PREDICTION OF PROTEIN SUBCELLULAR LOCALIZATION BASED ON PRIMARY SEQUENCE DATA

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

MERT ÖZARAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

September 2003

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ayşe Kiper
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Volkan Atalay
Supervisor

Examining Committee Members

Prof. Dr. Faruk Polat

Prof. Dr. Kemal Leblebicioğlu

Assoc. Prof. Dr. Volkan Atalay

Assist. Prof. Dr. Rengül Çetin Atalay

Dr. Özlen Konu

ABSTRACT

PREDICTION OF PROTEIN SUBCELLULAR LOCALIZATION BASED ON PRIMARY SEQUENCE DATA

Özarar, Mert

M.Sc., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Volkan Atalay

September 2003, 61 pages

Subcellular localization is crucial for determining the functions of proteins. A system called *prediction of protein subcellular localization (P2SL)* that predicts the subcellular localization of proteins in eukaryotic organisms based on the amino acid content of primary sequences using amino acid order is designed. The approach for prediction is to find the most frequent motifs for each protein in a given class based on clustering via self organizing maps and then to use these most frequent motifs as features for classification by the help of multi layer perceptrons. This approach allows a classification independent of the length of the sequence. In addition to these, the use of a new encoding scheme is described for the amino acids that conserves biological function based on *point of accepted mutations (PAM)* substitution matrix. The statistical test results of the system is presented on a four class problem. P2SL achieves slightly higher prediction accuracy than the similar studies.

Keywords: subcellular localization, protein sorting, clustering, classification.

ÖZ

BİRİNCİL DİZİ VERİ TEMELLİ PROTEİN HÜCREİÇİ YER BELİRLEME TAHMİNİ

Özarar, Mert

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Volkan Atalay

Eylül 2003, 61 sayfa

Proteinlerin işlevlerinin belirlenmesinde hücre içi yer belirleme çok önemlidir. Bu çalışmada, ökaryotik canlılarda, amino asit sırası kullanılarak amino asit birincil dizi içeriği temelli, protein hücre içi yer belirlenmesi için, P2SL adında, yeni bir sistem tasarlanmıştır. Tahmin yaklaşımı, öz düzenlemeli haritalara dayananarak verilen bir sınıfta her protein için, en yaygın motifleri bulmak ve bunları ,öznitelik olarak kullanarak çok katmanlı perseptronların yardımıyla sınıflandırmaktır. Bu yaklaşım dizi uzunluğundan bağımsız bir sınıflandırmaya izin vermektedir. Bunlara ek ve daha önemlisi, kabul edilebilir nokta mutasyon (PAM) değiştirme matrisi temelli, biyolojik işlevi muhafaza eden, yeni bir kodlama planı kullanımı tarif edilmektedir. Dört sınıflı bir problemde, sistemin istatistiksel test sonuçları sunulmaktadır. P2SL, benzer çalışmalardan biraz daha yüksek tahmin doğruluğuna ulaşmıştır.

Anahtar Kelimeler: hücre içi yer belirleme, protein sıralama, kümelendirme, sınıflandırma.

To my family

ACKNOWLEDGMENTS

I would like to thank to Volkan Atalay for his supervision, guidance and valuable suggestions throughout the development of the thesis. I would like to acknowledge my debt to Rengül Çetin Atalay for her support and assistance. I would like to thank my committee members for their feedback and comments. I would also thank my family for their encouragement to finish my study. Finally, it is my pleasure to express my deepest gratitude to my friends for sharing hard times of my work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation and Purpose

Classification of patterns in biological sequences is an important task in bioinformatics. Applications range from the identification of functional motifs in DNA sequences to the prediction of protein secondary and tertiary structure. Genome sequencing projects provide the scientific community with an ever-increasing rate of predicted protein sequences. To analyze these biochemically uncharacterized sequences, computer based methods have been established to provide researchers an initial characterization. Many of these methods make use of sequence similarity to already described proteins.

Eukaryotic cells are subdivided into functionally seperate membrane enclosed compartments. Each compartment and vicinity contain functionally linked proteins related to the activity of that cell compartment [1]. Most proteins in an eukaryotic cell are encoded in the nuclear genome and first synthesized in the cytosol, then carried to specified locations, such as mitochondria or nucleus which is named as subcellular localization of the protein in the cell. In most cases, the information determining the subcellular localization site is represented as a short amino acid sequence segment called a protein sorting signal. Subcellular protein sorting, i.e. the processes through which

proteins are routed to their proper final destination within a cell, is a fundamental aspect of cellular life. In many cases, sorting depends on signals that can already be identified by looking at the primary structure of a protein.

Owing to the dramatic increase in the number of proteins sent to the public data bank during the last few years, it is highly desirable to develop an effective algorithm to predict the subcellular location of new proteins so as to expedite the process of deducing their function.

Studying subcellular localization is useful for understanding the disease mechanism and developing novel drugs. Due to cellular functions are often localized in specific compartments, prediction of unknown proteins may be used to obtain of its function. If the rules for the prediction were biologically interpretable, this knowledge could help in designing artificial proteins with desired properties. Therefore, an automatic and reliable prediction system for protein subcellular localization would be very useful.

The aim of this work is to design and develop a system called *prediction of protein subcellular localization (P2SL)* that predicts the subcellular localization of proteins in eukaryotic organisms based on the amino acid content of primary sequences. The amino acid composition in the full or partial sequences can be taken as global features and the order may represent the local features such as the sequence order of amino acids that are found in protein sequence motifs [2]. In this study, we are interested in the prediction using only local features. Our approach for prediction is to find the most significant motifs for each protein (class) based on clustering and then to use these most significant motifs as features for classification. This approach allows a classification independent of the length of the sequence. Another important property of the approach is to provide a means to perform reverse analysis and analysis to extract rules. In addition to these and more importantly, we describe the use of a new encoding scheme for the amino acids that conserves biological function based on *point of accepted mutations (PAM)* substitution matrix [3]. *PAM* is used to score aligned peptide sequences to determine the similarity of these

sequences. The scores in *PAM* are derived by comparing aligned sequences of proteins with known homology and determining the observed. By using *PAM* substitution matrix, we believe that we are able to represent the chemical differences of each amino acid in protein sequences. In the literature, each amino acid is traditionally represented in binary form independent of their chemical properties. In this study, we present results of our system on multiple classes.

The proteins of plant organism are not considered in the data sets. We use the data set provided by Emanuelsson et.al. [4]. There are four kinds of classes in our system.

- Signal peptides (SPs)
- Cytoplasmic targeting peptides (CPs)
- Mitochondrial targeting peptides (MPs)
- Nuclear targeting peptides (NPs)

The used signals are known to be the on the N-terminal of the protein.

In a computer science perspective, it is a multi-class pattern recognition problem applied to molecular biology. The problem is finding the suitable patterns from variable length strings which are converted into multi-dimensional labelled feature vectors using a special encoding. Our aim is to find out most significant substrings from the local features. There can be many such substrings from the whole sequence and those should be classified in the recognition phase. By including a clustering approach as, the reference vectors provide a single result for similar substrings and this helps the classifier for recognition. Self organizing map (SOM) is used for clustering and multi layer perceptrons (MLPs) are used for classification as machine learning techniques.

Self organizing maps are chosen since a clustering mechanism should take place among the feature vectors which forms an infra structure for the classification phase. The frequency distributions of the SOM cells are used in the

MLPs whose purpose is a nonlinear classifier for the feature vectors. Multilayer batch perceptrons are used together with back-propagation learning algorithm. The results obtained from the experiments are slightly better than the previous ones.

1.2 Materials

The main idea for the prediction of protein subcellular localization using local features is based on finding the substrings which are common for a protein class and infrequent for the other classes. Such substrings are called as motifs. We use a self organizing map for this purpose. For an unknown input sequence, we determine which motifs exist and the sequence is classified according to this information. The flow diagram of *P2SL* is illustrated in Figure 1.1.

The vast majority of the proteins found in living organisms are composed of only 20 different kinds of amino acids, repeated many times and strung together in a particular order. Each type of protein has its own unique sequence of amino acids; this sequence, known as its primary structure, actually determines the shape and function of the protein. The input to the system is amino

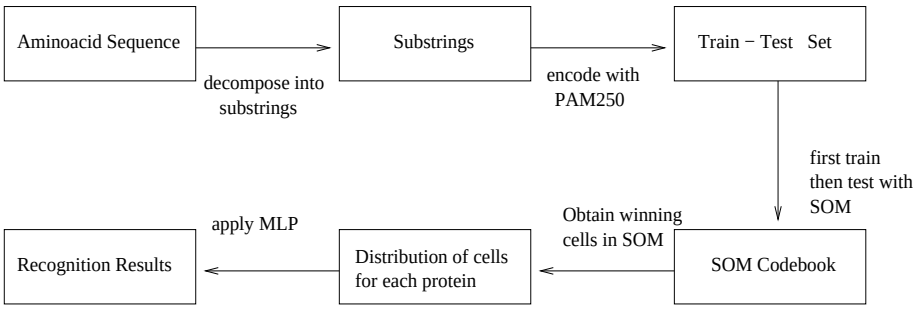

Figure 1.1: Flow diagram of *P2SL*.

acid sequences. The sequences are extracted from this data. The primary sequence is then decomposed into substrings. Each substring is encoded with *PAM250* substitution matrix. We apply clustering on the encoded substring via a self organizing map. During the training phase, motifs for each class are determined. Throughout the test phase, when the substrings of an unknown input sequence is given, according to the winning nodes in the self organizing map, a distribution of cells is formed for each class. Multi layer perceptrons are used for classification among the distribution.

1.3 Organization

The organization of the thesis is as follows. In Chapter 2, the background information about protein subcellular localization, related work and the machine learning techniques are explained in detail. The data and computational methods used in this study are presented in Chapter 3 together with experiments and comments on the results are discussed, circumstantiately. The thesis concludes with Chapter 4, the eventual improvements are indicated together with conclusions and future work.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, the biological background of the classes are presented with substitution matrices and the protein databases. An overview of related studies are done on the field of subcellular localization. The machine learning techniques used in the research are briefly explained.

2.1 PAM250 Substitution Matrix

The term substitution is often used for the alignment of two amino acid residues, since scoring schemes are frequently derived from a model of evolution that considers two protein sequences to be related via a series of point mutations. The pair-score matrix is usually symmetrical, since *Ala* aligned with *Gly* has the same meaning as *Gly* aligned with *Ala*. The simplest scoring scheme is the identity matrix. This scores 1 for an exact match of two amino acids, and 0 for a mismatch. Although the identity matrix is appealing in its simplicity, it does not reflect adequately similarities observed between proteins that have similar three dimensional structures. More sophisticated schemes take into account conservative substitutions. For example, *Val* aligned with *Leu* might score $+4$, but *Glu* with *Leu*, -3.

Until recently, matrices referred to as PAM or Dayhoff were the most widely

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	2																			
R	-2	6																		
N	0	0	2																	
D	0	-1	2	4																
C	-2	-4	-4	-5	4															
Q	0	1	1	2	-5	4														
E	0	-1	1	3	-5	2	4													
G	1	-3	0	1	-3	-1	0	5												
H	-1	2	2	1	-3	3	1	-2	6											
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5										
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6									
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5								
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6							
F	-4	-4	-4	-6	-4	-5	-5	-5	-2	1	2	-5	0	9						
P	1	0	-1	-1	-3	0	-1	-1	0	-2	-3	-1	-2	-5	6					
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	3				
T	1	-1	0	0	-2	-1	0	0	-1	0	-2	0	-1	-2	0	1	3			
W	-6	2	-4	-7	-8	-5	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	17		
Y	-3	-4	-2	-4	0	-4	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	0	10	
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	2	4

Figure 2.1: The PAM250 substitution matrix.

used. PAM matrices were derived by first aligning a small number of families of protein sequences by eye, then counting the observed amino acid substitutions within the families and normalizing the counts before extrapolating the observed substitutions to those expected at different evolutionary distances [5]. The measure of evolutionary distance used was the Point of Accepted Mutations, or PAM, and the most commonly applied matrix was that at 250 PAMS, normally known as PAM250 which is illustrated in Figure 2.1. It is a symmetric matrix so only the diagonal and lower triangular entries are shown.

2.2 SWISS PROT Database

All the sequences used in this study are extracted from the SWISS PROT [6] protein database. In this section, the format and the properties of the database are presented.

SWISS PROT is an annotated protein sequence database. It was established in 1986 and maintained collaboratively, since 1987, by the group of Amos Bairoch first at the Department of Medical Biochemistry of the University of Geneva and now at the Swiss Institute of Bioinformatics (SIB) and the EMBL Data Library (now the EMBL Outstation - The European Bioinformatics Institute (EBI)).

2.2.1 General Structure

The SWISS PROT protein knowledgebase consists of sequence entries. Sequence entries are composed of different line types, each with their own format. For standardization purposes the format of SWISS PROT follows as closely as possible that of the EMBL Nucleotide Sequence Database. In SWISS PROT, as in many sequence databases, two classes of data can be distinguished: the core data and the annotation.

For each sequence entry the core data consists of:

- The sequence data;
- The citation information (bibliographical references);
- The taxonomic data (description of the biological source of the protein).

The annotation consists of the description of the following items:

- Function(s) of the protein;
- Posttranslational modification(s). For example carbohydrates, phosphorylation, acetylation, GPI-anchor, etc.;

- Domains and sites. For example calcium-binding regions, ATP-binding sites, zinc fingers, homeoboxes, SH2 and SH3 domains, kringle, etc.;
- Secondary structure. For example alpha helix, beta sheet, etc.;
- Quaternary structure. For example homodimer, heterotrimer, etc.;
- Similarities to other proteins;
- Disease(s) associated with any number of deficiencies in the protein;
- Sequence conflicts, variants, etc.

In SWISS PROT, annotation is mainly found in the comment lines (CC), in the feature table (FT) and in the keyword lines (KW). Most comments are classified by 'topics'; this approach permits the easy retrieval of specific categories of data from the database.

The SWISS PROT protein sequence database is composed of sequence entries. Each entry corresponds to a single contiguous sequence as contributed to the bank or reported in the literature. In some cases, entries have been assembled from several papers that report overlapping sequence regions. Conversely, a single paper can provide data for several entries, e.g. when related sequences from different organisms are reported.

References to positions within a sequence are made using sequential numbering, beginning with 1 at the N-terminal end of the sequence.

Except for initiator N-terminal methionine residues, which are not included in a sequence when their absence from the mature sequence has been proven, the sequence data correspond to the precursor form of a protein before post translational modifications and processing.

2.3 Biological Background

The biological meaning and properties should be analyzed for the type of classes used in the study to comprehend the concept of subcellular localization sites in eukaryotic cells.

2.3.1 Signal Peptides

The properties of the amino acids that constitute the signal peptide region of a protein are the significant factors determining interaction with the protein transport system, hence the destination to which that protein is delivered. Different classes of signal peptide are used to specify different cellular placement. It should be reiterated that not all proteins possess signalling regions; those which do not are maintained in the cytoplasm. The common structure of signal peptides from various proteins is commonly described as a positively charged n-region, followed by a hydrophobic h-region and a neutral but polar c-region. The $(-3, -1)$-rule states that the residues at positions -3 and -1 (relative to the cleavage site) must be small and neutral for cleavage to occur correctly.

Different organelles have adopted subtle variations on the general theme of signal peptide targeting of proteins.

2.3.2 Cytosolic Proteins

Cytosolic proteins, that have an uptake-targeting sequence, combine with unfolding factors in the cytoplasm, that disrupt higher levels of protein folding. They bind to receptors on the outer membrane of the mitochondrion. Membrane translocation to the transport channel proteins occurs. They pass through the ouble membrane. Matrix protease cleaves the uptake-targeting sequence so the mature protein refolds.

2.3.3 Mitochondrial Proteins

Mitochondria are double membrane bound organelles involved in the production of energy. The internal membrane relies on an electrical potential to drive protein translocation. Mitochondrial proteins (with the exception of those that are produced by the organelles own ribosomes) are made by cytosolic free ribosomes and imported post-translationally by receptors which

reside at points of contact between the inner and outer membranes. Most of these proteins are present in the cytoplasm as precursors of the active forms. In contrast to nuclear proteins, those targeted across the mitochondrial membranes are only able to do so in an unfolded state.

2.3.4 Nuclear Proteins

All nuclear proteins are synthesised on free ribosomes in the cytoplasm. Proteins destined for the nucleus have to negotiate the nuclear membrane; a double membrane. Unlike the situation elsewhere these proteins are able to cross into the nucleus from the cytoplasm whilst still folded. The reason for this is due largely to the existence of specialised nuclear pores, which govern the transposition process, and involve the direct expenditure of energy. Protein coated gold beads have been instrumental in demonstrating the selectivity of the nuclear pore complexes. Comparison of a large number of nuclear proteins shows the presence of a short sequence of amino acids specifying nuclear import, although quite different sequences are utilised by different proteins. These nuclear localisation signals may be at the N-terminal or C-terminal ends of proteins.

Another unusual and important feature of nuclear proteins is that nearly all mature functional molecules still possess their signal peptides, i.e. there is no cleavage of this signal region upon importation into the nucleus. The reason for this becomes clear when one understands the processes that accompany cell division; during both mitosis and meiosis the nuclear envelope is completely (in higher eukaryotes) or partially (in lower eukaryotes) dissolved to allow proper segregation of the cellular contents, including the chromosomes. All nuclear proteins are exposed to the cytoplasm. However, once the nuclear membrane reforms around the chromosomes, these same proteins are redirected to the new nucleus because they still possess the appropriate signal peptides. If the signals had been removed then those proteins would not be shuttled back to the nucleus.

Some proteins are prevented from entering the nucleus immediately following their synthesis by masking the nuclear localization signal. This can be achieved via either chemical modification of the signal, or by interaction with inhibitory cytosolic proteins.

2.4 Related Work

Several attempts have been made to predict protein subcellular localization. In this subsection, the major studies on our subject explained briefly. Most of these prediction methods can be classified into two categories: one is based on the recognition of protein N-terminal sorting signals (i.e. local features) and the other is based on amino acid composition (i.e. global features). Among them, PSORT and MTS use global features, while iPSORT, MitoProt, TargetP, SignalP and ChloroP use local features. SortPred uses both global and local features. An overview of the related work is presented in the Figure 2.2.

Except the SortPred, only one of the feature types either global or local is used during the classification. Some studies are dichomtimers, while some are for 3 classes for non-plant organisms. Hidden Markov Models and Neural Networks become highly dominant in the recent years. General prediction accuracy is increased to 91%.

2.4.1 PSORT

PSORT is a computer program for the prediction of protein localization sites in cells. It was developed by Nakai et. al. [7] based on rules for various sequence features of known protein signals. A knowledge base by organizing various experimental and computational observations as a collection of if-then rules have been constructed. An expert system, which utilizes this knowledge base, for predicting localization sites of proteins only from the information on the amino acid sequence and the source origin were reported. 401 eukaryotic proteins with known localization sites (subcellular and extracellular) were collected and divided into training data and testing data. 14

Method	Authors	Year	Type	Class(es)	Technique Used	Accuracy
PSORT	Nakai et. al.	1992	Global	MP, SP, cTP	Expert System	64%
iPSORT	Nakai et. al.	1993	Local	MP, SP, cTP	Decison Trees	77%
MitoProt II	Claros et. al.	1996	Local	MP	Discriminant Analysis	75-97%
MTS	Fujiwara et. al.	1997	Global	MP	HMM	87%
SignalP	Nielsen et. al.	1997	Local	SP	Neural Networks	96%
ChloroP	Nielsen et. al.	1998	Local	CTP	Neural Networks	96%
TargetP	Emanuelsson et. al.	2000	Local	SP, MP, Other,	Neural Networks	90%
SortPred	Fujiwara et. al.	2001	Local,Global	SP, MP, Other	HMM	91%

Figure 2.2: Overview of related studies

localization sites were distinguished for animal cells and 17 for plant cells. When sorting signals were not well characterized experimentally, various sequence features were computationally derived from the training data. It was found that 66% of the training data and 59% of the testing data were correctly predicted by PSORT. Overall accuracy is 64%.

Although PSORT is the ancestor of all subcellular predictors, the results of PSORT are not satisfactory as a pattern recognition study.

2.4.2 iPSORT

iPSORT is a subcellular localization site predictor for N-terminal sorting signals developed by Nakai et. al. [8] following the previous one (PSORT). The accuracy rate is increased since there is a transition from global features to local ones. Given a protein sequence , it will predict whether it contains a Signal Peptide (SP), Mitochondrial Targeting Peptide (MP), or Chloroplast Transit Peptide (cTP). The structure of iPSORT is simply a decision list consisting of 3 nodes (2 for non-plant).

At the first node, the protein sequence is checked if it is a SP or not. If it is predicted as SP, then the output is simply "SP". At the second node, the protein sequence is judged if it is either a MP, or cTP. If it is determined not to be either of them, the sequence is predicted to be "Other". (For non-plant sequences, this is the final node). At the last node, the protein sequence is judged if it is a MP or not. If yes, then "MP" is the output, and if no, "cTP" is seen in the output. The rules deciding whether or not the given signals contain a certain signal consists of two elements: an amino acid index rule, and an alphabet indexing pattern rule (except for SP with only an amino acid index rule). To be judged "yes" at each node, the input amino acid sequence must satisfy both of the two rules (except SP). They are explained below.

An amino acid index is a mapping from an amino acid to a numerical value. For a given amino acid, its amino acid index represents some biochemical property of the amino acid. Using these amino acid index values, the average amino acid index value of certain substrings of the input amino acid sequence are calculated at each node. To be judged "yes" at a given node, the average must exceed (or be less than) a certain threshold. Its overall recognition rate is about 77% for 3 classes in test phase. No cross validation is applied.

The usage of local features instead of global ones is an improvement as a successor of PSORT which affects the prediction accuracies positively.

2.4.3 TargetP

A neural network-based tool, TargetP, for large-scale subcellular location prediction of newly identified proteins has been developed by Emanuelsson et.al. [4]. Using N-terminal sequence information only, it discriminates between proteins destined for the mitochondrion, the chloroplast, the secretory pathway, and other localizations with a success rate of 85% (plant) or 90% non-plant). Emanuelsson et.al. have developed a data set consists of 715 SPs, 438 CPs, 371 MPs and 1214 NPs which is used in our study as well. For eukaryotic cells, they assign three classes, namely SP, MP and Other. It is built from two layers of neural networks, where the first layer contains one dedicated network for each type of proseqeunce (SP, MP, Other), and the second is an integrating network that outputs the actual prediction. All predictions are fully automatic and the expected performance profile can be customized to fit less restrictive searches for candidate proteins as well as highly conservative criteria for, e.g. database annotations. Different size of substrings are extracted from the sequences and they fed into the network.

It is the leading predictor in predicting subcellular localization that uses local features only.

2.4.4 MitoProt II

MitoProt II calculates the N-terminal protein region that can support a Mitochondrial Targeting Sequence and the cleavage site which is developed by Claros et. al. [9]. In their work, discriminant analysis has been performed with 47 parameters and a large set of mitochondrial proteins extracted from the SWISS PROT database. A computational method that facilitates the analysis and objective prediction of mitochondrially imported proteins has been developed. If only the amino acid sequence is considered, 75-97% of the mitochondrial proteins studied have been predicted to be imported into mitochondria. Moreover, the existence of mitochondrial- targeting sequences is predicted in 76-94% of the analyzed mitochondrial precursor proteins. As

a practical application, the number of unknown yeast open reading frames that might be mitochondrial proteins has been predicted, which revealed that many of them are clustered.

Even though remarkable recognition rates are obtained, MitoProt II is a system dealing only with mitochondrial proteins.

2.4.5 MTS

The mitochondrial targeting signal (MTS) is the presequence that directs nascent proteins bearing it to mitochondria. It was developed by Fujiwara et. al. [10]. They have developed a hidden Markov model (HMM) that represents various known sequence characteristics of MTSs, such as the length variation, amino acid composition, amphiphilicity, and consensus pattern around the cleavage site. The topology and parameters of this model are automatically determined by the iterative duplication method, in which a small fully-connected HMM is gradually expanded by state splitting. The model can be used to predict the existence of MTSs for given amino acid sequences. Its prediction accuracy was estimated to be 86.9% using the cross validation test. Furthermore, a higher correlation was observed between the HMM score and the in vitro ATPase activity of MSF, which can be regarded as an experimental measure of signal strength, for various synthetic peptides than was observed with other methods.

MTS introduces the usage of HMMs in subcellular localization. Its prediction accuracy is well enough for a single class predictor.

2.4.6 SignalP

SignalP which is developed by Nielsen et. al. [11], is a method for identification of signal peptides and their cleavage sites based on neural networks trained on separate sets of prokaryotic and eukaryotic sequences. It is a combined neural network approach to the recognition of signal peptides and their cleavage site and another network to distinguish between signal peptides and

non-signal peptides. The data used were taken from the SWISS PROT [6] and divided into either prokaryotic or eukaryotic. The recognition rate for cleavage site finding is 78% for eukaryotes and signal peptide discrimination is 96% for human proteins in the test phase.

The results of SignalP are satisfactory and it forms an infra structure for TargetP.

2.4.7 ChloroP

ChloroP [12] is a neural network based method for identifying chloroplast transit peptides and their cleavage sites. Using cross-validation, 88% of the sequences in our homology reduced training set were correctly classified as transit peptides or nontransit peptides. This performance level is well above that of the publicly available chloroplast localization predictor PSORT. Cleavage sites are predicted using a scoring matrix derived by an automatic motif-finding algorithm. Approximately 60% of the known cleavage sites in our sequence collection were predicted to within +/-2 residues from the cleavage sites given in SWISS PROT. An analysis of 715 Arabidopsis thaliana sequences from SWISS PROT suggests that the ChloroP method should be useful for the identification of putative transit peptides in genome-wide sequence data.

It produces cogent results on chloroplast transit enzymes for plants. It is the successor and plant version of SignalP.

2.4.8 SortPred

SortPred is a method developed by Fujiwara et. al. [13] using amino acid composition and order. The composition represents the global features, e.g. the amino acid composition in the full or partial sequences, while the order represents, e.g. the amino acid sequence order. The former was represented by neural networks and the latter was represented by a hidden Markov model. This method predicted the signal peptides, the mitochondrial targeting peptides and the chloroplast transit peptides, and the nuclear or cytosolic se-

quences. Its prediction accuracy is 91% for non-plant aand 86% for plant cells.

It uses both local and global features. SortPred improves the performance of TargetP by adding composition data. Its prediction accuracy is higher than others.

2.5 Related Machine Learning Techniques

A combination of clustering followed by classification seems to be the crucial aspect of this study. As mentioned before, self organizing maps are used for clustering and multi layer perceptrons are used for classification purposes. In this subsection, these techniques are elucidated.

2.5.1 Self Organizing Maps

It has been established that the brain forms topologically correct mappings of sensory experience, including connections from eye, ear, and skin to the cortex, and connections between different areas of the cortex [14]. Presence of such spatial maps for features has strong implications on how symbolic representations for concepts can automatically be formed. Note that, the models developed to explain the dynamics of these maps are not confined within the domain of biology; they also find use in practical applications where a compact and relevant representation of signals is desired.

The neurons are generally modeled as simple units having weighted incoming connections from input signals. The activation of a unit is determined by the sum of its weighted inputs. Connections with positive weights are called excitatory, and those with negative weights are called inhibitory. Once the neurons are modeled as such, the formation of topographic mappings becomes equivalent to finding the correct connection weights such that nearby input patterns activate nearby units. A network of units that perform such a mapping is called a *feature map* which is illustrated in Figure 2.3

A self organization of a feature map is an unsupervised learning, where input patterns are presented to the network and the weights of the units are

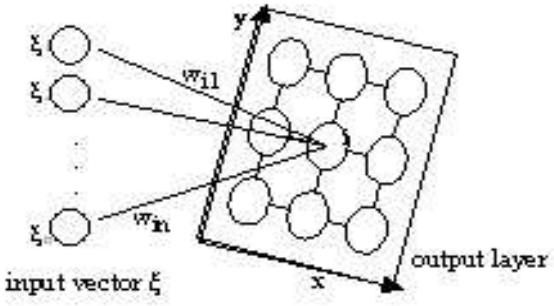

Figure 2.3: Architecture of a fully connected feature map.

updated properly to bring the network toward the desired result. Technically, we desire a feature map to have two properties: the weights of the output units should represent a clustering of input patterns, and the neighborhood of these units should convey information about the relationship of the corresponding clusters. The first of these properties can be achieved by ordinary competitive learning mechanism, where the weights of a winner unit (one that has highest activation) is made more sensitive to the presented pattern [15]. For the second property, there are two alternatives: using appropriate lateral connections between output layer units, or imposing the neighborhood relation algorithmically during the weight-update.

The alternative for producing topographically correct maps is to change the learning algorithm such that the neighborhood relation between units is preserved via an enforced correlation in their weight-updates. Kohonen's Self Organizing Map (SOM) uses competitive learning scheme with such neighborhood enforcement [16].

Each time a pattern ζ^{μ} is presented to the SOM, the unit i^{*} whose weight vector is most similar to the input vector, is selected. The distance measure of

choice effective in this selection is usually the Euclidean:

$$\arg\min_i \|w_i - \zeta^\mu\| \tag{2.1}$$

Once the winning unit is determined, the weights of all units are updated according to the learning rule:

$$\Delta w_{ij} = \eta \Lambda(i, i^*) \|\zeta_j^\mu - w_{ij}\| \tag{2.2}$$

where η is the *gain* parameter ($0 \leq \eta \leq 1$) affecting how much the weights are changed during each update. The neighborhood function $\Lambda(i, i^*)$ provides the desired proximity relation between units; it is chosen such that its value is 1 for i=i^* and falls off with distance between units i and i^*. A typical choice for $\Lambda(i, i^*)$ is the Gaussian function:

$$\Lambda(i, i^*) = \exp(\frac{-\|r_i - r_{i^*}\|^2}{2\sigma^2}) \tag{2.3}$$

where σ is the width parameter.

In order to examine the convergence of the self organizing process, we need a measure to determine the quality of the map produced after each iteration. Ritter and Schulten [17] define the following cost function for each weight:

$$E = \frac{1}{2} \sum_{i,\mu} \Lambda(i, i^*) \|\zeta_j^\mu - w_{ij}\|^2 \tag{2.4}$$

Gradient descent on this cost function yields:

$$-\eta \frac{\partial E}{\partial w_{ij}} = \eta \sum_\mu \Lambda(i, i^*) \|\zeta_j^\mu - w_{ij}\| \tag{2.5}$$

which is the sum of the Kohonen's learning rule over all patterns.

This implies that for sufficiently small η, the Kohonen rule decreases the cost until it reaches a local minimum. The gradient descent averages over all patterns, whereas learning rule updates the weights for each presentation of an input pattern. Thus, the actual behavior may diverge from this theoretical analysis.

Kohonen's simple and compact model finds use in a wide range of application domains, including pattern recognition tasks, image processing, decision making, optimization, and data compression. The simplicity of the model also suggests that the nature would make use of such mechanisms somewhere. Kohonen describes biological mechanisms that can give rise to topological maps of the somato-sensory input on the cortex. He suggests that the on-center, off-surround organization seen in some of the brain regions would accomplish similar mappings as that of SOM. However, it is also pointed out that the softwiring done during the training would not be sufficient to explain the biological maps. In addition to synaptic efficacy adjustments through excitatory and inhibitory connections between neurons, current theories use also the chemoaffinity, where chemical cues help define the target sites of the growing axons.

2.5.2 Multi Layer Perceptrons

Inspired by the biological learning systems, Multi Layer Perceptrons are built out of an interconnected set of simple units, where each unit takes a number of real-valued input and generates a single real-valued output, according to an activation function applied to the sum of the inputs. This aggregate structure provides a generic tool for function approximation. It has been proven that any continuous function from input to output can be implemented in a three-layer network of such units, using sufficient number of hidden units and proper nonlinear activation functions [18].

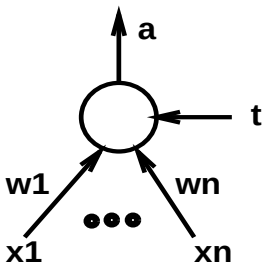

Figure 2.4: A Perceptron

The single unit of the neural network (also called a neuron, or a perceptron) is just a function applied to the weighted sum of inputs. The perceptron shown in Figure 2.4, implements the function:

$$y = f(\sum_{i=0}^{n} w_i x_i) \tag{2.6}$$

where f is the activation function of choice, and w_0 is the constant threshold value. The functions that can be approximated by a single perceptron are limited to linear functions. In pattern classification terms, a perceptron can correctly classify only linearly separable patterns. A generic three-layer network structure composed of these single units, which can approximate any continuous function, is shown in Figure 2.5.

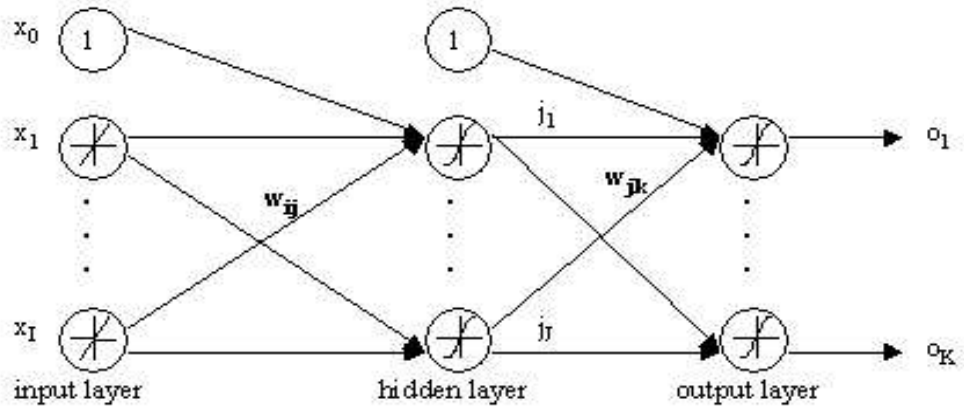

Figure 2.5: A 3-layer neural network

Even when we assume that a specific network structure and the activation functions for each unit are given, the expressive power of the neural network would be meaningless, unless can figure out the correct weights for the connections. Fortunately, we have an algorithm called back-propagation that allows the network to learn the weights [19].

The learning in the neural network works by back-propagating the error that occur at the output units. At each step, an input is presented to the network and the output is compared to the correct target value. The error made

by each unit is calculated in accordance with its share of blame. The weights of the units are then readjusted so as to minimize the error they have made. For a concise analysis, let us assume that the error is defined as the sum of square errors over all output units:

$$E_d = \frac{1}{2}\sum_k (t_k - o_k)^2 \tag{2.7}$$

And assume that hyperbolic tangent is used as the activation function in each of the network units.

$y = \tanh(x)$ is a squashing function that maps its input to the range $[-1, +1]$. It also has an easily computed derivative 1 - y^2, which makes it a good candidate for a learning neural network where the derivative needs to be calculated as we shall demonstrate.

In general, for each training example p, each weight w_{ij} is updated by adding Δw_{ij} :

$$\Delta w_{ij} = -\eta \frac{\partial E_p}{\partial w_{ij}} \tag{2.8}$$

where E_d is the error made on training pattern p, and η is the update rate (learning rate). The weight-update term takes each weight downward along its error-curve. Notice that the weight w_{ij} can influence the output only through net_j. Therefore, by using the chain rule, we obtain

$$\frac{\partial E_p}{\partial w_{jk}} = \frac{\partial E_p}{\partial net_k}\frac{\partial net_k}{\partial w_{jk}} = \frac{\partial E_p}{\partial net_k} x_{jk} \tag{2.9}$$

For an *output unit*, the following reductions can be made

$$\delta_k = \frac{\partial E_p}{\partial net_k} = \frac{\partial E_p}{\partial o_{kj}}\frac{\partial o_{kj}}{\partial net_{kj}} \tag{2.10}$$

$$\frac{\partial E_p}{\partial net_k} = \frac{\partial}{\partial o_k}\frac{1}{2}\sum_{outputs}(t_s - o_s)^2 = \frac{\partial}{\partial o_k}\frac{1}{2}(t_k - o_k)^2 = (o_k - t_k) \tag{2.11}$$

$$\frac{\partial o_k}{\partial net_k} = \frac{\partial f(net_k)}{\partial net_k} = \frac{\partial \tanh(net_k)}{\partial net_k} = 1 - o_k^2 \tag{2.12}$$

Note that the derivatives in the summation of Equation (2.11) is zero for all output units except when k=j. Substituting values found in (2.11) and (2.12)

into (2.10), we obtain

$$\delta_k = \frac{\partial E_p}{\partial net_k} = (o_k - t_k)1 - o_k^2 \tag{2.13}$$

for output unit k

$$\Delta w_{jk} = \eta(t_k - o_k)(1 - o_k^2)x_{jk} \tag{2.14}$$

When deriving the weight update rule Δw_{ij} for the *hidden* units, we need to take into account the indirect ways w_{ij} influences the error term through each of the output units. Therefore, we can do the following reductions

$$\delta_j = \frac{\partial E_p}{\partial net_j} = \sum_{outputs} \frac{\partial E_p}{\partial net_k} \frac{\partial net_k}{\partial o_j} \frac{\partial o_j}{\partial net_j} =$$

$$\sum_{outputs} \frac{\partial E_p}{\partial net_k} w_{jk}(1 - o_j^2) = (1 - o_j^2) \sum_{outputs} \delta_k w_{jk} \tag{2.15}$$

and we obtain the learning rule for a *hidden* unit i:

$$\Delta w_{ij} = \eta(1 - o_j^2)x_{ij} \sum_{outputs} \delta_k w_{jk} \tag{2.16}$$

Establishing the learning rules is important but it does not guarantee a solution to be found. The network weight vectors reachable by the gradient procedure outlined by the above equations may not include all possible weight vectors. In particular, the network weights may get stuck at a local minima in the error space, and never find the optimum solution. The problems in a learning net is not limited to local minima. In general there are no prescribed learning parameters, such as learning rate, momentum, and number of hidden units which are challenging design choices.

CHAPTER 3

METHODS AND EXPERIMENTAL RESULTS

In this chapter, data representation and the generation of the data sets are discussed in Section 3.1 and Section 3.2, respectively. In Section 3.3, clustering methods and in Section 3.4, classification properties are explained. The results of the experiments are explained in Section 3.5.

3.1 Data Representation

Protein sequences are strings of arbitrary size and amino acids correspond to the letters in a protein sequence. Let $\hat{X}$ represent a protein sequence whose length is $len(\hat{X})$. $\hat{X}$ can be decomposed into substrings of some fixed length, κ. If $\kappa < len(\hat{X})$, there are exactly $(len(\hat{X}) - \kappa + 1)$ substrings in $\hat{X}$. $\hat{X}(j : \kappa + j)$ then denotes j^{th} substring in a protein sequence $\hat{X}$. The following example illustrates the substrings well.

Example 1 *The amino acid sequence of a mitochondrial protein starts and ends as follows :*

MQTHVRRVALQALRP . . . DHDEDATPAE. The actual length of the sequence is 101. First three substrings are as follows for κ=10.

"MQTHVRRVAL"

"QTHVRRVALQ"

"THVRRVALQA"

In order to perform further computational analysis, we need to encode the amino acids. Although, the most popular way of encoding reported in the literature is to represent each amino acid in binary form, in this study, we make use of substitution matrices. While aligning two protein sequences, certain methods are used to score the alignment of one residue against another. Substitution matrices indicate score values for this purpose. We employ *PAM250* scoring matrix to encode an amino acid. In the rest of the thesis, X denotes a *PAM* encoded protein sequence $\hat{X}$.

Example 2 *Assuming that our κ=3 on the same protein in the previous example, let us encode the first substring ("MQT") with PAM250 scoring matrix. Three amino acids are represented by as follows:*
M = [-1 0 -2 -3 -5 -1 -2 -3 -2 2 4 0 6 0 -2 2 -1 -4 -2 2]
Q = [0 1 1 2 -5 4 2 1 3 -2 -2 1 -1 -5 0 -1 -1 -5 -4 2]
T = [1 -1 0 0 -2 -1 0 0 -1 0 -2 0 -1 -2 0 1 3 -5 -3 0].
Hence the whole substring is encoded as:
"MQT" = [-1 0 -2 -3 -5 -1 -2 -3 -2 2 4 0 6 0 -2 2 -1 -4 -2 2 0 1 1 2 -5 4 2 1 3 -2 -2 1 -1 -5 0 -1 -1 -5 -4 2 1 -1 0 0 -2 -1 0 0 -1 0 -2 0 -1 -2 0 1 3 -5 -3 0]. Remark that since PAM250 is a 20×20 matrix, the length of the encoded vector is $20 \cdot \kappa$

3.2 Generation of the Data Sets

As described before, all sequences are extracted from SWISS PROT [6] and inappropriate sequences are removed before redundancy reduction, which was undertaken to avoid problems related to redundant data during neural network training and testing. To increase the size of the data sets as far as possible, also sequences annotated as "POTENTIAL", "BY SIMILARITY" or "PROBABLE" are included in their respective sets. These sequences lack experimental evidence.

Sequences are extracted by requiring the keyword EUKARYOTA in the OC (Organism Classification) field. Targeting peptide entries marked as POTENTIAL, BY SIMILARITY or PROBABLE in their FT field, but still with an explicitly annotated endpoint of the presequence, are also included in their respective sets (except SP which was considered large enough without including such sequences). In the nuclear and cytosolic sets, sequences with any of these annotations as to their subcellular location annotations in their CC field are also accepted. Only sequences with an N-terminal 'Met' residue are considered, and the very few sequences containing 'B', 'Z' or 'X' are excluded, in order to avoid possible noise from the ambiguous positions in the training. Following the removal of these and other inadequate entries sequences with a high degree of similarity to other sequences are removed by redundancy reduction.

In each experiment, 100 proteins from every class yielding total 400 proteins are taken from the whole database consisting of 715 SPs, 438 CPs, 371 MPs and 1214 NPs sequences. Each protein in a class is indexed by an integer starting from 1 up to cardinality of total number of proteins in that class. A random generator produces 100 different numbers between the bounds of index the class. If the index of the protein is one of such 100 numbers then the protein is included to the data set for the class. 320 proteins where 80 per class are used in the training and the rest 80 where 20 per class are used for testing.

Proteins are in "Fasta" format and they are parsed before clustering. Substrings of size κ=30 are extracted from the protein sequences.

Example 3 *A protein belonging to the class SP in "Fasta" format. Initial line starting with > gives the identification number (P20334) in SWISS PROT and the amino acid seqeunce specific to the protein.*

>P20334; 24 4-1BB LIGAND RECEPTOR PRECURSOR (T-CELL ANTIGEN 4-1BB).

MGNNCYNVVVIVLLLVGCEKVGAVQNSCDNCQPGTFCRKYNPVCKSCPPS

TFSSIGGQPNCNICRVCAGYFRFKKFCSSTHNAECECIEGFHCLGPQCTR
CEKDCRPGQELTKQGCKTCSLGTFNDQNGTGVCRPWTNCSLDGRSVLKTG
TTEKDVVCGPPVVSFSPSTTISVTPEGGPGGHSLQVLTLFLALTSALLLA
LIFITLLFSVLKWIRKKFPHIFKQPFKKTTGAAQEEDACSCRCPQEEEGG
GGGYEL

In the Table 3.1, the minimum, maximum and average length of the substrings per class is depicted. Our κ value, which is 30, extracts enough features for each class.

Table 3.1: The minimum, maximum and average length of the substrings per class.

Class	*Minimum*	*Maximum*	*Average*
SP	57	4548	434
CP	15	4725	582
MP	47	1500	408
NP	39	3759	595

The SWISS PROT database is parsed by the help of Swiss-Knife [20]. Swiss-Knife, an object-oriented Perl library to read, manipulate and write records in SWISS PROT flat file format. By the help of it, entries are converted into 'Fasta' format which is more suitable for computation.

3.3 Clustering

In our system, clustering occurs during training and in this phase substrings are topologically grouped. The problem of finding motifs of a protein class turns out to finding the nodes specific to a class. At the end of training, the winning cell for each substring is determined. If the dimension of our SOM is $m \times n$, any SOM cell (i,j) can be represented as an integer via the map $f(i,j) = i \cdot m + j$. Hence, for each protein $\hat{X}$, a vector $\hat{v}$ can be associated whose length is $len(m \cdot n)$. All the entries of $\hat{v}$ are initialized to 0. For each substring with the winning cell w=(k,l) belonging to it, $\hat{v}[\mathrm{f(w)}]$ is incremented. When all

the substrings are traversed, each entry of $\hat{v}$ is normalized by dividing with $(len(\hat{X}) - \kappa + 1)$ which is the cardinality of substrings in $\hat{X}$. Let $\hat{V}$ represents the set of vectors (i.e., $\hat{v}$ per protein) for all proteins. This gives the distribution of SOM cells over the proteins. The set, $\hat{V}$, gives the data needed for classification.

3.3.1 Parameters of SOM

The following items should occur when defining a SOM:

- Dimensionality of the feature vectors.
- Topology type.
- Map dimension in x-direction.
- Map dimension in y-direction.
- Neighborhood type.

The map topology can be either rectangular or hexagonal.

Table 3.2: Parameters for SOM.

Parameter	*Value*
Max. Iterations	3000
Dimensionality of the feature vectors	600
Topology type	Rectangular
Map dimension in x-direction	25
Map dimension in y-direction	25
Neighborhood type	Gaussian
Training Patterns	320
Test Patterns	80

The neighborhood type is either “bubble” or “Gaussian”. This shows the kernel function for update of the reference vectors.

The parameters used in our experiments are in Table 3.2

3.3.2 Map File Format

The x-coordinates of the map (column numbers) may be thought to range from 0 to n-1, where n is the x-dimension of the map, and the y-coordinates (row numbers) from 0 to m-1, respectively, where m is the y-dimension of the map. The reference vectors of the map are stored in the map file in the following order:

```
1       The unit with coordinates (0, 0).

2       The unit with coordinates (1, 0).
         ...
n       The unit with coordinates (n-1, 0).

n+1     The unit with coordinates (0, 1).
         ...

n.m      The last unit is the one with coordinates
(n-1, m-1).
```

The distance between two units in the map is computed as an Euclidean distance in the (two dimensional) map topology.

The reference vectors of the map are first initialized to tentative values. The lattice type of the map and the neighborhood function used in the training procedures are also defined in the initialization. The map is initialized using random numbers.

Some local parameters should be set before training. These are:

- learning Rate (α) which decreases to 0 during training,
- radius (r) which decreases linearly to 1 during training,
- number of training steps (N)

Training is done in two phases. The first of them is the ordering phase during which the reference vectors of the map units are ordered. During the second phase the values of the reference vectors are fine-tuned.

In ordering, the neighborhood radius is taken almost equal to the diameter of the map and decreases to one during training, while the learning rate decreases to zero.

Algorithm 1 Ordering Algorithm in Training

for i=1 **to** 10 **do**
  for *class* $\in \{SP, CP, MP, NP\}$ **do**
    Train with $\langle\ alpha$ = 0.05, r = 10, N = 5000 $\rangle$
  end for
end for

For each class, 50000 epochs are completed during fine tuning in total.

During the second phase the reference vectors in each unit converge to their correct values. The second phase is usually longer than the first one. The learning rate is thereby smaller. The neighborhood radius is also smaller on the average: in the beginning the units up to a distance of three are covered. In our experiments, the training time of the second phase is four times longer than in the first phase.

Algorithm 2 Fine Tuning Algorithm in Training

for i=1 **to** 10 **do**
  for *class* $\in \{SP, CP, MP, NP\}$ **do**
    Train with $\langle\ alpha$ = 0.02, r = 3, N = 20000 $\rangle$
  end for
end for

For each class, 200000 epochs are completed during fine tuning in total.

After these steps of training, the map is ready to be tested.

Recall that X denotes a *PAM* encoded protein sequence . Thus S denotes a *PAM* encoded protein substring of length κ=30. The method explained is depicted in Algorithm 3.

Algorithm 3 Algorithm for forming the data for classification

for *class* $\in \{SP, CP, MP, NP\}$ **do**
  for each protein sequence X **do**
    initialize each entry of $\hat{v}_X$ **to** 0
      for each encoded substring S **do**
        Find the winning cell **k** among the SOM cells
        increment $\hat{v}_X(k)$ by 1
      end for
    for each component i of $\hat{v}_X$ **do**
      Divide $\hat{v}_X(i)$ by $(len(X) - \kappa + 1)$
    end for
  end for
end for

For each protein sequence, a vector $\hat{v}_X$ is assigned and the corresponding components of the winning cells are increased by 1 in $\hat{v}_X$. The winning cells are found by the competitive learning algorithm. At the end, all the vectors are normalized by dividing the total number of substrings.

3.4 Classification

Clustering phase prepares the labeled data for supervised learning. As mentioned before, MLPs are used for classification. The properties of the neural network used in the experiments are described in this Section.

The back-propagation algorithm [21] on MLPs learns the weights for a multi layer network, given a network with a fixed set of units and inter connections. It employs gradient descent to attempt to minimize the squared

error between the network output values and the target values for those outputs.

The learning task here involves classifying the set $\hat{V}$ whose labels are already known. $\hat{V}$ is a partitioned into mutually exclusive two subsets used for training and testing. The proportion between the cardinalities of the subsets is $\frac{1}{4}$ on behalf of the training set.

In applying back-propagation to a given task, a number of design choices must be made. The design described here learns the target function quite well.

3.4.1 Input Encoding

Given that the MLP input is to be some representation, one design choice how to encode it. Since all the sequences have different lengths, no brute force encoding can be applied since the number of input neurons must be some fixed value. It is known that all the elements in $\hat{V}$ have the length $m{\cdot}n$ which automatically forms the input encoding. All the values in a vector $\hat{v}$ are real numbers between 0 and 1 because it is a distribution of SOM cells. Each input represents a protein sequence.

3.4.2 Output Encoding

The MLP must output one of four values indicating the subcellular localization. We could encode this four way classification using a single output unit, assigning outputs of, say, 0.2, 0.4, 0.6 and 0.8 to encode these four possible values. Instead, we use four distinct output units, each representing one of four possible localization sites, with the highest values output taken as the network prediction. This is often called a *1-of-n* output encoding. There are two motivations for choosing the 1-of-n output encoding over the single unit option. First, it provides more degrees of freedom to the network for representing the target function (i.e., there are n times as many weights available in the output layer of units). Second, in the 1-of-n encoding the difference

between the highest valued output and the next highest one can be used as a measure of confidence in the network prediction. A further design choice is determining the target values for these four output units. One obvious choice would be use the four target values (1,0,0,0) to encode the class SP, (0,1,0,0) to encode the class MP, etc. Instead of 0 and 1, 0.1 and 0.9 is used so that (0.9,0.1,0.1,0.1) is the target output vector for SP. The reason for avoiding target values 0 and 1 is that sigmoid units cannot produce these output values given finite weights. If we attempt to train the network to fit target values of exactly 0 and 1, gradient descent will force the weigths to grow without bound. On the contrary, values of 0.1 and 0.9 are achievable using a sigmoid unit with finite weights.

3.4.3 Network Graph Structure

The back-propagation algorithm can be applied to any acyclic directed graph of sigmoid units. Therefore, another design choice is how many units to include the network and how to interconnect them. A layered network with feed-forward connections from every unit in one layer unit to every unit in the next. In our design, this standard structure, using two layers of sigmoid units (one hidden layer and one output layer) is chosen but the alternatives are also tried. It is common to use one or two hidden layers, elsewhere training time can be too long. A different number of neurons is experimented yet in many applications it has been found that some minimum number of hidden units is required in order to learn the target function accurately. Recall that our SOM has size 25×25 yielding a 625-dimensional feature vector so there must be 625 neurons in the input layer. Using 10 neurons in hidden layer yield promising results and extra hidden units above this number do not dramatically affect the generalization accuracy. Increasing the number of hidden units often increases the tendency to overfit the training data, thereby reducing the accuracy. In summary, a $\langle\, 625-10-4 \,\rangle$ feed-forward multi layer back-propagation neural network is used for classification.

Table 3.3: Parameters for MLP.

Parameter	*Value*
Maximum Iterations	3000
Learn Rate Start Control Iteration	10
Learn Rate Minimum	0.01
Learn Rate Minimum	0.3
Momentum	0.8
Tolerance	0.4
Quit at Training RMS Error	0.02
Transfer Functions	Sigmoid
Training Patterns	320
Test Patterns	80

3.4.4 Other Learning Algorithm Parameters

The learning rate η was set to 0.01, and the momentum α was set to 0.8. Lower values for both parameters produced roughly equivalent generalization accuracy, but longer training times. If these values are set too high, training fails to converge to a network with acceptable error over the training set. Network weights in the output units are initialized to small random values. However, input unit weights are initialized to 0, because it yields much more intelligible results of the learned weights, without any noticeable impact on generalization accuracy. After every 5 gradient descent steps the performance of the network was evaluated over the test set. The number of maximum iterations is 3000. There is a variable learning rate changes between 0.01 and 0.3, The tolerance value for the winning output is 0.4 which means that the weight of the winning output neuron should be in the interval $[0.5, 1.3]$.

Training parameters are summarized in the Table 3.3.

3.5 Results of the Experiments

3.5.1 Cross Validation

Cross validation is a model evaluation method that is better than residuals. The problem with residual evaluations is that they do not give an indication

of how well the learner will do when it is asked to make new predictions for data it has not already seen. One way to overcome this problem is to not use the entire data set when training a learner. Some of the data is removed before training begins. Then when training is done, the data that was removed can be used to test the performance of the learned model on new data. This is the basic idea for a whole class of model evaluation methods called **cross validation**.

The **holdout** method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The predictor fits a function using the training set only. Then the predictor is asked to predict the output values for the data in the testing set (it has never seen these output values before). The errors it makes are accumulated as before to give the mean absolute test set error, which is used to evaluate the model. The advantage of this method is that it is usually preferable to the residual method and takes no longer to compute. However, its evaluation can have a high variance. The evaluation may depend heavily on which data points end up in the training set and which end up in the test set, and thus the evaluation may be significantly different depending on how the division is made.

k**-fold cross validation** is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. The variance of the resulting estimate is reduced as k is increased. The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation. A variant of this method is to randomly divide the data into a test and training set k different

times. The advantage of doing this is that you can independently choose how large each test set is and how many trials you average over.

3.5.2 Measuring Prediction Performance

5-fold cross validation method is performed to estimate the prediction accuracy of our method: the data are randomly divided into five, and the four-fifth is used for training while one-fifth is used for testing. The classifier is trained 5 times, each time with a different set held out as a validation set. The estimated performance is their mean. A performance test can produce two kinds of errors: a **false positive** result or a **false negative** result. In general, there are four groups when evaluating the accuracy of a test in a statistical manner.

- True Positives(tp): those which test positive for a class and are positive
- False Positives(fp): those which test positive, but are negative
- True Negatives(tn): those which test negative and are negative
- False Negatives(fn): those which test negative, but are positive

For instance, if a SP protein is classified as MP, then it is treated as a false negative for MP class and a false positive for SP class.

Performances are in general measured as percentage correctly predicted sequences, and as sensitivity (fraction of positive examples predicted as positives):

$$sens = \frac{tp}{tp + fn} \tag{3.1}$$

and specificity (fraction of all positive predictions that are true positives):

$$spec = \frac{tp}{tp + fp} \tag{3.2}$$

The Matthews correlation coefficient, MCC [22], defined as:

$$MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fn)(tp + fp)(tn + fp)(tn + fn)}} \tag{3.3}$$

are used in the comparison of performance of different predictors. MCC equals 1 for a perfect prediction while it is 0 for a completely random assignment. The accuracy field shows how many proteins are correctly predicted in the test set for all classes. Sensitivity is a measure of the probability of correctly diagnosing/classifying a case or event (i.e. true positive rate). Specificity is a measure of the probability of correctly identifying/classifying a non-case or non-event (i.e. true negative rate). There are no entries for true negatives for specificity and sensitivity. MCC uses all the possible groups and it is more deterministic criteria comparing with the others.

3.5.3 Results

The results of 5-fold cross validation is given in the Table 3.4. The accuracy of predictions change between $[88.8\%, 93.7\%]$. For specificity, 0.76 is the minimum value attained in the third experiment for CP, and 1 is the maximum value for SP in both fourth and fifth experiments. 1 is obtained for MP and SP classes in the second and fifth experiments, respectively. The value 1 for SP in the last experiment shows that it is a perfect prediction for SP class. The values for three criteria is almost higher than others for all classes in the second experiment so the prediction accuracy is higher than the others as well.

Overall prediction performance is calculated by calculating the mean of the each test which is depicted in the Table 3.5. In SP class, both the specificity and sensitivity values are higher than the other classes. The best results are attained in SP, then in MP, then in MP and finally CP.

The results of P2SL compared with non-plant versions of SortPred, TargetP and iPSORT are summarized in Table 3.6. The bold value corresponds to the highest score among four methods. P2SL prediction accuracy is slightly higher than the other methods. SortPred, TargetP and iPSORT classify only 3 classes “SP”, “MP” and “Other”. Since some of the entries in the class “Other”

Table 3.4: Prediction performance of P2SL on 5-fold cross validation test.

Experiment	*Accuracy*	*Category*	*Specificity*	*Sensitivity*	*MCC*
1	88.8%	SP	0.96	0.96	0.95
		CP	0.83	0.83	0.82
		MP	0.83	0.88	0.88
		NP	0.88	0.88	0.87
2	93.7%	SP	0.94	0.94	0.93
		CP	0.95	0.90	0.92
		MP	0.96	1.00	0.98
		NP	0.89	0.89	0.89
3	87.5%	SP	0.83	0.88	0.85
		CP	0.76	0.81	0.78
		MP	0.92	0.85	0.88
		NP	0.95	0.95	0.95
4	88.8%	SP	1.00	0.95	0.97
		CP	0.89	0.81	0.84
		MP	0.85	0.88	0.86
		NP	0.80	0.92	0.85
5	90.0%	SP	1.00	1.00	1.00
		CP	0.92	0.69	0.79
		MP	0.86	0.95	0.90
		NP	0.84	0.91	0.87

are different than "NP" or "CP", only the classes "SP" and "MP" are taken into the comparison. P2SL scores are better than others except the sensitivity in "SP" class.

The distribution of the false negatives is given in Table 3.7. In each experiment, for each column, the type of misclassified proteins are indicated. The most dense column belongs to the class CP whose sensitivity is the lowest among others. There are no entries in the cell $(2, MP)$ and $(5, SP)$ so the sensitivity values are 1. There are mostly MP entries in SP class and mostly CP entries in MP class.

Similarly, the distribution of the false positives is given in Table 3.8. No MP entry occurs as a positive in SP class. Remark that, there are no false positives in SP for test 4 and 5 so specificity for them is 1. Moreover, the results

Table 3.5: Overall prediction performance of P2SL.

Category	*Specificity*	*Sensitivity*	*MCC*
SP	0.95	0.95	0.93
CP	0.87	0.81	0.83
MP	0.88	0.91	0.90
NP	0.87	0.91	0.89

Table 3.6: Results of P2SL compared with its competants.

Prediction Program	*Category*	*Specificity*	*Sensitivity*	*MCC*
P2SL	SP	**0.95**	0.95	**0.93**
	MP	**0.88**	**0.91**	**0.90**
SortPred	SP	0.86	0.95	0.87
	MP	0.79	0.88	0.80
TargetP	SP	0.92	**0.96**	0.92
	MP	0.67	0.89	0.73
iPSORT	SP	0.91	0.87	0.85
	MP	0.70	0.79	0.70

of SP class in test 5 is perfect since MCC is 1. Some rules can be extracted by considering these tables by analyzing the distibution of false values.

Table 3.7: Distribution of false negatives on 5-fold cross validation test.

Experiment	*SP*	*CP*	*MP*	*NP*
1	MP	NP NP SP MP	CP CP	CP CP
2	MP	NP MP		SP CP
3	CP MP	SP SP MP	NP CP CP CP	SP
4	MP	MP NP NP MP	CP NP CP	MP
5		MP NP NP MP NP	NP	MP CP

Table 3.8: Distribution of false positives on 5-fold cross validation test.

Experiment	*SP*	*CP*	*MP*	*NP*
1	CP	MP MP NP NP	SP CP	CP CP
2	NP	NP	CP	SP CP
3	CP CP NP	SP MP MP MP	SP CP	MP
4		MP MP	SP CP CP NP	CP CP MP
5		NP	CP CP NP	CP CP CP MP

CHAPTER 4

CONCLUSIONS

Subcellular localization is a key functional characteristic of proteins. A fully automatic and reliable prediction system is needed, especially for the analysis of large scale genome sequences.

This research is focused on prediction of protein subcellular localization in eukaryotic organisms using amino acid sequence. There are two type of features can be extracted from primary sequences. The composition of residues can be taken as global features while the order in partial subsequences can be taken as local features. In this work, we deal with local features and develop a new method, prediction of protein subcellular localization (P2SL), for the aim of classifying four different locations in a cell.

Our approach for prediction is to find the most frequent substrings for each protein (class) based on clustering and then to use these most frequent substrings as features for classification. This approach allows a classification independent of the length of the sequence. Another important property of the approach is to provide a means to perform reverse analysis and analysis to extract rules. In addition to these and more importantly, we describe the use of a new encoding scheme for the amino acids that conserves biological function based on *point of accepted mutations (PAM)* substitution matrix. If no method for searching the significant motifs is used, an exhaustive search

should be done on the set of all substrings to find out motifs. This scheme should need an extreme allocation of space and time.

The database is taken from a previous work [4]. Fixed length motifs are extracted from variable length strings, then encoded with a special matrix (PAM250) to form the feature vectors. This encoding scheme preserves the biological meaning of each amino acid found in protein subcellular targeting sequence motifs and never used before related studies.

Application of clustering is meaningful, since the labels of substrings are not known. Only the the class of primary sequences are known. Self organizing maps are used to cluster the feature vectors on a two-dimensional grid. The most significant motifs for each protein in a given class are determined and their normalized distribution constructs the features for classification. The length of the sequence does not affect the classifier by this technique.

Multi layer perceptrons is used for classification purposes as a supervised learning method. Adjusting suitable parameters to a three layer neural network which implements the back-propagation algorithm yields the recognition results.

Mutually exclusive training and test sets are formed. 5-fold cross validation is employed and the statistical measures like sensitivity, specificity and Matthews correlation coefficient, MCC, are obtained as usual in a pattern recognition study. P2SL achieved slightly better results than its competants. MCC of our study is 0.93 for SP, 0.83 for CP, 0.90 for MP and 0.89 for SP. The prediction rate changes from 87.5% to 93.7%.

P2SL method integrates self organizing maps for clustering, with multi layer perceptrons for classification. It is found out that the clustering followed by classification well represent the biological features of the seqeunces. By the usage of PAM250 matrix, the chemical differences of each amino acid in protein are symbolized and more meaningful feature vectors are obtained.

The motif size κ is problem-specific parameter and it should be well adjusted. Our κ value, which is 30, seems to extract sufficient features for each

class.

There are a number of design parameters used in the experiments. The most important ones are as follows:

- SOM size.
- SOM neighborhood function.
- SOM training radius.
- MLP topology.
- MLP tolerance value.
- MLP algorithm.

After intensive trials, almost optimal values for those parameters are determined. Any significant change in this crucial attributes may affect the recognition results.

The experimental results show that out overall prediction accuracy is remarkable especially for the class MP. There is a quite important difference between P2SL and its competants in the view of MP statistics. It is in fact as reliable as the systems that predicts only mitochondrial targeting peptides although there are three other classes in out study.

In our previous work [23], nearest neighborhood method is used as a classifier but its results are not satisfactory in quite large data sets and in cross validation experiments. This shows that the problem of determining subcellular location of proteins is not linearly separable. At that point, our strategy is changed upon using MLPs as the classification technique.

Since the aim of our clustering scheme is deriving the most significant motifs in a protein sequence for a class, it can provide a means to perform reverse analysis. Certain motifs can be determined for a given class if κ is small enough. This technique can be extended for large κ values as well by adding some "do not care" signs to certain position in a motif. Using the

information obtained from reverse analysis, some rules can be extracted. An expert system employing those rules can be designed.

More locations in a cell such as

- Lysosome,
- Plasma membrane,
- Vacuole,
- Golgi apparatus,
- Peroxisome.

can be added to P2SL, hence it can classify much more localization sites even though the latter are not as important as the ones used in the current version of our work.

A new data set can be formed consisting of only human proteins. All the studies up to now include some proteins that are belonging to yeast or bacteria. It may be a novel work designed for human only. For realizing this, SWISS PROT should be re-parsed to separate human genes.

Our ultimate goal is to combine global and local features in a single system. This can be a combination of a rule-based system for composition together with a neural network classifier for order of amino acid sequences. Outputs of the MLP can be evaluated in a rule based system with suitable modifications. Another idea is adding some more input neurons to MLP, reflecting the information obtained from global features.

A web site can be designed for the running of the system in a batch mode, passing through the steps explained in the study to help the ones working in this field.

REFERENCES

[1] A. Merino-Trigo R.D. Teasdale P.A. Gleeson C. van Vliet, E.C. Thomas. Intracellular sorting and transport of proteins. *Prog. Biophys. Mol. Biol.*, 83(1):1–45, 2003.

[2] J. Gouzy F. Corpet, F. Servant and D. Kahn. Prodom and prodom-cg: tools for protein domain analysis and whole genome comparisons. *Nucleic Acids Research*, 28:267–269, 2000.

[3] S.F. Altschul. Amino acid substitution matrices from an information theoretic perspective. *Lecture Notes on Computer Science*, pages 602–610, 2003.

[4] S. Brunak O. Emanuelsson, H. Nielsen and G. von Heijne. Predicting subcellular localization of proteins based on their n-terminal amino acid sequence. *Journal of Molecular Biology*, 300:1005–1016, 2000.

[5] R.M. Schwartz M.O. Dayhoff and B.C. Orcutt. A model of evolutionary change in proteins. *Atlas of protein sequence and structure. Vol. 5, Suppl. 3. National Biomedical Research Foundation, Washington, D.C.*, pages 345–352, 1979.

[6] A. Bairoch and R.Apweiler. The swiss-prot protein sequnce database and its supplement trembl in 2000. *Nucleic Acids Research*, 28:45–48, 2000.

[7] K. Nakai and M. Kanehisa. A knowledge base for predicting protein localization sites in the eukaryotic cells. *Genomics*, 14:897–991, 1992.

[8] iPSORT is available at: http://hypothesiscreator.net/iPSORT.

[9] M.G. Claros and P. Vincens. Computational method to predict mitochondrially imported proteins and their targeting seqeunces. *European Journal of Biochemisty*, 241:779–786, 1996.

[10] H. Asogawa Y. Fujiwara and K. Nakai. Prediction of mitochondrial targeting signals using hidden markov models. *Genome Informatics*, 8:53–60, 1997.

[11] S. Brunak G. von Heijne H. Nielsen, J. Engelbrecht. A neural network method for identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *International Journal of Neural Systems*, 8(5-6):581–599, 1997.

[12] H. Nielsen O. Emanuelsson and G. von Heijne. Chlorop, a neural network-based method for predicting chloroplast transit peptides and their cleavage sites. *Protein Science*, 8:978–984, 1999.

[13] Y. Fujiwara and M. Asogawa. Prediction of subcellular localization using amino acid composition and order. *Genome Informatics*, 12:103–112, 2001.

[14] S.du Lac E.I.Knudsen and S.D.Esterly. Computational maps in the brain. *Annual Review of Neuroscience*, 10:41–65, 1987.

[15] D.E. Rumelhart and D. Zipser. Feature discovery by competitive learning. *Cognitive Science*, 9:75–112, 1985.

[16] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[17] H. Ritter and K. Schulten. Kohonen's self-organizing maps: Exploring their computational capabilities. *IEEE International Conference on Neural Network (San Diego 1988)*, 1:109–116, 1988.

[18] A.N. Kolmogorov. On the representation of continous functions of several variables by superposition of continuous functions of one variable and addition. *Doklady Akademiia Nauk SSSR*, 14(5):953–956, 1957.

[19] D.E. Rumelhart and J.L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. MIT Press, Cambridge, MA, and London, England, 1986.

[20] The Swiss-Knife is available at: http://swissknife.sourceforge.net.

[21] T. Mitchell. *Machine Learning*. McGraw-Hill Series in Computer Science, 1997.

[22] B.W. Matthews. Comparison of predicted and observed secondary structure, of t4 phage lysozyme. *Biochim. Biophys. Acta*, 405:442–451, 1975.

[23] V. Atalay M. Ozarar and R. Atalay. Prediciton of protein subcellular localization based on primary seqeunce data. *Lecture Notes on Computer Sciences on the Proceedings of ISCIS'O3*, pages 602–610, 2003.

[24] The SOMPAK package is available at: http://www.cis.hut.fi/nnrc/papers.

APPENDIX A

Format of a Sequence Entry in SWISS PROT

The entries in the SWISS PROT database are structured so as to be usable by human readers as well as by computer programs. The explanations, descriptions, classifications and other comments are in ordinary English. Wherever possible, symbols familiar to biochemists, protein chemists and molecular biologists are used.

Each sequence entry is composed of lines. Different types of lines, each with their own format, are used to record the various data that make up the entry. A sample sequence entry is shown below.

Example 4 *An entry in SWISS PROT database that belongs to the class "SP":*

```
ID   TNR9_MOUSE     STANDARD;      PRT;   256 AA.

AC P20334;

DT   01-FEB-1991 (Rel. 17, Created)

DT   01-FEB-1991 (Rel. 17, Created)

DT   15-SEP-2003 (Rel. 42, Last annotation update)
```

DE Tumor necrosis factor receptor superfamily member 9 precursor

DE ligand receptor) (T-cell antigen 4-1BB) (CD137 antigen).

GN TNFRSF9 OR ILA OR LY63 OR CD137 OR CD157.

OS Mus musculus (Mouse).

OC Eukaryota; Metazoa; Chordata; Craniata; Vertabrata; Euteleostomi;

OC Mammalia; Eutheria; Rodentia; Sciurognathi; Muridae; Murinae; Mus.

OX NCBI_TaxID=10090;

RN [1]

RP SEQUENCE FROM N.A.

RX MEDLINE=89184547; PubMed=2784565;

RA Kwon B.S., Weissman S.M.;

RT "cDNA sequences of two inducible T-cell genes.";

RL Proc. Natl. Acad. Sci. U.S.A. 86:1963-1967(1989).

RN [2]

RP SEQUENCE FROM N.A.

RC STRAIN=BALB/c;

RX MEDLINE=94179805; PubMed=8133039;

RA Kwon B.S., Kozak C.A., Kim K.K., Pickard R.T.;

RT "Genomic organization and chromosomal localization of the T-cell

RT antigen 4-1BB.";

RL J. Immunol. 152:2256-2262(1994).

RN [3]

RP CHARACTERIZATION, AND SEQUENCE OF 25-29.

RX MEDLINE=93139510; PubMed=7678621;

RA Pollok K.E., Kim Y.-J., Zhou Z., Hurtado J., Kin K.K., Pickard R.T.,

RA Kwon B.S.;

RT "Inducible T cell antigen 4-1BB. Analysis of expression and function.";

RL J. Immunol. 150:771-781(1993).

CC -!- FUNCTION: Receptor for TNFSF14/4-1BBL. Possibly active during T

CC cell activation.

CC -!- SUBUNIT: PRINCIPALLY AN HOMODIMER, BUT ALSO FOUND AS A MONOMER.

CC ASSOCIATES WITH P56-LCK. Interacts with TRAF1, TRAF2 AND TRAF3 (By

CC similarity).

CC -!- SUBCELLULAR LOCATION: Type I membrane protein.

CC -!- TISSUE SPECIFICITY: Expressed on the surface of activated T cells.

CC -!- INDUCTION: Optimal by PMA and ionomycin.

CC -!- SIMILARITY: Contains 4 TNFR-Cys repeats.

DR EMBL; J04492; AAA40167.1; -.

DR EMBL; U02567; AAA93113.1; -.

DR PIR; B32393; B32393.

DR PDB; 1D0J; 26-SEP-01.

DR MGD; MGI:1101059; Tnfrsf9.

DR InterPro; IPR001368; TNFR_c6.

DR Pfam; PF00020; TNFR_c6; 1.

DR SMART; SM00208; TNFR; 2.

DR PROSITE; PS00652; TNFR_NGFR_1; 1.

DR PROSITE; PS50050; TNFR_NGFR_2; FALSE_NEG.

KW Receptor; Transmembrane; Glycoprotein; Repeat; Signal; 3D-structure.

FT SIGNAL 1 24

FT CHAIN 25 256 TUMOR NECROSIS FACTOR RECEPTOR

FT SUPERFAMILY MEMBER 9.

FT DOMAIN 25 187
EXTRACELLULAR (POTENTIAL).

FT TRANSMEM 188 208 POTENTIAL.

FT DOMAIN 209 256 CYTOPLASMIC
(POTENTIAL).

FT REPEAT 17 45 TNFR-CYS 1.

FT REPEAT 46 85 TNFR-CYS 2.

FT REPEAT 86 117 TNFR-CYS 3.

FT REPEAT 118 159 TNFR-CYS 4.

FT DISULFID 28 37 BY SIMILARITY.

FT DISULFID 31 44 BY SIMILARITY.

FT DISULFID 47 61 BY SIMILARITY.

FT DISULFID 64 77 BY SIMILARITY.

FT DISULFID 67 85 BY SIMILARITY.

FT DISULFID 87 93 BY SIMILARITY.

FT DISULFID 98 105 BY SIMILARITY.

FT DISULFID 101 116 BY SIMILARITY.

FT DISULFID 119 133 BY SIMILARITY.

FT DISULFID 139 158 BY SIMILARITY.

```
FT   CARBOHYD    128    128       N-LINKED (GLCNAC...)
(POTENTIAL).

FT   CARBOHYD    138    138       N-LINKED (GLCNAC...)
(POTENTIAL).

SQ   SEQUENCE   256 AA;  27598 MW;
 93A10D03C60813C4 CRC64;

     MGNNCYNVVV IVLLLVGCEK VGAVQNSCDN CQPGTFCRKY
     NPVCKSCPPS TFSSIGGQPN CNICRVCAGY FRFKKFCSST
     HNAECECIEG FHCLGPQCTR CEKDCRPGQE LTKQGCKTCS
     LGTFNDQNGT GVCRPWTNCS LDGRSVLKTG TTEKDVVCGP
     PVVSFSPSTT ISVTPEGGPG GHSLQVLTLF LALTSALLLA
     LIFITLLFSV LKWIRKKFPH IFKQPFKKTT GAAQEEDACS
     CRCPQEEEGG GGGYEL
//
```

Each line begins with a two-character line code, which indicates the type of data contained in the line. The current line types, line codes and the order in which they appear in an entry are shown in the Table A.1.

Some line types are found in all entries, others are optional. Some line types occur many times in a single entry. Each entry must begin with an identification line (ID) and end with a terminator line (//).

Table A.1: SWISS PROT Line Code Table.

Line code	*Content*	*Occurrence in an entry*
ID	Identification	Once;starts the entry
AC	Accession number(s)	Once or more
DT	Date	Three times
DE	Description	Once or more
GN	Gene name(s)	Optional
OS	Organism species	Once or more
OG	Organelle	Optional
OC	Organism classification	Once or more
OX	Taxonomy cross-reference(s)	Once or more
RN	Reference number	Once or more
RP	Reference position	Once or more
RC	Reference comment(s)	Optional
RX	Reference cross-reference(s)	Optional
RA	Reference authors	Once or more
RT	Reference title	Optional
RL	Reference location	Once or more
CC	Comments or notes	Optional
DR	Database cross-references	Optional
KW	Keywords	Optional
FT	Feature table data	Optional
SQ	Sequence header	Once
	(blanks) sequence data	Once or more
//	Termination line	Once; ends the entry

APPENDIX B

Example of a Feature Vector

In the following example, a feature vector used in the experiments is illustrated. Since our κ=30 and each row in PAM250 matrix has dimension 20, it yields 20×30 components for each feature vector. Recall that, there are $(len(\hat{X}) - \kappa + 1)$ substrings in a protein sequence in total.

```
-1 0 -2 -3 -5 -1 -2 -3 -2 2 4 0 6 0 -2 -2 -1 -4 -2

2 -2 6 0 -1 -4 1 -1 -3 2 -2 -3 3 0 -4 0 0 -1 2 -4

-2 1 0 1 0 0 -1 0 1 -1 -1 -3 0 -2 -3 1 2 1 -2 -3 -1

-1 2 2 1 -3 3 1 -2 6 -2 -2 0 -2 -2 0 -1 -1 -3 0 -2

1 -1 0 0 -2 -1 0 0 -1 0 -2 0 -1 -3 0 1 3 -5 -3 0 1

-3 0 1 -3 -1 0 5 -2 -3 -4 -2 -3 -5 0 1 0 -7 -5 -1

-2 -3 -3 -4 -6 -2 -3 -4 -2 2 6 -3 4 2 -3 -3 -2 -2
```

-1 2 -2 6 0 -1 -4 1 -1 -3 2 -2 -3 3 0 -4 0 0 -1 2

-4 -2 2 -2 0 0 -2 0 0 1 -1 -1 -2 -1 -1 -3 1 1 1 -6

-3 0 -2 -3 -3 -4 -6 -2 -3 -4 -2 2 6 -3 4 2 -3 -3 -2

-2 -1 2 0 -2 -2 -2 -2 -2 -2 -1 -2 4 2 -2 2 -1 -1 -1

0 -6 -2 4 2 -2 0 0 -2 0 0 1 -1 -1 -2 -1 -1 -3 1 1 1

-6 -3 0 1 0 0 -1 -3 0 -1 0 0 -2 -3 -1 -2 -5 6 1 0

-6 -5 -1 1 -3 0 1 -3 -1 0 5 -2 -3 -4 -2 -3 -5 0 1 0

-7 -5 -1 -3 -4 -2 -4 0 -4 -4 -5 0 -1 -1 -4 -2 7 -5

-3 -3 0 10 -2 1 0 0 -1 -3 0 -1 0 0 -2 -3 -1 -2 -5 6

1 0 -6 -5 -1 -2 -3 -3 -4 -6 -2 -3 -4 -2 2 6 -3 4 2

-3 -3 -2 -2 -1 2 -2 -3 -3 -4 -6 -2 -3 -4 -2 2 6 -3

4 2 -3 -3 -2 -2 -1 2 -2 -3 -3 -4 -6 -2 -3 -4 -2 2

6 -3 4 2 -3 -3 -2 -2 -1 2 -2 -3 -3 -4 -6 -2 -3 -4

-2 2 6 -3 4 2 -3 -3 -2 -2 -1 2 -2 -4 -4 -5 12 -5

-5 -3 -3 -2 -6 -5 -5 -4 -3 0 -2 -8 0 -2 -2 -3 -3

-4 -6 -2 -3 -4 -2 2 6 -3 4 2 -3 -3 -2 -2 -1 2 -2

-3 -3 -4 -6 -2 -3 -4 -2 2 6 -3 4 2 -3 -3 -2 -2 -1

2 2 -2 0 0 -2 0 0 1 -1 -1 -2 -1 -1 -3 1 1 1 -6 -3

0 2 -2 0 0 -2 0 0 1 -1 -1 -2 -1 -1 -3 1 1 1 -6 -3

0 1 -1 0 0 -2 -1 0 0 -1 0 -2 0 -1 -3 0 1 3 -5 -3 0

-2 6 0 -1 -4 1 -1 -3 2 -2 -3 3 0 -4 0 0 -1 2 -4 -2

1 0 0 -1 -3 0 -1 0 0 -2 -3 -1 -2 -5 6 1 0 -6 -5 -1

0 -1 2 4 -5 2 3 1 1 -2 -4 0 -3 -6 -1 0 0 -7 -4 -2

1 0 0 -1 -3 0 -1 0 0 -2 -3 -1 -2 -5 6 1 0 -6 -5 -1

APPENDIX C

Usage of SOM-PAK for Clustering

The package SOM-PAK [24] is used for clustering purposes. This software package contains all programs necessary for the correct application of the Self Organizing Map algorithm in the visualization of complex experimental data. There exist many versions of the SOM. The basic philosophy, however, is very simple and already effective as such, and has been implemented by the procedures contained in this package. Some crucial points os SOM-PAK related with our study is briefly explained.

C.1 Map File Format

The x-coordinates of the map (column numbers) may be thought to range from 0 to n-1, where n is the x-dimension of the map, and the y-coordinates (row numbers) from 0 to m-1, respectively, where m is the y-dimension of the map. The reference vectors of the map are stored in the map file in the following order:

```
1       The unit with coordinates (0, 0).
2       The unit with coordinates (1, 0).
         ...
n       The unit with coordinates (n-1, 0).
```

```
n+1     The unit with coordinates (0, 1).
         ...

n.m      The last unit is the one with coordinates
 (n-1, m-1).
```

The distance between two units in the map is computed as an Euclidean distance in the (two dimensional) map topology.

C.2 Running of Modules

The initialization is done by "randinit" program. This program initializes the reference vectors to random values. The vector components are set to random values that are evenly distributed in the area of corresponding data vector components. The size of the map is given by defining the x-dimension (-xdim) and the y-dimension (-ydim) of the map. The topology of the map is defined with option (-topol) and is either hexagonal (hexa) or rectangular (rect). The neighborhood function is defined with option (-neigh) and is either step function (bubble) or Gaussian (gaussian). The call of the program with our parameters is as follows:

```
> randinit -din ex.dat -cout ex.cod -xdim 25 -ydim 25
 -topol rect -neigh gaussian -rand 11513
```

For the training, "vsom" program is used. This program trains the reference vectors using the self organizing map algorithm. The topology type and the neighborhood function defined in the initialization phase are used throughout the training. The program finds the best-matching unit for each input sample vector and updates those units in the neighborhood of it according to the selected neighborhood function.

The initial value of the learning rate is defined and will decrease linearly to zero by the end of training. The initial value of the neighborhood radius is also defined and it will decrease linearly to one during training (in the end

only the nearest neighbors are trained). The call of the program for the ordering is:

```
\> vsom -din file.dat -cin file1.cod -cout file2.cod
 -rlen 5000 -alpha 0.05 -radius 10
```

and for the fine tuning is:

```
\> vsom -din file.dat -cin file1.cod -cout file2.cod
 -rlen 20000 -alpha 0.02 -radius 3
```