngLOC data was drawn from the Swiss-Prot data repository, which has a clear sample selection bias toward proteins from higher-order eukaryotic species, and will thus perform very well for these species.

Table 39 - Most represented species in ngLOC dataset

Number of sequences	Species
5271	<i>Homo sapiens</i> (Human)
3288	<i>Mus musculus</i> (Mouse)

2108	<i>Rattus norvegicus</i> (Rat)
1724	<i>S.cerevisiae</i> (Yeast)
838	<i>B.taurus</i> (Cow)
13229 (51.1% of ngLOC data)	

This analysis shows the difficulties of assuming that one method has global applicability for prediction over all possible sequences over all species. The reality is that most methods have local predictive abilities over the protein space. In other words, as long as the data that is tested on a predictive method has the same sample bias as the training data, then this does not represent a problem. This is precisely why any validation technique (including the standard n -fold cross validation) that uses data drawn from the same skewed distribution as the training data can not be used for detection of a sample selection bias. An independent test set that hopefully represents data that is drawn more in accordance with the true, underlying distribution is the most straightforward means of detecting sample selection bias in a classifier, or at least detecting when the test data is created under a different sample bias than the training data.

New modern genomic and high-throughput proteomic technology for sequencing and experimentally determining proteins is allowing new data to become available, likely representing portions of the protein universe that have yet to be observed. It is likely that existing methods will not perform well due to being tuned with a skewed data sample. The exponential increase of available data implies that it is only a matter of time before portions of the protein universe that have remained untouched will be visited. A method for detecting, and even correcting for, sample selection bias in biological data such as protein data is appearing to become more important.

7.1.6 Learning with multiple class labels

The research on the ngLOC method showed that prediction of multiple class labels was possible, as the method had reasonable accuracy for predicting when a sequence was multi-localized into up to two organelles, as well as predicting what those organelles were. The prediction was limited to predicting two localizations. However, it is possible that a single sequence could have 3 or more localizations. The problem of protein subcellular localization prediction makes a good platform for the development of a framework for probabilistic classification in situations where each object being classified can have multiple labels.

7.1.7 Prediction of suborganellar locations

Many organelles in the cell have subcompartments. For example, the mitochondrion has an outer membrane enclosing its structure, and an inner membrane separate the matrix from the inter membrane space and inner matrix. Each one of these substructures of the organelle harbor hundreds of proteins to carry out specific functions in those locations. Accurate prediction at this level of granularity would have significant benefits to biomedical research, including drug targeting, as it would greatly increase the understanding of the subcellular proteome on a new level.

Part of accomplishing this will require a new dataset containing annotation at this level of granularity. A hierarchical classification model could be considered in this work, where the first layer predicts at the major organelle level. The output of this first classifier is input to a second layer that predicts at the suborganelle level.

7.1.8 Natural amino acid groupings

The ngLOC method uses all possible n -grams as the feature space for representing proteins into the classifier. It was observed that in feature spaces over longer n -grams, there are many n -grams that do not exist. Of those that do exist, a large fraction of them only occur once in the entire dataset, and thus are useless for cross-validation testing. A longer n -gram feature space lacks ideal characteristics of allowing generalization for new n -grams. It would be desirable to have a method that can reduce the feature space for longer subsequences without losing information contained in the subsequence.

Recall that a folded protein is made of several secondary structures. At the secondary structure level, there are many primary structures that map to the same secondary structures. This suggests that there are certain amino acids that must share highly similar physicochemical or structural properties that would allow such mutations in the sequence to form identical secondary structures. Indeed, it is possible to use these known properties to form natural groupings of amino acids, thereby reducing the number of possible labels at each position. The amino acid sequence can then be converted into a sequence of physicochemical or structural properties.

There are many possible properties that can be considered, and many natural groups that have been used in related studies. Table 40 presents a list of the most common amino acid properties considered, as well as the most common groupings of those properties (Acquaah-Mensah, et al., 2006; Dubchak, et al., 1999; Yu, et al., 2006). The idea of using these groupings is based on the work of Nakai et al., which performed a cluster analysis over different secondary structures in relation to various physicochemical properties (Nakai, et al., 1988). They analyzed the amino acid propensities in these clusters to derive the groupings.

Table 40 - Amino acid groups

Group	Label	Amino Acids
Hydrophobicity	Polar Neutral Hydrophobic	RKEDQN GASTPHY CVLIMFW
Normalized van der Waals volume	Small (0 – 2.78) Medium (2.95 – 4.0) Large (4.43 – 8.08)	GASCTPD NVEQIL MHKFRYW
Polarity	Low (4.9 – 6.2) Neutral (8.0 – 9.2) High (10.4 – 13.0)	LIFWCMVY PATGS HQRKNED
Polarizability	Low (0 – 0.108) Medium (0.128 – 0.186) High (0.219 – 0.409)	GASDT CPNVEQIL KMHFRYW
Charge	Positive Negative Other	HRK DE MFYWCPNVQILN

Future work will look at using these natural groupings to form alternative feature sets for classification. It is expected that converting the amino acid sequence into a sequence of labels over particular groupings will allow longer subsequences over the primary structure to be used for classification. (Additionally, it is likely that these groupings could form the basis for different "views", as required in the co-training method presented in section 7.1.3.)

7.1.9 Substitution matrices

Substitution matrices, such as the common BLOSUM62 substitution matrix, are used for scoring an alignment between two aligned sequences. The matrix has 20 rows and 20 columns, representing each of the 20 amino acids. Each entry in the matrix has a score whose value is dependent on how common a possible mutation or match is. Specifically, the entries in the matrix reward substitutions that are common, and penalize

those that are rare. The aim is to give the highest score to the sequence that is most closely related (evolutionarily speaking) to the original sequence (Mount, 2004).

The substitution matrix would be used in the following way. During the learning phase, the model accumulates a count of all n -grams that occurred in the training data. However, it is highly likely that n -grams will be observed in test sequences that were not observed in the training data. Currently, the LaPlace correction allows such an occurrence to generate a usable probability measure from the model by assuming that any possible n -gram in the entire universe of n -grams occurs at least once – an inductive bias that might cause skewed results. Instead, by using a substitution matrix, we can scan the observed n -grams for a close match to the unobserved n -gram in the test sequence.

There are two major hurdles that need to be crossed in order for such a method to be tractable. The first one involves searching for the best n -gram candidate to use as a representative for the unobserved n -gram. Clearly, an exhaustive search is way too costly, as it would take $O(n \log |\mathcal{X}|)$ time to search, and there are $O(20^n)$ theoretical possibilities to try! There are numerous possible methods to prune the search space, such as limiting the search for candidates with only one mutation. More importantly, an advanced data structure for all of the observed n -grams will need to be used, such as a suffix tree. The second challenge involves how to use the n -gram representative to estimate the probability of the unobserved n -gram. Clearly, the resulting substitution score will need to be considered as some type of weight on the n -gram representative, as it can not be considered as an equal to the closely-related unobserved n -gram. It is currently unclear how the substitution score should be used.

In closing, there are numerous paths that one can consider when attempting to build subcellular localization models. A researcher only needs to search on any of the well known research repositories for methods in protein subcellular localization prediction, and he/she will find that it is still among the most researched areas in bioinformatics. While it is encouraging to see the plethora of research in this field, there is a severe lack of quality control in the methods used to report performance results. It would be extremely advantageous to the research community to invest time in development of rigorous gold standard performance measurement methods that all methods should report. Equally important would be a gold standard test dataset that is periodically updated to incorporate a wide range of proteins for localization testing. This dataset should include well-understood proteins across the most common species, as well as new proteins that are difficult to predict (i.e. low homology). Surely such a task is easy to suggest, but difficult to implement. However, the research community would benefit significantly from such efforts.

Appendix

A.1 Mathematical notation

A list of all mathematical and symbolic notation used is indicated below

Table 41 - Mathematical notation

d_i	The i^{th} protein sequence in the dataset
s_j	The j^{th} amino acid in the protein sequence
y_i	The j^{th} category that a protein may be assigned to
$\mathcal{S}$	True, unknown distribution from which data is drawn from
$\mathcal{X}$	Feature space used for representing objects being classified
$\mathbf{X}$	Random variable representing the attributes of an object being classified
$\mathcal{C}$	Set of categories that each data instance may be assigned to
Y	Random variable representing the class of categories (or labels) assigned to objects
$\mathbf{Z}$	Hidden or latent variable representing unobserved data
$\mathbf{x}_i$	The i^{th} instance, represented by features
x_{ij}	The j^{th} feature of object i
x_j	The j^{th} feature
y_j	The j^{th} category
$\mathcal{D}$	The space of all possible proteins
D	Set of training data used to train the classifier
D_j	Set of training data that belongs to category j
Θ	The space of parameters for the class-conditional densities over all classes
θ	Vector of parameters for all class-conditional densities
θ_j	Vector of parameters for the class-conditional density for class j
θ_{jt}	Parameter for feature t for the class-conditional density for class j
$\hat{\theta}$	Estimates of parameters for class-conditional densities over all classes
Φ	Parameter for the class prior probability
$\hat{\phi}$	Estimator for the class prior probability
$h:\mathbf{X} \rightarrow Y$	The classification model, mapping objects to their classification

A.2 Table of abbreviations

The following is a list of the abbreviations used in this paper

Table 42 - Table of abbreviations

ANN	Artificial Neural Network
SVM	Support Vector Machine
EM	Expectation Maximization
GEM	Generalized Expectation Maximization
IID	Identically and independently distributed
pdf	Probability density function

A.3 Mathematical concepts

A.3.1 Maximum Likelihood Estimation (MLE)

Maximum likelihood estimation (MLE) is a statistical method used for finding the parameters of a probabilistic model when the model parameters are not observed. With a set of data given beforehand, and a probability distribution function assumed, MLE can be used to estimate the parameters of the distribution that maximizes the likelihood of the data, given the model (DeGroot and Schervish, 2002; Duda, et al., 2001). The basis of this method is the *likelihood function*, which is a probability function where the parameters of the model form the formal parameters of the pdf, and the data is fixed.

Formally, let f represent a probability density function (pdf, or pmf in the discrete case) of some vector $\mathbf{X}$ of IID random variables $X_1, \dots, X_n$, representing n samples of data drawn from the distribution. Let $\boldsymbol{\theta}$ represent the vector of parameters of the pdf. Then, the likelihood function is given by:

$$\begin{aligned} L(\boldsymbol{\theta}; \mathbf{x}) &= P(\mathbf{X} = \mathbf{x} | \boldsymbol{\theta}) \\ &= \prod_{i=1}^n P(X_i = x_i | \boldsymbol{\theta}) \\ &= \prod_{i=1}^n f(x_i | \boldsymbol{\theta}) \end{aligned}$$

The log-likelihood is more often used, as it is convenient for analytical reasons. We define the log-likelihood function as:

$$l(\boldsymbol{\theta}; \mathbf{x}) = \log L(\boldsymbol{\theta}; \mathbf{x}) = \sum_{i=1}^n \log f(x_i | \boldsymbol{\theta})$$

The maximum likelihood estimator, denoted $\hat{\theta}$, is the value of the model parameters in which the probability of the observed data $\mathbf{X}$ is maximized. There are several methods for estimating the maximum likelihood, most of which use differentiation of the likelihood with respect to the parameters to find the maxima of the likelihood function. As long as the pdf is of a form that can be differentiated, then a gradient can be determined, thereby allowing classical gradient ascent approaches to be used to find the estimates.

Some methods consider determining an estimate of $P(\hat{\theta})$, the probability that $\hat{\theta}$ is the correct estimate of the model. Methods that find the best estimate of θ that maximizes the equation $l(\theta, \mathbf{x})P(\theta)$, are known as maximum a posteriori (MAP) estimators. This probability can form an estimate for the class prior $P(Y)$. As an estimator for $P(Y)$, it requires that the dataset, from which the estimate is being determined, does not have a sample selection bias.

A.3.2 Expectation Maximization (EM)

In many statistical inference problems, it is unusual to have a complete set of data that is fully observed. There may be missing features in the data, or there may be instances of data missing altogether. The expectation maximization (EM) algorithm provides a means of being able determine estimates for model parameters without having a complete set of data (Bishop, 2006; Duda, et al., 2001; Hastie, et al., 2001).

Models that consider missing data start by introducing one or more *hidden*, or *latent* variables to represent the unobserved data. Let $\mathbf{X}$ represent the observed data, let $\mathbf{Z}$ represent the latent data, and let $p(\mathbf{x}, \mathbf{z} | \theta)$ represent the joint pdf over both observed and unobserved data. We desire to maximize the probability of the *complete joint log-likelihood* of both the observed and hidden data, denoted as:

$$\begin{aligned} l_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{z}) &= \log P(\mathbf{X} = \mathbf{x}, \mathbf{Z} = \mathbf{z} | \boldsymbol{\theta}) \\ &= \log p(\mathbf{x}, \mathbf{z} | \boldsymbol{\theta}) \end{aligned}$$

However, because of the unobserved data, the complete joint log-likelihood can no longer be directly maximized. Instead, the aim of EM is to maximize the *expected value* of the complete log-likelihood, given the observed data and current estimates for the parameters. To do so, we treat the complete log-likelihood function itself as a random variable, which is possible due to the unobserved data $\mathbf{Z}$. We fix the model parameters to some estimate, denoted $\hat{\boldsymbol{\theta}}_i$. A new function, called Q , is evaluated that represents the expected complete log-likelihood over the unobserved data, given fixed current parameter estimates $\hat{\boldsymbol{\theta}}_i$ and observed data $\mathbf{x}$:

$$Q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}_i, \mathbf{x}) = E(l_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{Z}) | \hat{\boldsymbol{\theta}}_i, \mathbf{x})$$

Assuming that the unobserved data is generated according to a conditional distribution $p(\mathbf{z} | \mathbf{x}, \boldsymbol{\theta})$, we can expand the Q function as:

$$\begin{aligned} Q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}_i, \mathbf{x}) &= E(l_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{Z}) | \hat{\boldsymbol{\theta}}_i, \mathbf{x}) \\ &= \int l_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{Z}) p(\mathbf{z} | \hat{\boldsymbol{\theta}}_i, \mathbf{x}) d\mathbf{z} \end{aligned} \tag{3.1}$$

Determining the Q function makes up the expectation step of the EM algorithm. Once this function has been determined, the maximization step involves finding a new estimate of $\boldsymbol{\theta}$ to maximize the Q function.

$$\hat{\boldsymbol{\theta}}_{i+1} = \arg \max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}_i, \mathbf{x}) \tag{3.2}$$

The EM algorithm iterates between these two steps, continually improving on its estimate of the parameters until a specified convergence criterion is met. The EM algorithm does not guarantee convergence to a maximum likelihood estimator, and could (and often does) converge to a local maxima of the likelihood function. Despite these

limitations, it has been widely used in a variety of statistical inference and machine learning algorithms.

A.3.3 Generalized EM (GEM)

The EM algorithm, as presented, is guaranteed to converge to a locally optimal value under the stated assumptions. However, there are cases where the EM algorithm as presented may be difficult due to the difficulty in maximizing the likelihood, as specified in Eq. (3.2). The generalized EM (GEM) algorithm relaxes the requirement of doing a full maximization in this step. It can be shown that one only needs to find a value of θ_{i+1} that *improves* the estimate of the Q function (Hastie, et al., 2001). In other words, find θ_{i+1} such that:

$$Q(\hat{\theta}_{j+1}; \hat{\theta}_j, \mathbf{x}) > Q(\hat{\theta}_j; \hat{\theta}_j, \mathbf{x})$$

The k -means clustering algorithm is an example of a generalized EM algorithm.

A.4 Subcellular proteome estimates for eight eukaryotic species

This section contains both the percentage and numeric subcellular proteome estimates for eight eukaryotic species, as reported in King et al. 2007 (King and Guda, 2007).

A.4.1 HUMAN (*H.sapiens*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	14.14									
CSK	0.64	1.48								
END	0.10	0.01	3.04							
EXC	0.22	0.01	0.04	11.71						
GOL	0.29	0.03	0.31	0.17	1.56					
LYS	0.02			0.03	< 0.01	0.67				
MIT	0.31		0.07	0.02	< 0.01		4.80			
NUC	4.74	0.07	0.09	0.12	0.01		0.09	28.38		
PLA	0.77	0.02	0.14	0.94	0.09	0.00	0.03	0.19	24.08	
POX	0.05			< 0.01			0.03			0.46

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	5097									
CSK	231	533								
END	36	3	1095							
EXC	79	5	15	4224						
GOL	105	9	113	60	564					
LYS	8			12	2	243				
MIT	113		25	8	2		1730			
NUC	1708	25	31	42	5		31	10235		
PLA	279	8	50	341	34	2	12	70	8682	
POX	17			2			11			167

A.4.2 MOUSE (*M. Musculus*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	13.44									
CSK	0.64	1.50								
END	0.08		2.99							
EXC	0.19	< 0.01	0.04	11.52						
GOL	0.27	0.04	0.32	0.12	1.52					
LYS	0.03		0.01	0.04	0.01	0.59				
MIT	0.31		0.07	0.05	0.01		4.24			
NUC	4.51	0.08	0.09	0.16	0.02		0.11	27.35		
PLA	0.65	0.03	0.08	0.76	0.10	0.01	0.05	0.26	27.18	
POX	0.08						0.01	< 0.01	0.01	0.44

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	4210									
CSK	200	469								
END	26		937							
EXC	58	1	13	3609						
GOL	85	11	99	38	477					
LYS	8		4	11	3	184				
MIT	96		21	15	3		1329			
NUC	1411	25	29	51	6		35	8566		
PLA	203	10	25	238	31	4	17	82	8511	
POX	25						3	1	3	138

A.4.3 CHICKEN (*G.gallus*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	13.66									
CSK	1.01	1.24								
END	0.21		2.53							
EXC	0.23		0.07	12.65						
GOL	0.32	0.02	0.48	0.25	1.47					
LYS	0.02			0.05		0.44				
MIT	0.28		0.07	0.02	0.05		4.16			
NUC	6.27	0.07	0.02	0.11			0.14	28.24		
PLA	0.90	0.02	0.11	1.17	0.14		0.02	0.39	22.78	
POX	0.09						0.02		0.02	0.25

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	736									
CSK	54	67								
END	11		136							
EXC	12		4	681						
GOL	17	1	26	14	79					
LYS	1			2		23				
MIT	15		4	1	2		224			
NUC	338	4	1	6	0		7	1521		
PLA	48	1	6	63	7		1	21	1226	
POX	5						1		1	14

A.4.4 ZEBRAFISH (*D. rerio*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	15.01									
CSK	0.81	1.31								
END	0.15		3.34							
EXC	0.33	0.01	0.03	9.91						
GOL	0.17	0.02	0.40	0.17	1.68					
LYS	0.02			0.07	0.01	0.65				
MIT	0.39		0.09	0.07	0.03		4.72			
NUC	5.40	0.03	0.04	0.25	0.01		0.14	30.31		
PLA	0.87	0.02	0.04	0.77	0.11		0.09	0.27	21.66	
POX	0.06						0.01			0.51

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	2043									
CSK	111	179								
END	20		455							
EXC	45	1	4	1349						
GOL	23	3	55	23	228					
LYS	3			10	1	89				
MIT	53		12	10	4		643			
NUC	735	4	5	34	1		19	4128		
PLA	119	3	5	105	15		12	37	2949	
POX	8						1			70

A.4.5 MOSQUITO (*A.gambiae*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	14.43									
CSK	0.25	1.11								
END	0.10		3.25							
EXC	0.32		0.06	14.28						
GOL	0.06	0.01	0.21	0.03	1.07					
LYS	0.01			0.02	0.02	0.53				
MIT	0.40		0.02	0.10	0.01		5.52			
NUC	3.44	0.05	0.02	0.66	0.05		0.08	29.50		
PLA	0.38	0.05	0.05	1.34	0.07		0.13	0.48	21.36	
POX	0.02						0.01			0.48

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	2140									
CSK	38	164								
END	15		483							
EXC	48		9	2118						
GOL	9	2	31	5	159					
LYS	2			3	3	79				
MIT	60		3	15	2		818			
NUC	510	7	3	98	7		12	4376		
PLA	56	7	7	199	10		19	72	3169	
POX	3						2			72

A.4.6 FRUITFLY (*D.melanogaster*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	12.74									
CSK	0.24	1.05								
END	0.06		2.85							
EXC	0.22	0.01	0.02	12.26						
GOL	0.07	0.03	0.12	0.04	1.40					
LYS	0.01			0.07	0.01	0.55				
MIT	0.32		0.02	0.12			4.86			
NUC	2.76	0.03	0.01	0.37	0.02		0.09	37.38		
PLA	0.28	0.01	0.02	0.88	0.05		0.07	0.42	20.06	
POX							0.01			0.42

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	1682									
CSK	35	139								
END	11		377							
EXC	30	1	7	1619						
GOL	7	1	23	7	184					
LYS	1			11	3	72				
MIT	49		1	11			642			
NUC	405	7	3	67	4		11	4936		
PLA	38	3	3	132	7		10	51	2649	
POX	1						1			56

A.4.7 WORM (*C.elegans*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	14.80									
CSK	0.20	1.19								
END	0.08		3.47							
EXC	0.23	0.02	0.07	12.60						
GOL	0.09	0.02	0.11	0.05	1.31					
LYS				0.03	0.01	0.58				
MIT	0.25		0.02	0.08			5.84			
NUC	2.36	0.04		0.39	0.01		0.06	29.75		
PLA	0.20	0.03	0.04	0.60	0.08		0.05	0.24	24.41	
POX	0.03						0.01			0.66

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	3147									
CSK	44	252								
END	16		738							
EXC	49	4	15	2679						
GOL	20	4	24	11	279					
LYS				5	2	123				
MIT	53		4	16			1242			
NUC	502	9		83	2		13	6325		
PLA	44	7	9	127	16		11	51	5190	
POX	7						2			140

A.4.8 YEAST (*S.cerevisiae*)

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	15.22									
CSK	0.28	1.07								
END	0.06		2.71							
EXC	0.09			8.88						
GOL	0.11		0.43		1.48					
LYS				0.02		0.11				
MIT	0.77	0.02	0.09	0.11			9.55			
NUC	6.49	0.17	0.19	0.21	0.02		0.30	33.53		
PLA	0.13		0.04	0.50	0.04		0.09	0.47	16.19	
POX	0.06						0.04			0.54

	CYT	CSK	END	EXC	GOL	LYS	MIT	NUC	PLA	POX
CYT	860									
CSK	16	60								
END	3		153							
EXC	5			502						
GOL	6		24		84					
LYS				1		6				
MIT	43	1	5	6			540			
NUC	367	10	11	12	1		17	1895		
PLA	7		2	29	2		5	26	915	
POX	3						2			31

References

- Acquaah-Mensah, G.K., Leach, S.M. and Guda, C. (2006) Predicting the subcellular localization of human proteins using machine learning and exploratory data analysis, *Genomics, proteomics & bioinformatics / Beijing Genomics Institute*, **4**, 120-133.
- Alpaydin, E. (2004) *Introduction to machine learning*. MIT Press, Cambridge, Mass.
- Altschul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman, D.J. (1990) Basic local alignment search tool, *Journal of molecular biology*, **215**, 403-410.
- Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs, *Nucleic acids research*, **25**, 3389-3402.
- Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M. and Sherlock, G. (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium, *Nature genetics*, **25**, 25-29.
- Bejerano, G. and Yona, G. (2001) Variations on probabilistic suffix trees: statistical modeling and prediction of protein families, *Bioinformatics (Oxford, England)*, **17**, 23-43.
- Bhasin, M., Garg, A. and Raghava, G.P. (2005) PSLpred: prediction of subcellular localization of bacterial proteins, *Bioinformatics (Oxford, England)*, **21**, 2522-2524.
- Bishop, C.M. (2006) *Pattern recognition and machine learning*. Springer, New York.
- Blum, A. and Mitchell, T. (1998) Combining labeled and unlabeled data with co-training. *COLT: Proceedings of the eleventh annual conference on Computational Learning Theory*. Morgan Kaufmann Publishers, 92-100.
- Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.C., Estreicher, A., Gasteiger, E., Martin, M.J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S. and Schneider, M. (2003) The SWISS-PROT protein knowledgebase and its supplement TrEMBL, *Nucleic acids research*, **31**, 365-370.
- Bottou, L. and Lin, C.J. (2007) Support Vector Machine Solvers. In Bottou, L., Chapelle, O., DeCoste, D. and Weston, J. (eds), *Large-Scale Kernel Machines*. The MIT Press, Cambridge, MA, 1-27.
- Brinkman, F.S. (2007) Personal Communication.
- Chang, J.-M., Su, E.C.-Y., Lo, A., Chiu, H.-S., Sung, T.-Y. and Hsu, W.-L. (2008) PSLDoc: Protein subcellular localization prediction based on gapped-dipeptides and

probabilistic latent semantic analysis, *Proteins: Structure, Function, and Bioinformatics*, **NA**, NA.

Chapelle, O., Scholkopf, B. and Zien, A. (eds) (2006) *Semi-Supervised Learning*. The MIT Press, Cambridge, MA.

Chapelle, O., Vapnik, V., Bousquet, O. and Mukherjee, S. (2002) Choosing Multiple Parameters for Support Vector Machines, *Machine Learning*, **46**, 131-159.

Chapelle, O., Weston, J. and Scholkopf, B. (2003) Cluster Kernels for Semi-Supervised Learning. In Becker, S., Thrun, S. and Obermayer, K. (eds), *Advances in Neural Information Processing Systems 15*. The MIT Press, Cambridge, MA, 585-592.

Cheng, B.Y., Carbonell, J.G. and Klein-Seetharaman, J. (2005) Protein classification based on text document classification techniques, *Proteins*, **58**, 955-970.

Chou, K.C. (2001) Prediction of protein cellular attributes using pseudo-amino acid composition, *Proteins*, **43**, 246-255.

Chou, K.C. and Cai, Y.D. (2005) Predicting protein localization in budding yeast, *Bioinformatics (Oxford, England)*, **21**, 944-950.

Chou, K.C. and Elrod, D.W. (1999) Protein subcellular location prediction, *Protein engineering*, **12**, 107-118.

Chou, K.C. and Shen, H.B. (2006) Large-Scale Predictions of Gram-Negative Bacterial Protein Subcellular Locations, *J. Proteome Res.*, **5**, 3420-3428.

Chung, C.H., Levy, S., Chaurand, P. and Carbone, D.P. (2006) Genomics and proteomics: Emerging technologies in clinical cancer research, *Crit Rev Oncol Hematol*.

Collobert, R., Sinz, F., Weston, J. and Bottou, L. (2006) Large Scale Transductive SVMs, *The Journal of Machine Learning Research*, **7**, 1687-1712.

Cortes, C. and Vapnik, V. (1995) Support-vector networks, *Machine Learning*, **20**, 273-297.

DeGroot, M.H. and Schervish, M.J. (2002) *Probability and statistics*. Addison-Wesley, Boston.

Dempster, A.P., Laird, N.M. and Rubin, D.B. (1977) Maximum Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**, 1-38.

Deng, M., Mehta, S., Sun, F. and Chen, T. (2002) Inferring Domain-Domain Interactions From Protein-Protein Interactions, *Genome research*, **12**, 1540-1548.

- Donnes, P. and Hoglund, A. (2004) Predicting protein subcellular localization: past, present, and future, *Genomics, proteomics & bioinformatics / Beijing Genomics Institute*, **2**, 209-215.
- Drish, J. (2001) Obtaining calibrated probability estimates from support vector machines, *Technical Report -- University of California at San Diego*.
- Dubchak, I., Muchnik, I., Mayor, C., Dralyuk, I. and Kim, S.H. (1999) Recognition of a protein fold in the context of the Structural Classification of Proteins (SCOP) classification, *Proteins*, **35**, 401-407.
- Duda, R.O., Hart, P.E. and Stork, D.G. (2001) *Pattern classification*. Wiley, New York.
- Eskin, E. and Agichtein, E. (2004) Combining text mining and sequence analysis to discover protein functional regions. *9th Pacific Symposium on Biocomputing*. Hawaii, 288-299.
- Fawcett, T. (2006) An introduction to ROC analysis, *Pattern Recognition Letters*, **27**, 861-874.
- Fofanov, Y. (2004) How independent are the appearances of n-mers in different genomes?, *Bioinformatics (Oxford, England)*, **20**, 2421-2428.
- Ganapathiraju, M., Weisser, D., Rosenfeld, R., Carbonell, J., Reddy, R. and Klein-Seetharaman, J. (2002) Comparative n-gram analysis of whole-genome protein sequences. *HLT '02: Human Language Technologies Conference*. San Diego, California.
- Gardy, J.L. and Brinkman, F.S.L. (2006) Methods for predicting bacterial protein subcellular localization, *Nat Rev Micro*, **4**, 741-751.
- Gardy, J.L., Laird, M.R., Chen, F., Rey, S., Walsh, C.J., Ester, M. and Brinkman, F.S. (2005) PSORTb v.2.0: expanded prediction of bacterial protein subcellular localization and insights gained from comparative proteome analysis, *Bioinformatics (Oxford, England)*, **21**, 617-623.
- Gardy, J.L., Spencer, C., Wang, K., Ester, M., Tusnady, G.E., Simon, I., Hua, S., deFays, K., Lambert, C., Nakai, K. and Brinkman, F.S. (2003) PSORT-B: Improving protein subcellular localization prediction for Gram-negative bacteria, *Nucleic acids research*, **31**, 3613-3617.
- Garg, A., Bhasin, M. and Raghava, G.P. (2005) SVM-based method for subcellular localization of human proteins using amino acid compositions, their order and similarity search, *J Biol Chem*, **280**, 14427-14432.
- Guda, C., Fahy, E. and Subramaniam, S. (2004) MITOPRED: a genome-scale method for prediction of nucleus-encoded mitochondrial proteins, *Bioinformatics (Oxford, England)*, **20**, 1785-1794.

- Guda, C. and Subramaniam, S. (2005) pTARGET [corrected] a new method for predicting protein subcellular localization in eukaryotes, *Bioinformatics (Oxford, England)*, **21**, 3963-3969.
- Hamady, M., Cheung, T.H., Resing, K., Cios, K.J. and Knight, R. (2005) Key challenges in proteomics and proteoinformatics. Progress in proteins, *IEEE Eng Med Biol Mag*, **24**, 34-40.
- Hastie, T., Tibshirani, R. and Friedman, J.H. (2001) *The elements of statistical learning : data mining, inference, and prediction*. Springer, New York.
- Hofmann, T. (2001) Unsupervised Learning by Probabilistic Latent Semantic Analysis, *Machine Learning*, **42**, 177-196.
- Hogg, R.V., McKean, J.W. and Craig, A.T. (2005) *Introduction to mathematical statistics*. Pearson Education, Upper Saddle River, N.J.
- Hoglund, A., Blum, T., Brady, S., Donnes, P., Miguel, J.S., Rocheford, M., Kohlbacher, O. and Shatkay, H. (2006) Significantly improved prediction of subcellular localization by integrating text and protein sequence data, *Pacific Symposium on Biocomputing*, 16-27.
- Hoglund, A., Donnes, P., Blum, T., Adolph, H.W. and Kohlbacher, O. (2006) MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs and amino acid composition, *Bioinformatics (Oxford, England)*, **22**, 1158-1165.
- Holland, I.B. (2004) Translocation of bacterial proteins - an overview, *Biochimica et biophysica acta*, **1694**, 5-16.
- Horton, P. and Nakai, K. (1997) Better prediction of protein cellular localization sites with the k nearest neighbors classifier, *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB*, **5**, 147-152.
- Hua, S. and Sun, Z. (2001) Support vector machine approach for protein subcellular localization prediction, *Bioinformatics (Oxford, England)*, **17**, 721-728.
- Huang, Y. and Li, Y. (2004) Prediction of protein subcellular locations using fuzzy k-NN method, *Bioinformatics (Oxford, England)*, **20**, 21-28.
- Ideker, T., Thorsson, V., Ranish, J.A., Christmas, R., Buhler, J., Eng, J.K., Bumgarner, R., Goodlett, D.R., Aebersold, R. and Hood, L. (2001) Integrated genomic and proteomic analyses of a systematically perturbed metabolic network, *Science*, **292**, 929-934.
- Joachims, T. (1999) Transductive inference for text classification using support vector machines. *Sixteenth International Conference on Machine Learning (ICML 1999)*. Morgan Kaufmann, Bled, Slovenia, 200-209.

Kasabov, N. and Pang, S. (2004) Transductive support vector machines and applications in bioinformatics for promoter recognition, *Neural Information Processing - Letters and Reviews*, **3**, 31-38.

Kersey, P., Bower, L., Morris, L., Horne, A., Petryszak, R., Kanz, C., Kanapin, A., Das, U., Michoud, K., Phan, I., Gattiker, A., Kulikova, T., Faruque, N., Duggan, K., McLaren, P., Reimholz, B., Duret, L., Penel, S., Reuter, I. and Apweiler, R. (2005) Integr8 and Genome Reviews: integrated views of complete genomes and proteomes, *Nucl. Acids Res.*, **33**, D297-302.

King, B.R. and Guda, C. (2007) ngLOC: an n-gram-based Bayesian method for estimating the subcellular proteomes of eukaryotes, *Genome Biology*, **8**, R68.

King, B.R. and Guda, C. (2008) Semi-supervised learning for classification of protein sequence data, *Scientific Programming*, **16**, 5-29.

Komatsu, S. and Yano, H. (2006) Update and challenges on proteomics in rice, *Proteomics*, **6**, 4057-4068.

Kroegel, M.A. and Scheffer, T. (2004) Multi-Relational Learning, Text Mining, and Semi-Supervised Learning for Functional Genomics, *Machine Learning*, **57**, 61-81.

Kumar, A., Agarwal, S., Heyman, J.A., Matson, S., Heidtman, M., Piccirillo, S., Umansky, L., Drawid, A., Jansen, R., Liu, Y., Cheung, K.-H., Miller, P., Gerstein, M., Roeder, G.S. and Snyder, M. (2002) Subcellular localization of the yeast proteome, *Genes Dev.*, **16**, 707-719.

Lee, K., Kim, D.-W., Na, D., Lee, K.H. and Lee, D. (2006) PLPD: reliable protein localization prediction from imbalanced and overlapped datasets, *Nucl. Acids Res.*, **34**, 4655-4666.

Lei, Z. and Dai, Y. (2005) An SVM-based system for predicting protein subnuclear localizations, *BMC bioinformatics*, **6**, 291.

Lei, Z. and Dai, Y. (2006) Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction, *BMC bioinformatics*, **7**, 491.

Li, F.M. and Li, Q.Z. (2008) Using pseudo amino acid composition to predict protein subnuclear location with improved hybrid approach, *Amino Acids*, **34**, 119-125.

Li, G. and Sheng, H. (2007) Extracting Features from Gene Ontology for the Identification of Protein Subcellular Location by Semantic Similarity Measurement. In, *Emerging Technologies in Knowledge Discovery and Data Mining*. 112-118.

Li, W. and Godzik, A. (2006) Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences, *Bioinformatics (Oxford, England)*, **22**, 1658-1659.

Ling, C.X., Huang, J. and Zhang, H. (2003) AUC: A Better Measure than Accuracy in Comparing Learning Algorithms. *Advances in Artificial Intelligence: 16th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2003*. Halifax, Canada, 991-991.

Lio, P. (2003) Wavelets in bioinformatics and computational biology: state of art and perspectives, *Bioinformatics (Oxford, England)*, **19**, 2-9.

Lopez, M.F., Kristal, B.S., Chernokalskaya, E., Lazarev, A., Shestopalov, A.I., Bogdanova, A. and Robinson, M. (2000) High-throughput profiling of the mitochondrial proteome using affinity fractionation and automation, *Electrophoresis*, **21**, 3427-3440.

Lu, Y., Tian, Q., Liu, F., Sanchez, M. and Wang, Y. (2007) Interactive semisupervised learning for microarray analysis, *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM*, **4**, 190-203.

Lu, Z. and Hunter, L. (2005) Go molecular function terms are predictive of subcellular localization, *Pacific Symposium on Biocomputing*, 151-161.

Lu, Z., Szafron, D., Greiner, R., Lu, P., Wishart, D.S., Poulin, B., Anvik, J., Macdonell, C. and Eisner, R. (2004) Predicting subcellular localization of proteins using machine-learned classifiers, *Bioinformatics (Oxford, England)*, **20**, 547-556.

Mandel, M.J., Stabb, E.V. and Ruby, E.G. (2008) Comparative genomics-based investigation of resequencing targets in *Vibrio fischeri*: focus on point miscalls and artefactual expansions, *BMC genomics*, **9**, 138.

Marcotte, E.M., Xenarios, I., van Der Bliek, A.M. and Eisenberg, D. (2000) Localizing proteins in the cell from their phylogenetic profiles, *Proceedings of the National Academy of Sciences of the United States of America*, **97**, 12115-12120.

Matthews, B.W. (1975) Comparison of the predicted and observed secondary structure of T4 phage lysozyme, *Biochimica et biophysica acta*, **405**, 442-451.

McCallum, A.K. and Nigam, K. (1998) A comparison of event models for naive bayes text classification, *AAAI-98 Workshop on Learning for Text Categorization*, 752.

Mitchell, T.M. (1997) *Machine Learning*. McGraw-Hill, New York.

Mount, D.W. (2004) *Bioinformatics: sequence and genome analysis, 2nd ed.* Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY.

Mulder, N.J., Apweiler, R., Attwood, T.K., Bairoch, A., Bateman, A., Binns, D., Bork, P., Buillard, V., Cerutti, L., Copley, R., Courcelle, E., Das, U., Daugherty, L., Dibley, M., Finn, R., Fleischmann, W., Gough, J., Haft, D., Hulo, N., Hunter, S., Kahn, D., Kanapin, A., Kejariwal, A., Labarga, A., Langendijk-Genevaux, P.S., Lonsdale, D., Lopez, R., Letunic, I., Madera, M., Maslen, J., McAnulla, C., McDowall, J., Mistry, J., Mitchell, A., Nikolskaya, A.N., Orchard, S., Orengo, C., Petryszak, R., Selengut, J.D., Sigrist, C.J.A.,

- Thomas, P.D., Valentin, F., Wilson, D., Wu, C.H. and Yeats, C. (2007) New developments in the InterPro database, *Nucl. Acids Res.*, **35**, D224-228.
- Nair, R. and Rost, B. (2003a) Better prediction of sub-cellular localization by combining evolutionary and structural information, *Proteins*, **53**, 917-930.
- Nair, R. and Rost, B. (2003b) LOC3D: annotate sub-cellular localization for protein structures, *Nucleic acids research*, **31**, 3337-3340.
- Nakai, K. and Horton, P. (1999) PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localization, *Trends in biochemical sciences*, **24**, 34-36.
- Nakai, K. and Kanehisa, M. (1992) A knowledge base for predicting protein localization sites in eukaryotic cells, *Genomics*, **14**, 897-911.
- Nakai, K., Kidera, A. and Kanehisa, M. (1988) Cluster analysis of amino acid indices for prediction of protein structure and function, *Protein engineering*, **2**, 93-100.
- Nakashima, H. and Nishikawa, K. (1994) Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies, *Journal of molecular biology*, **238**, 54-61.
- Nigam, K., McCallum, A.K. and Mitchell, T. (2006) Semi-Supervised Text Classification Using EM. In Chapelle, O., Scholkopf, B. and Zien, A. (eds), *Semi-Supervised Learning*. The MIT Press, Cambridge, MA, 33-54.
- Nigam, K., McCallum, A.K., Thrun, S. and Mitchell, T. (2000) Text Classification from Labeled and Unlabeled Documents using EM, *Machine Learning*, **39**, 103-134.
- Nishikawa, K., Kubota, Y. and Ooi, T. (1983) Classification of proteins into groups based on amino acid composition and other characters. II. Grouping into four types, *Journal of biochemistry*, **94**, 997-1007.
- Pandey, A. and Mann, M. (2000) Proteomics to study genes and genomes, *Nature*, **405**, 837-846.
- Park, K.J. and Kanehisa, M. (2003) Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs, *Bioinformatics (Oxford, England)*, **19**, 1656-1663.
- Peng, F. and Schuurmans, D. (2003) Combining Naive Bayes and n-Gram Language Models for Text Classification. *Advances in Information Retrieval: 25th European Conference on IR Research, ECIR 2003*. Springer Berlin / Heidelberg, Pisa, Italy, 547.
- Pierleoni, A., Martelli, P.L., Fariselli, P. and Casadio, R. (2006) BaCelLo: a balanced subcellular localization predictor, *Bioinformatics (Oxford, England)*, **22**, e408-416.

- Provost, F., Fawcett, T. and Kohavi, R. (1998) The case against accuracy estimation for comparing induction algorithms. In Shavlik, J. (ed), *Proceedings of the Fifteenth International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 445–453.
- Pruitt, K.D., Tatusova, T. and Maglott, D.R. (2005) NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins, *Nucleic acids research*, **33**, D501-504.
- Reinhardt, A. and Hubbard, T. (1998) Using neural networks for prediction of the subcellular location of proteins, *Nucleic acids research*, **26**, 2230-2236.
- Rey, S., Acab, M., Gardy, J.L., Laird, M.R., deFays, K., Lambert, C. and Brinkman, F.S. (2005) PSORTdb: a protein subcellular localization database for bacteria, *Nucleic acids research*, **33**, D164-168.
- Rost, B. and Sander, C. (1993) Prediction of protein secondary structure at better than 70% accuracy, *Journal of molecular biology*, **232**, 584-599.
- Scott, M.S., Calafell, S.J., Thomas, D.Y. and Hallett, M.T. (2005) Refining protein subcellular localization, *PLoS computational biology*, **1**, e66.
- Scott, M.S., Thomas, D.Y. and Hallett, M.T. (2004) Predicting subcellular localization via protein motif co-occurrence, *Genome research*, **14**, 1957-1966.
- Seeger, M. (2001) Learning with Labeled and Unlabeled Data. *Institute for Adaptive and Neural Computation, Technical Report*. University of Edinburgh.
- Segal, E. (2003) Discovering molecular pathways from protein interaction and gene expression data, *Bioinformatics (Oxford, England)*, **19**, 264-272.
- Shen, H.-B. and Chou, K.-C. (2007) Gpos-PLoc: an ensemble classifier for predicting subcellular localization of Gram-positive bacterial proteins, *Protein Engineering, Design and Selection*, **20**, 39-46.
- Shen, H.-B. and Chou, K.-C. (2007) Nuc-PLoc: a new web-server for predicting protein subnuclear localization by fusing PseAA composition and PsePSSM, *Protein Engineering, Design and Selection*, gzm057.
- Shi, J., Zhang, S., Pan, Q. and Zhang, Y. (2007a) Using Decision Templates to Predict Subcellular Localization of Protein. In, *Pattern Recognition in Bioinformatics*. 71-83.
- Shi, J.Y., Zhang, S.W., Pan, Q., Cheng, Y.M. and Xie, J. (2007b) Prediction of protein subcellular localization by support vector machines using multi-scale energy and pseudo amino acid composition, *Amino Acids*, **33**, 69-74.
- Shin, H., Tsuda, K. and Scholkopf, B. (2004) Protein functional class prediction with a combined graph, *Proceedings of the Korean Data Mining Conference*, 200–219.

- Smith, C. (2003) Drug target validation: Hitting the target, *Nature*, **422**, 341, 343, 345 passim.
- Sprenger, J., Fink, J.L. and Teasdale, R.D. (2006) Evaluation and comparison of mammalian subcellular localization prediction methods, *BMC bioinformatics*, **7 Suppl 5**, S3.
- Su, E., Chiu, H.-S., Lo, A., Hwang, J.-K., Sung, T.-Y. and Hsu, W.-L. (2007) Protein subcellular localization prediction based on compartment-specific features and structure conservation, *BMC bioinformatics*, **8**, 330.
- Szummer, M. and Jaakkola, T. (2001) Kernel expansions with unlabeled examples, *Advances in Neural Information Processing Systems*, **13**, 626-632.
- Taylor, S.W., Fahy, E. and Ghosh, S.S. (2003) Global organellar proteomics, *Trends in biotechnology*, **21**, 82-88.
- van Rijsbergen, C. (1979) *Information Retrieval*. Butterworths, London.
- Vapnik, V. (2006) Transductive Inference and Semi-Supervised Learning. In Chapelle, O., Scholkopf, B. and Zien, A. (eds), *Semi-Supervised Learning*. The MIT Press, Cambridge, MA, 453-472.
- Vapnik, V.N. (1995) *The nature of statistical learning theory*. Springer, New York.
- Vapnik, V.N. (1998) *Statistical learning theory*. Wiley, New York.
- Weston, J., Leslie, C., Ie, E., Zhou, D., Elisseeff, A. and Noble, W.S. (2005) Semi-supervised protein classification using cluster kernels, *Bioinformatics (Oxford, England)*, **21**, 3241-3247.
- Weston, J., Leslie, C., Zhou, D., Elisseeff, A. and Noble, W.S. (2004) Semi-supervised protein classification using cluster kernels. In Thrun, S., Saul, S. and Scholkopf, B. (eds), *Advances in Neural Information Processing Systems (NIPS-2003)*. The MIT Press, Cambridge, MA.
- Wikipedia (2006) Organelle. *Wikipedia, The Free Encyclopedia*.
- Witten, I.H. and Frank, E. (2005) *Data mining : practical machine learning tools and techniques*. Morgan Kaufmann, Amsterdam; Boston.
- Xie, D., Li, A., Wang, M., Fan, Z. and Feng, H. (2005) LOCSVMPSI: a web server for subcellular localization of eukaryotic proteins using SVM and profile of PSI-BLAST, *Nucleic acids research*, **33**, W105-110.
- Yang, Y. (1999) An Evaluation of Statistical Approaches to Text Categorization, *Information Retrieval*, **1**, 69-90.

- Yang, Y. and Liu, X. (1999) A re-examination of text categorization methods. *22nd annual international ACM SIGIR conference on Research and development in information retrieval*. Berkeley, CA, 42-49.
- Yu, C.S., Chen, Y.C., Lu, C.H. and Hwang, J.K. (2006) Prediction of protein subcellular localization, *Proteins*, **64**, 643-651.
- Yu, C.S., Lin, C.J. and Hwang, J.K. (2004) Predicting subcellular localization of proteins for Gram-negative bacteria by support vector machines based on n-peptide compositions, *Protein Sci*, **13**, 1402-1406.
- Zhang, S., Zhang, Y., Li, J., Yang, H., Cheng, Y. and Zhou, G. (2007) A New Hybrid Approach to Predict Subcellular Localization by Incorporating Protein Evolutionary Conservation Information. In, *Life System Modeling and Simulation*. 172-179.
- Zhu, X. (2006) Semi-Supervised Learning Literature Survey, *Computer Sciences, University of Wisconsin-Madison, Tech. Rep*, **1530**.
- Zhu, X., Ghahramani, Z. and Lafferty, J. (2003) Semi-supervised learning using Gaussian fields and harmonic functions. *Twentieth International Conference on Machine Learning (ICML-2003)*. Washington DC.

Index

A

amino acid.....4, 24, 46
 amino acid composition 28, 29, 74
 artificial neural network.....29, 36
 AUC.....102

B

BaCellLo.....39
 bacteria
 Gram-negative 95
 Gram-positive 95
 Bayes rule 16, 126
 Bayesian classification.....15, 16
 Bernoulli trial.....48
 bioinformatics 3

C

cancer.....1
 cd-hit.....142
 CELLO 32, 38
 Central Dogma of Molecular Biology.....7
 classification 11
 classifier.....14
 clustering.....12
 k-means.....22
 coding region See protein:coding region
 confidence score.....54, 55, 69, 104
 covariant discriminant analysis 28

D

data mining 11, 13
 database.....5
 databases.....2
 discriminative.....16
 disease.....23
 cancer.....26
 distribution
 binomial.....48
 multinomial.....49
 DNA.....1, 7
 drug targeting.....95

E

EM See Expectation Maximization
 EM-CS.....138
 EM- λ 134
 Expectation Maximization 12, 21, 124, 130, 186
 Generalized EM.....188

F

F_1 measure..... 101, 145
 macro-averaged..... 101
 false positive rate..... 60, 102
 feature space..... 13, 46
 feature vector..... 14

G

gene expression 7
 Gene Ontology 4, 36
 generative 16
 genome 1, 4, 7
 Gneg-PLoc 110

H

high-throughput proteomic methods..... 9
 homology..... 88
 hypersphere 37

I

identically and independently distributed..... 20, 48, 50
 IID See identically and independely distributed
 inductive inference 41
 Integr8 58
 InterPro..... 40

J

jack-knife validation..... 100

K

k -nearest neighbors..... 31, 39

L

LaPlace correction..... 52
 leave-one-out validation 100
 likelihood..... 17
 LOC3D..... 36
 LOC3Dnet 35
 localization signal.....36, 40, 65, 68, 96
 LOCSVMPSI 32
 log odds ratio..... 54

M

machine learning 11, 13, 14
 macro-average 101, 145
 macro-averaged F_1 145
 Matthews correlation coefficient..... 60

maximum likelihood estimation.....21, **185**
MCC*See* Matthews correlation coefficient
micro-average 145
mitochondria 26, 38
mixture model 21
MLCS *See* confidence score
MLE.....*See* maximum likelihood estimation
Multi-EM 134
multi-localized confidence score. *See* confidence score

N

Naïve Bayes 50, 88, 128
NCBI.....4, 46
ngLOC 45
 data 57
 ngLOC-X 55, 76, 82, 90, 93, 112
n-gram.....46, 48
 class-unique 67
NNPSL.....30
nucleus 38
Nuc-PLoc 39

O

objective function.....18, 19
open reading frame (ORF) *See* protein:open reading frame
organelle 26
overall accuracy 59
overfitting 19

P

pattern recognition *See* classification
performance measure 14, 59
PLOC 32, 33, 74, 89
position specific scoring matrix 34
post-translational modification.....7
precision.....60
primary structure..... *See* protein:primary structure
probability.....11
promoter recognition.....42
protein.....1, 3, 4, 7, 23
 coding region 4
 domain 86
 function.....2, 23
 interaction.....23, 26
 open reading frame 4, 6
 physicochemical properties.....29, 34, 35
 primary structure.....4, 5, 8, 23, 46
 secondary structure 23, 29
 subcellular localization 25
 tertiary structure.....4, 23, 36
proteome 1, 7, 77, 90
 data 58, 99
 subcellular proteome.....39
Proteome Analyst.....97
proteomics.....2
PseAA *See* pseudo amino acid composition
pseudo amino acid composition 29, 34, 39
PSI-BLAST.....32, 34

PSLdoc 34
PSLT 38, 40, 86
 PSLT2 40
PSORT 28, 72, 96
 v2.0 35
PSORTb 96, 110
PSORTdb 99
pTARGET 29, 72

R

random variable.....15
receiver operating characteristic.....*See* ROC
RNA 7
ROC curve.....64, 101, 104, 118
 macro-averaged.....102

S

sample selection bias 9, 21, 50
semi-supervised learning.....10, 12, 123, *See*
sensitivity 60
sequence *See* protein:primary structure
specificity 60
statistical inference.....12, *See* classification
string kernels 41
subcellular localization.....47, *See* protein:subcellular
 localization
subcellular proteome 81, *See* proteome:subcellular
 proteome
suborganelle 38
supervised learning.....11, 122
support vector machine 30, 32, 41, 89, 97
 transductive.....42, 124, **139**
SVM.....*See* support vector machine
Swiss-Prot 5, 57, 98, 119

T

targeting signal 91
ten-fold cross validation 61
tertiary structure *See* protein:tertiary structure
test data 19
text classification.....41, 46
training data.....13, 20, 21
transcription.....7
transductive inference.....41
translation 7
transport signal 96
TrEMBL 4, 6
true positive rate 102
TSVM..... *See* support vector machine:transductive

U

unbalanced data 9
UniProt 4, 6
UniverSVM.....**141**
unlabeled data.....10
unsupervised learning.....12, 22

W

wavelet 35