# Towards protein function annotations for matching remote homologs

*Seak Fei Lei*

Submitted to the Department of Electrical Engineering & Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master's of Science

**Thesis Committee:**

Dr. Jun Huan (Committee Chair)

Dr. Xue-wen Chen (Committee Member)

Dr. Arvin Agah (Committee Member)

Date Defended

Defended: June 27, 2008

The Thesis Committee for Seak Fei Lei certifies
That this is the approved version of the following thesis:

**Towards protein function annotations for matching remote homologs**

Approved: June 27, 2008

Committee:

Committee Chair

Committee Member

Committee Member

Date Approved

# Abstract

**Motivation:** Identifying functional homologs from their protein structures is an important problem in biology. One way to approach this is to discover their common local structures (i.e. motif) among protein families. Unfortunately, most of the motif models are inadequate to characterize the structural diversities, especially when the proteins are distantly related. In this study, we first introduce a statistical model, together with a semi-supervised refinement method, to perform post-processing on the motif obtained from a motif discovery algorithm or from motif database. Our model makes use of Markov Random Fields (MRF), which provide a large search space to optimize the motif while preserving the dependencies among neighbor elements. The resulting model not only can better represent the underlying patterns for different protein families, but also can enable the power of functional annotation from a set of unknown proteins using probability approximations. In addition, we develop two filter approaches (three methods) to further eliminate the false positives introduced by any motif models. By considering the local environment around the active sites of each family, the filters reject proteins to match with the model without similar environment profiles.

**Results:** Our experimental results, as evaluated in five sets of enzyme families with less than 40% sequence identity, demonstrated that our methods can obtain more remote homologs that could not be detected by traditional sequence-based methods. At the same time, our method could reduce large amount of random matches which were originally introduced by the motif representations. On average, our methods could improve about 13 % of the functional annotation ability (measured by their AUCs). In certain experiments, our improvement went even up to 70%.

# Acknowledgements

I want to thank my parents and my sister for their love and support throughout my academic career. They not only provided me this wonderful opportunity to study at University of Kansas, but also encouraged me not to give up when I faced difficulties.

I want to thank my advisor, Professor Jun Huan for his guidance of my research works. This project would not be possible without his supervision. Professor Huan paid numerous efforts for assisting me to finish my thesis and defense presentation. In addition, he gave me advices which are very useful for my future career. I would also wish to thank my committee members, Professor Agah and Profess Chen for reviewing and giving me valuable comments on my paper.

Finally, I want to thank all my friends and colleagues who made this journey fun and memorable.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This paper focuses on one of the fundamental problems in biology — identifying protein functions by optimizing the structure motifs for a specific protein family. Although there are many experimental methods available to determine their functions, such as ChIP on chip [15], or microarray analysis etc., most of them engage in expensive and lengthy processes. To make the situation worse, the rapid increase of protein structures with unknown functions in the last ten years (as seen from the exponential growth of Protein Data Bank [8] in figure 1.1) makes the experimental methods become infeasible. As a result, there is an urge need for performing protein function predictions *in silico*.

## 1.1 Motivation

Numerous connections between proteins and their functions have been investigated when creating computer-based annotation methods, including algorithms that utilize proteins' primary sequences and protein-protein interactions throughout a biological process. Specifically in this paper, we will design our methods

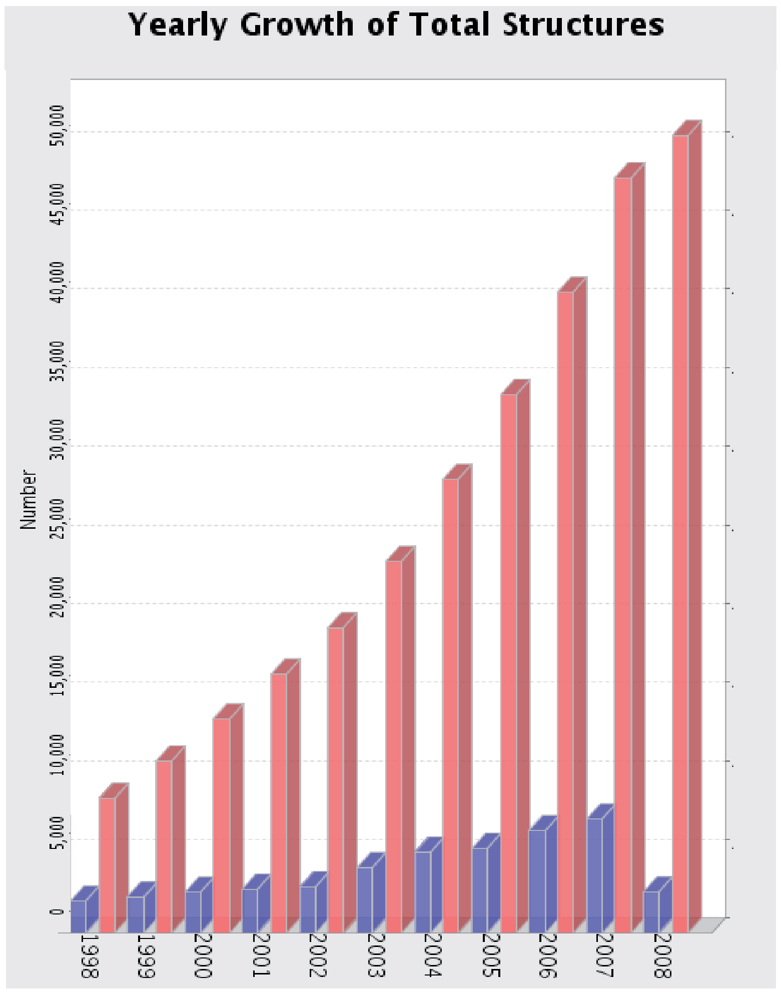


**Figure 1.1.** The growth of Protein Data Bank for the last 10 years. Protein Data Bank is a well-known public repository for 3-D molecular structures such as proteins. The red bars indicate total number of proteins in the repository, and the blue bars reveal the number of proteins added to the repository. The graph is obtained from the RCSB web site [1]

based on the complex relationship between protein 3D structures and protein functions [22]. Compare with sequence-based methods, structure-based methods bring an extra layer of information— spatial arrangements of amino acids within proteins, which further increase the knowledge about the locations of active sites (i.e. local structural information of proteins) and their mechanisms of reactions.

Furthermore, studying protein structures also provides us with the understanding of proteins from different perspectives:

1) It opens the possibilities of organizing proteins in an unified manner: Due to the limited number of possible folds in protein structures, people create many schemes to classify proteins based on their global substructure similarities. These schemes not only reduce the time needed to access the huge body of proteins, but also allow us to study the links between protein functions and their global structures, as well as predicting protein structures from existing sources [39].

2) It reveals the evolutionary relationship: Studies on retinol-binding and odorant-binding proteins in [49] show that proteins' tertiary structures generally evolve slower than their primary sequential structures. Two proteins which have low sequence identity could still come from the same ancestor if they have a similar structure. In other words, the homolog relationship can be derived from their structures. In addition, since homolog proteins ( or proteins that come from the same protein family) are likely to have similar functions, protein structures become an useful tool for function prediction (However, the mappings between protein structures and functions are unclear, as seen in the Challenges section).

3) It assists the structure-based drug design: According to [40], drug design involves in two steps. The first step is to identity the functional regions in the target protein we want to activate/deactivate. The second step is to synthesis or to discover molecules which can tightly adhere to target regions. Both steps require the use of 3D information, for example, protein structure prediction and functional prediction are needed in the first step; whereas protein docking or ligand design also refers protein structures for creating its complement sites.

In short, the ability to offer accurate functional annotations without the cost of lab experiments, along with the advantages brought by the knowledge of protein structures necessitate the development of automated tools for protein function annotation based on structure analysis.

## 1.2 Challenges

As mentioned in the previous section, although the there exists a connection between the protein structure and protein function through homology, the mappings between them are complicated. An extensive number of researches have shown that global structure similarity does not necessarily imply similar function. For example, proteins with TIM barrels have remarkably similar global structures, but having diverse functions [46]. Likewise, similar functions do not necessary imply similar global structure [29]. One reason is that a protein can contain multiple substructures (domains), which are responsible for different functions. Therefore, a naive mapping between protein structures and protein functions will result in poor annotations [30]. On the other hand, local structures may expose their molecular and biological functions such as protein-protein interaction, ligand binding, and catalytic reactions. These local structures can be expressed as structural motifs.

There exist many types of structural motif representations like lists or graphs etc., and there are many methods to discover them respectively. Nonetheless, not all motif models can truly characterize the protein families. This may due to a couple of reasons:

1) Some motif models have limited search space. A good example will be a simple sequence model with k residues. Assume that no regular expression

[1]http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100

characters is allowed, the maximum number of motif instances it can represent would be $20^k$, which restricts its ability to fully describe the protein with similar functions.

2) Some motif models impose strict assumptions, e.g. independence among residues.

3) Some motif search algorithms do not work efficiently when discovering the best motif for the model. As a result, the motif model requires multiple motif instances to describe a single protein family, which may generate large amount of false matches when detecting distant homologs.

In general, there are two challenges to functional annotations using motif models:

1) The motif model needs to be *sensitive* enough to accommodate variations of global structures as well as the active sites within a family. This can be caused by different reasons such as combinations of various non-functional domains, mutations during evolution, or experimental errors etc.

2) The motif model needs to be *specific* enough to a protein family. Sometimes random matches may cause proteins which have different functions to be considered as the same protein family. A good motif model should eliminate as many false positives as possible, so that no extra resources would be wasted on verifying their functions by experiments.

## 1.3 Contributions of this thesis

To tackle these challenges, we introduce three approaches (total four methods) to refine the existing structural motif model. Instead of building a brand new motif from sketch, we take an existing (unrefined) motif as an initial motif. This motif will then be further refined by one of our proposed methods. The resulting (refined) motif should recover more function homolog proteins from the data set. In other words, it is an optimized representation of the motif. The following lists the three approaches we explored in this paper:

1) Iterative motif refinement algorithm with environment filter approach

2) The environment filter approach

3) The extended motif filter approach, which contains two possible methods for aggregating the final results — voting method and feature vectors method.

The first approach is also known as the model based models since we make use of a statistical model known as Markov Random Filed to generate a new refined motif. Markov Random Field, as one of the statistical graph models, preserves the overall topology of the protein structures, while still maintains the distribution of each node. Therefore, MRF is a good model for motif refinement. Distinct from other existing models (see section 3), our algorithm has the following advantages:

1. Markov Random Field, as one of the statistical graph models, preserves the overall topology of the protein structures, while still can maintain the distribution of each node. Therefore, MRF is a perfect model for motif refinement.

2. MRF has a relaxed assumption by allowing dependencies among neighboring elements, which is more practical to the real world data set like chemical structures.

3. Our resulting model has discriminative power to predict if a protein has similar functions (i.e. functional annotation).

The second and the last approach is called the filter-based methods. We assume that surrounding environment around a motif plays an important role on determining if that position is a binding site or not. This assumption is set up as we observe motif pattern appears in different parts of a protein other than its active location. Based on this assumption, we develop these two filters to eliminate those false matches. For each protein family, the first filter calculates frequencies of residues around the active site. If a candidate protein had a very different distribution than other proteins from the same family, it would be rejected. For the second filter, we include the neighboring residues to enlarge the initial motif. As a result, the number of random matches should be reduced.

## 1.4 Thesis organization

The rest of the paper is organized as follows. In chapter 2, we go through some basic ideas about proteins and their functions, as well as some mathematical backgrounds for our methods. In chapter 3, we review some of the latest developments on motif discovery and refinements from protein structures and sequences. In chapter 4, we introduce our novel statistical motif representation and the filters. Also, we discuss our approach of refining or filtering the preliminary results. In chapter 5, we present our experimental study and provide some performance analyses. In chapter 6, we draw some conclusions from our experiments and discuss the future works.

# Chapter 2

# Background

This section introduces several concepts utilized in this paper. The basic idea of proteins and their links to our bodies will be discussed in detail, that includes topics like protein structures, their synthesis methods, and their functions etc. Next, we will talk about how proteins are organized and stored electronically. That includes structural database like SCOP. Finally, two mathematical models which represent proteins and their motifs will be revealed in last two sections.

## 2.1 Protein sequence and structure

In biology, proteins are manufactured by two biochemical processes inside a cell — transcription and translation. Originally, the genetic information, which is the blueprint of protein synthesis, stores in the deoxyribonucleic acid (DNA) sequences. DNA uses four types of nucleotide bases: adenine (A), cytosine (C), guanine (G) and thymine (T) to record genetic information and to form a double helix shape in the cell nucleus (or the center of the prokaryotic cells). DNAs are normally organized as chromosome, a structure that holds the DNAs with a set of proteins. Depend on the species, the total number of chromosomes varies in a cell, for example, there are total 23 pairs of chromosomes in human cells. However, during the *transcription* process, the DNA strand is loosen, allowing RNA polymerase to go through one of the strand. The product of the *transcription* process is a messenger ribonucleic acid (mRNA) sequence, which is built based on the following complementary base paring rules:

*Nucleotide base mappings from DNAs to RNAs during transcription*

- $A(adenine) \rightarrow U(uracil)$
- $T(thymine) \rightarrow A(adenine)$
- $C(cytosine) \rightarrow G(guanine)$
- $G(guanine) \rightarrow C(cytosine)$

Compare with DNA structure, RNA only contains one strand (but it still can fold into a hairpin-like structure). Also, RNA has uracil instead of thymine for its basic composition. Note that only portions of the DNA sequence are actually transcribed into mRNA, and such regions are known as genes. Although other regions may involve in controlling the gene expression, it will not be encoded into a protein.

In eukaryotic cells, some mRNAs (or called pre-mRNAs in this case) will continue to go through the slicing process by spliceosomes so that all the intronic regions are removed (intros are sequences that will not be translated into a protein, sequences that will eventually be translated into a protein are known as exons). The post-processed RNAs, which are called mature RNAs, will move towards the ribosome that locates in the cytoplasm or on the rough endoplasmic reticulum for the *translation* process. Basically, ribosome provides a spot for encoding mRNA sequence to a polypeptide chain. The peptide chain will finally form a protein after folding and transformations take place. Specifically, each codon (three consecutive non-overlapping nucleotide bases) from a mRNA is converted into an amino acid residue brought by the transfer RNA (tRNA). The entire transcription/translation process is also defined in the central dogma of molecular biology [19] since the genes in DNA are finally expressed through proteins, and proteins finally affect the organism through its phenotype.

In chemistry, protein is formed by polymerization process from amino acids. There are 20 different types of amino acids in total. However, their molecular compositions are somewhat similar. They all consist of five parts:

1) a central carbon atom ($C_\alpha$) which connects other parts of the amino acid

2) a hydrogen atom (H)

3) a carboxylic acid functional group (COOH), and

4) an amino group (the side chain) that composes of varies element groups such as hydrogen, carbon, nitrogen, oxygen etc.

Based on the structures of the amino group, the corresponding amino acids demonstrate different chemical properties, and thus forming 20 amino acids. As seen in Figure 2.1, amino acids can be classified by their polarities, hydrophobicities/hydrophilicities, and acid/basic properties. These characteristics are preserved even when the polypeptide chain is formed. This is actually one of the reasons why protein folding occurs after translation takes place.

During the translation phase, the hydrogen atom from one amino acid creates a covalent bond with the carboxylic acid group from another amino acid by the

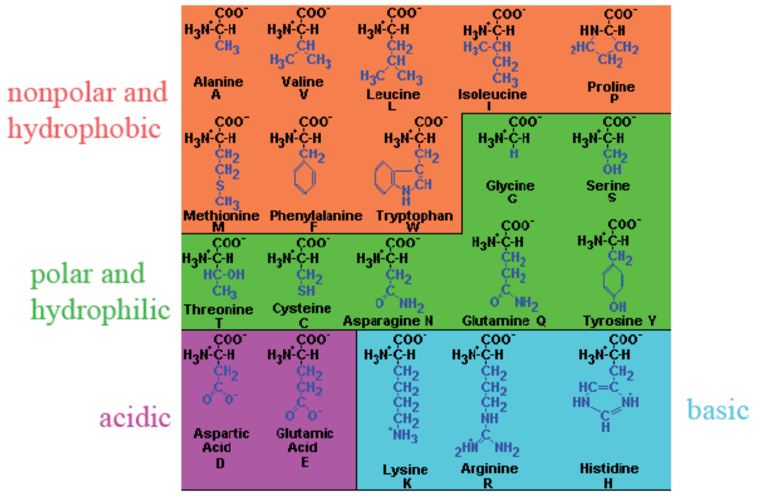


**Figure 2.1.** Twenty amino acids are grouped by their chemical properties. (This figure is obtained from http://www.chromnet.net)

dehydration synthesis reaction, which results in one water molecule. This chain reaction retains as tRNA carries more amino acids to the ribosome. As a result, a long chain of amino acid, known as polypeptide chain is synthesized.

The polypeptide chain does not have a rigid shape. This is because two bonds on the $C_\alpha$ atom can freely rotate in two angles ($\Psi$ and $\Phi$ angles), which causes the chain to fold in three dimensions and thus creating different conformation of protein structures. Researchers analyze these structures and come up with four levels of protein structures:

1) Primary Structure: The primary structure of a protein refers to the order of amino acid sequence in the polypeptide chain.

2) Secondary Structure: The secondary structure describes the local consecutive substructure of a protein. This is formed by the non-covalent bonds among different residues, like hydrogen bonds and ionic bonds, which make the chain to fold itself. Some examples of secondary structures include $\alpha\ Helix$, $\beta Sheet\ and\ \beta\ Turn$

etc.

3) Tertiary structure: Further folds of the secondary structural elements form the tertiary structure of a protein. Unlike secondary structure, folding of the tertiary structure is motivated by the hydrophobic and hydrophilic regions of a protein. The hydrophobic part of the protein tends to hide inside the protein (the protein core), whereas the hydrophilic part tends to flip outside to interact with the fluid molecules around it.

4) Quaternary structure: Multiple polypeptide chains may bind together with other inorganic chemicals in order to construct a protein complex. At this stage, the protein is active and ready to perform its function.

In this paper, we will use the proteins' tertiary structure for functional annotation. This makes sense since their 3D arrangements decide which atoms should expose to the outside environment. Other substances like chemical compounds may interact with them only when that particular region can be reached. On the other hand, primary structure analysis alone cannot provide such information. Indeed, people have observed that higher level of protein structure is often a better indicator when predicting their functions.

The existence of structural levels in proteins also allows us to categorize proteins systemically, since proteins share similar structures may imply similar functions. There are many structural databases using their folding and sequential information to distinguish proteins. One of them is called SCOP database [45] (Structure Classification of Proteins). SCOP is maintained manually by experts. It uses four hierarchical structural levels to classify proteins:

Class—it categorizes the general construction of proteins based on their secondary structures, for example, the largest four classes in SCOP are $\alpha$ class, $\beta$ class, $\alpha/\beta$ class, and $\alpha + \beta$ class.

Fold—it collects proteins which have similar secondary structure arrangements.

Superfamily—groups of proteins with probable evolutionary relationships due to their structural and functional similarities. However, their sequence identities can be very low.

Family— proteins with clear evidences to be related, and their residue identities $>= 30\%$.

Other structural databases such as DALI [33](now called SSM) and CATH [47] use slightly different hierarchical schemes to classify proteins, and they are assisted by computer-based structural comparison algorithms. Although these semi-automatic structural databases can process large amount of proteins in a short period of time, there is no guarantee to their classification quality. In the experiment section, we will use SCOP database to identify homologs from our dataset.

## 2.2 Protein functions and structural motifs

One of the reasons why determining protein functions becomes an important problem is that proteins are building blocks for many organisms. According to [49], their functions can be viewed from different perspectives: A protein has its function in molecular level, e.g. enzyme's molecular function is to catalyze substrates to products, while transport protein's molecular function is to carry substances from one place to another. A protein also has its function within a biochemical pathway. For example, a protein called proteasome involves in the fatty acid oxidation process. Finally, a protein's function can be determined from border biological processes context like cell division, neurons signal transduction etc. Most of the methods we introduce in this paper refer proteins functions to their molecular functions. This is because people normally believe that molecular functions can be better described by their 3D structures, which is our central topic of the paper. Additionally, molecular function is often a good indicator of other perspectives of protein function.

As mentioned in the last section, protein functions can be determined by global structural hierarchies, i.e. structural databases offer connections between protein structures and their functions. Yet, global structure similarity does not always imply functional similarity. The TIM barrels proteins example in the introduction section presents a good piece of evidence. Moreover, studies [25] on SCOP and enzyme functional database(EC) discover that about 69% of the enzyme functions (third levels) can be found in different SCOP super-families. Some SCOP super-families can include proteins from up to five different enzyme functions. Local structural similarity, on the other hand, may reveal protein function. In fact, only small part of a protein actually controls its function. This region is called active site. Experiments have shown that enzyme T4 Lysozyme has good structural and functional stability under large amounts of residue substitutions [43], as long as the active site is not affected.

In order to associate the local protein structures with their functions, two concepts need to be introduced— Domains and motifs. Domain is a subset of protein which can fold into a stable structure independently. It is believed that there is limited number of domains. Because of the shuffling effects during evolution (also known as gene fusion), proteins considered as related often share the same domain(s). Researchers [9] also point out that domains can be treated as functional units. A protein may contain more than one domain, which is called multi-domain proteins.

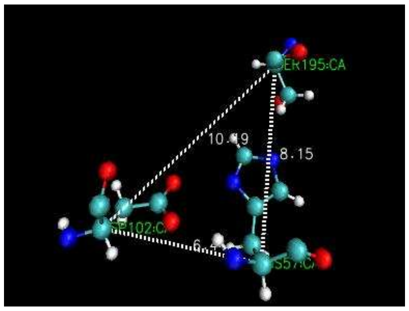


**Figure 2.2.** The structure of catalytic triad motif from Eukaryotic serine protease family (ESP)

Motif is a small part of a protein which actually interacts with other substances to perform its molecular function. Motifs are normally very short in length (under 20 aa [49]) and they often locate inside protein domains. Proteins which share the same motif can be characterized as the same family [10]. There are two types of motifs: sequential and structural motifs. Typically, they both use some pattern matching schemes (e.g. regular expression) to represent the amino acid compositions of the motif. However, structural motif provides additional spatial and/or folding information of amino acids. For example, figure 2.2 shows a well-known catalytic triad motif consists of serine, histidine, and aspartic acid in Eukaryotic serine protease family (ESP).

In this paper, the problem protein function annotation is stated as follows: Given a protein with its corresponding motif from a protein family, retrieve all the unknown proteins sharing the same molecular function (or come from the same family) from a data set. The given protein is called the query protein, and its motif is known as the initial motif.

## 2.3 Graph Theory

In this paper, we model the 3D structure of the protein using labeled graph from graph theory in mathematics . A labeled graph is defined as the following,

**Definition 1** *(**Labeled Graph**)A labeled graph $G$ is a five elements tuple $G = (V, E, \Sigma_V, \Sigma_E, \lambda)$ where,*

- *$V$ is a set of vertices or nodes.*
- *$E$ is set of undirected edges $E = V \times V$.*
- *$\Sigma_V$ is disjoint sets of vertex labels, e.g. in protein structural analysis, this can be the set of 20 amino acid types*
- *$\Sigma_E$ is disjoint sets of edge labels, e.g. Euclidean distance between two connected nodes*
- *$\lambda$ is a function that assigns labels to the vertices and edges.*

The node ($V$) in our protein representation is a set containing the coordinates of the $C_\alpha$ atoms of each amino acid residue; while the $\Sigma_V$ is a set of 20 amino acid types. Figure 2.3 shows an example of three proteins using labeled graph representation. In fact, this representation has been used in many researches, including our previous work [34]. However, unlike our prior representation, discrete distance labels ($\Sigma_E$) are replaced by their actual Euclidean distances for

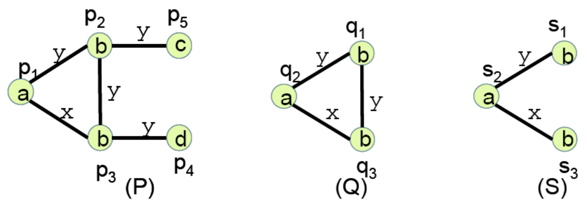


**Figure 2.3.** Examples of labeled graphs. In this paper, protein/motif structures are modeled using labeled graph. Node labels such as $a, b, c, d$ represent amino acids, and the edge labels like $x, y$ are the Euclidean distance between two nodes.

greater flexibility when matching to a motif (we use distance root-mean-square deviation(dRMSD) to match two graphs, see 4.1.1 for more detail). To increase the matching efficiency, two types of edges distances are included in the graph — bond edges and proximity edges. Bond edges are the polypeptide chains appear in the protein primary sequence. Proximity edges, on the contrary, consider the relations of the neighbors in its 3D structure. Two residues are treated as connected with a proximity edge if their Euclidean distance is less than a threshold $\delta$. In this paper, we select the $\delta$ to be 8.5 $\mathring{A}$.

## 2.4 Markov Random Field

Markov Random Field (MRF) is a statistical model used for studying spatial patterns. Although one of its special variations, known as Markov Chain, is widely used in multiple sequence alignment. Due to the fact that MRF combines both statistical and graph theories, it is a perfect tool for summarizing structural patterns. In fact, we will make use of MRF to construct an optimized motif representation based on a set of proteins—a probability distribution of the motif node labels.

MRF is defined as a generalized stochastic process with a collection of random

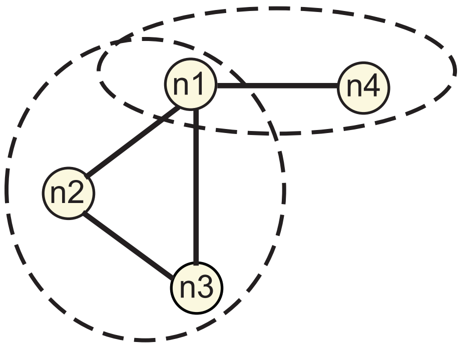


**Figure 2.4.** Example of Markov Random Field Topology, the broken circles indicate two maximal cliques (n1-n2-n3 and n1-n4),which imply two potential functions are needed for this model.

variables $X_s : s \in V[G]$ where $V[G]$ is a set of nodes in an undirected graph $G$. Fig (2.4) shows an example of MRF. Similar to Markov Chain, MRF also follows the Markov property (1st order). For all nodes $t \in \{1, \ldots, n\}$, we have:

$$\begin{aligned} &P\{X_t = x_t | X_s = x_s, s = V[G] - t\} \\ &= P\{X_t = x_t | X_{s'} = x_{s'}, s' = N(t)\} \end{aligned}$$

where $N(x)$ is the neighbor of node $x$ (nodes that connected to $x$ by an edge in $G$).

In other words, Markov property assumes that the conditional probability distribution of each node will depend on its neighbors' distributions. In our example, Fig. (2.4), n1 depends on n2 ,n3, and n4. This assumption can greatly reduce the parameters needed to compute the joint probability of the model. Distinct from the directed graphical model, e.g. Bayesian Network, where the joint probability can be calculated directly using Bayes theorem; the joint probability of MRF is defined by the Hammersley-Clifford theorem,

**Theorem 1** *Suppose that $X = (x_1, \ldots, x_n)$ has a joint probability mass function. $X$ is a Markov Random Field on $G$ if and only if $X$ has a Gibbs distribution with respect to $G$.*

The proof of the theorem is available in [27]. And the joint probability of the Gibbs distribution can be stated as the following,

$$p(x) = \frac{1}{N} \prod_{c \in C} V_c(x_c)$$

where $N$ is the normalization factor , $V_c$ is known as a potential function of a maximal clique $c$, and $x_c$ is one configuration of clique $c$. A maximal clique is a complete subgraph which cannot be a subset of another larger complete subgraph. A potential function is mapping between a set of node labels from a maximal clique (an instance) and a real number (potential value). Different cliques in G correspond to different potential functions. Notice that each potential function is now independent with each other, and thus using multiplication in the formula. In our algorithm, suppose a MRF has been trained from a set of proteins obtained from the previous iteration, we can then re-apply the graph matching technique to the protein data set, as mentioned in section 4.1.1, together with the newly built MRF as a scoring function. The joint probability output will be a similarity measure between the motif model and a specific protein. Proteins with high joint probabilities (or high scores) will be considered as related, and thus being included in the final result.

Figure 2.5 shows a sample training set (upper portion of the figure), which are composed of four labeled graphs. Given vertex label set equals to $0, 1$, a MRF model is constructed (lower portion of the figure). Since the training

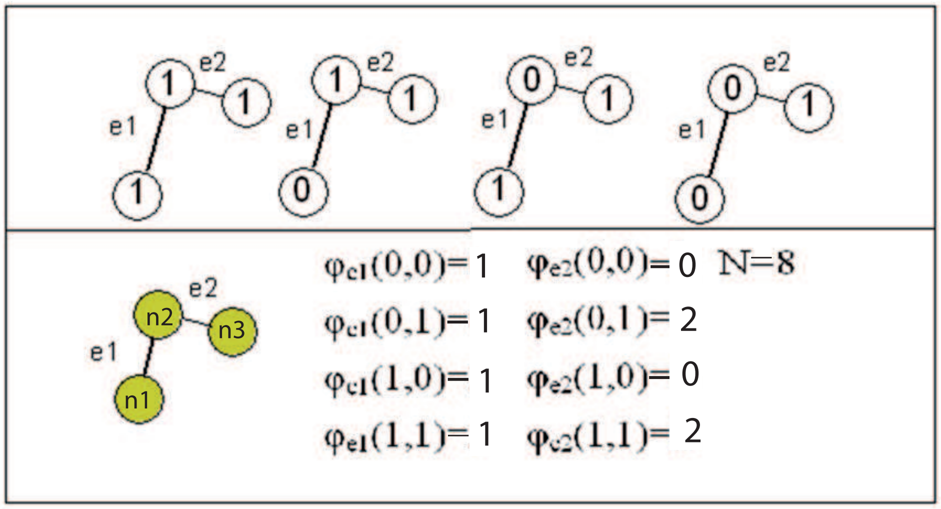


**Figure 2.5.** (Upper) An example graph database with node label domain $0, 1$. (Lower) The Markov random field generated from the data set. The total number of parameters needed to construct the model is nine.$\Phi_{e1}$ and $\Phi_{e2}$ are known as the potential functions. $N$ is called the normalization factor

graphs contain two maximum cliques ($e_1$ *and* $e_2$), there are two potential functions in the MRF—$\Phi_{e1}, and \Phi_{e2}$. Within each potential function, there are four pairs of values in its function domain, as all possible inputs for this function are: $(0, 0), (0, 1), (1, 0), (1, 1)$. $N$ is the normalization factor of the MRF. In short, to build up this MRF, total 9 parameters are needed to be estimated (eight functions values for $\Phi_{e1}, and \Phi_{e2}$, plus the normalization factor $N$). The process of determining each potential value (i.e. training the MRF) will be discussed in section 4.1.2.1. To calculate the joint probability, on the other hand, we can use the formula in Theorem 1. For example, if we want to calculate joint probability p(1,0,1) and p(0,0,0):

$$p(1, 0, 1) = 1/N * \Phi_{e1}(1, 0) * \Phi_{e2}(0, 1) = 1/8 * 1 * 2 = 1/4$$

$$p(0, 0, 0) = 1/N * \Phi_{e1}(0, 0) * \Phi_{e2}(0, 0) = 1/8 * 1 * 0 = 0$$

Both results are consistent with the maximum likelihood of the training set. In other words, the MRF fully describes the training set.

# Chapter 3

# Related Work

Although we concentrate on predicting protein functions using local structural information in structural motifs, functional annotations using proteins' primary structures have been greatly researched. Many structural-based annotation methods are also inspired by the sequence-based methods. Therefore, we will first present some important techniques in sequence-based annotation. We will then switch to the current structural-based methods.

## 3.1 Sequence-based annotation

Function annotations using protein sequences involve in two related topics. The first problem is to develop a motif representation such that the representation can accurately recognize variants of the active sites in a protein family. The second problem is to develop algorithms for predicting protein functions based on existing protein families with the above motif representations. And depend on the comparison methods, they can be divided into two categories—model-based and homology-based sequence comparison annotation.

### 3.1.1 Motif representation

Most of the current sequence-based motifs can be grouped into two popular representations: String representations and probability matrix representations. Despite the fact that many methods use strings for their motif representations, their definitions are somewhat different. One of the simplest string representations is known as consensus sequence. Consensus sequence is an amino acid sequence of length $L$. A protein matches to the motif if there exists a subsequence with length $L$ that has the exact same sequence order. Although this motif can recognize the active site without any ambiguity, as mentioned in the introduction, the search space of the motif is limited. To solve this problem, people use ideas like allowing mismatches and wild card characters to achieve partial matches. For instance, in protein domain database PROSITE [32], character 'x' can match to all 20 amino acids. Nevertheless, the introduction of these 'tricks' may reduce the precision of the motif, as the probability of random matches also increases. In short, there is always a trade-off between the precision and recall of a representation. [21] compares the pros and cons of different consensus sequence motif representations.

Probability matrix is another common sequence motif representation. Again, many definitions of those values are possible. One of the common approaches is the position weight matrix (PWM) [14], it is matrix of size $20*L$ where $L$ is the length of the motif. Each column is the probability distribution of 20 amino acids appears at position m, and $0 <= m <= L$. This representation allows infinite solution space since all the numbers (or the probability values) are adjustable. However, so far there is no motif construction method which can provide an optimized motif using this representation [41]. Other probability matrices values including statistical model profiles [38] have also been explored.

### 3.1.2 Homolog-based sequence comparison methods

Given a protein sequence with unknown function, these group of methods compute the similarities between the input sequence and the sequences with known functions. Since high sequence identity usually entails analogous functions, the function of the protein can be determined from the comparison results. Alternatively, in machine learning community, these method can be viewed as K-nearest neighbor algorithms, which label the input sequence by the distances to the training samples (here distance are used as a similarity measure). Actually, sequence alignment, which is extensively used in bioinformatics, can use this idea to perform functional annotation. Specifically, the alignment process is to search for the greatest similarity among sequences. The alignment results can then be quantified to select the closest matches.

One the of most popular local sequence alignment algorithms is known as Basic Local Alignment Search Tool (BLAST) [56]. BLAST algorithm is divided into three phases: In phase one, the sequence is broken down into word snippets, which are the list of subsequences generated by a sliding window of width $w$. In phase two, the word snippets from one sequence will try to match word snippets from other sequences using scoring matrix such as BLOSUM and PAM (scoring matrix records the rate of one amino acid mutated to other residues overtime during evolution). The word pairs which have a score higher than a threshold $T$ will be recorded. In phase three, the matched word pairs extend their matches from either direction until the matching score drops below a cutoff. For each matched pair gathered from these three phases, a statistical value is calculated to illustrate the significance of the match. The alignment results will be used for creating motifs or performing function predictions.

Another sequence alignment method utilizes iterative approach like PSI-BLAST [3], which is an extension of BLAST algorithm. To start, one of the input sequences is used as a query sequence. And we apply BLAST with the conventional scoring matrix like BLOSUM and PAM to align a set of matched sequences with certain E-value threshold. Then we customize our scoring matrix by computing the probability distribution of each aligned position from previous BLAST result. The resulting matrix is known as PSSM (position-specific scoring matrix). This newly built PSSM will be used to retrieve a new set of sequences which will be the source for computing a new PSSM in the next iteration. This process continues until the number of discovered sequences converges. The advantage of this approach is that remote homology can be detected since this scoring scheme is tailored for that particular protein family. However, as seen later in the experimental section, we will show that PSI-BLAST can get into a local optimal solution due to its inability to pick up proteins with diverse 3D structures.

### 3.1.3 Model-based sequence annotation methods

With a set of protein sequences from the same protein family, these methods first construct statistical models to characterize this family. Note that models can either stand for the entire length of the sequences, or just the subsequences (sequential motifs) of the families. Typically, the output of these models calculates the degree of fitness between the input sequence and a given family. The function of a protein can then be assigned to the family with highest fitness (or probability) values. Among all the model based annotation methods, Hidden Markov model and Mixture model are two of the most popular models employed in this category. The rest of this section will discuss them in detail.

Hidden Markov Model (HMM) assumes that all sequences variations are controlled by a set of finite states, and the probability distributions of one state only depends on its previous state. There are three types of states when profiling the HMM— 1) Insertion state: indicates a gap is inserted when aligning to the model at specific position. 2) Deletion state: indicates removal of a residue occurred in the sequence. 3) Matched state: indicates a residue is aligned to this state in the model. In addition, there are two types of probability values associated with these states— 1) Transition probability: the probability of one state changes to another state. 2) Emission probability (only for the matched state): probability of amino acid that aligns to that matched state. Figure 3.1 shows the general structure of the HMM states. Three types of states (insertion, deletion, and matched) are shown in diamonds, circles and squares respectively. Notice that none of these states is observable— we do not know what states it goes through to align the sequence. To summarize, the following [2] lists the formal definition of HMM:

1. $N$ : Number of states in the model $S = \{S_1, S_2, ..., S_N\}$
2. $M$ : Number of distinct symbols can be observed, which is 20 amino acid types. $V = \{v_1, v_2, ..., S_M\}$
3. Transition probabilities $\mathbf{A} = [a_{ij}]$ *where* $a_{ij} = P(q_{p+1} = S_j | q_p = S_i)$
   $q_p$ is the current state at position $p$
   $\mathbf{A}$ is a $N$ by $N$ matrix
4. Observation (Emission) probabilities $\mathbf{B} = [b_j(m)]$ *where* $b_j(m) = P(O_p = v_m | q_p = S_j)$
   $O_t$ is the current (observed) amino acid at position $p$

**B** is a $N$ by $M$ matrix

5. Initial state probabilities $\Pi = [\pi_i]$ *where* $\pi_i = P(q_1 = S_i)$

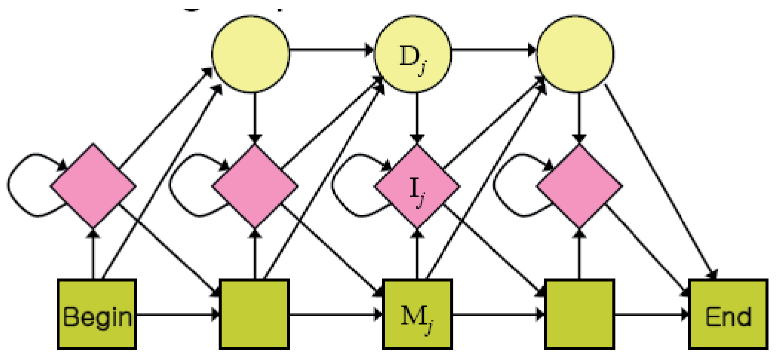


**Figure 3.1.** The transition structure of a profile HMM. Three types of states (insertion, deletion, and matched) are shown in diamonds, circles and squares respectively. (This figure is obtained from [49])

For profiling sequence motifs using HMM, the initial state probabilities are always zeroes except for the start state. Furthermore, only matched state has emission probabilities, i.e. all the insertion and deletion states have zero observation probability. Since $S$ *and* $V$ are implicitly restricted by other the probability matrices. The parameter needs to be determined is $\lambda = (\mathbf{A}, \mathbf{B})$. To calculate these parameters, we can use Baum-Welch algorithm, which is an Expectation-Maximization (EM) algorithm. EM algorithm is a two-step process. It starts with some random parameter values. In the E-step, the probabilities of the input sequences given the parameters are computed. In the M-step, the parameters are updated based on the outcome from the E-step. E-step and M-step will alternate until likelihood converges.

After the HMM is trained, we can align the sequence with its hidden states using the Viterbi algorithm. Viterbi algorithm is a dynamic programming method which considers all possible paths of the HMM and decides the best path based on the conditional probabilities. See [2] for the detail implementations of the Viterbi and Baum-Welch algorithms.

EM algorithms may also be used to build the Mixture model for functional annotation. Given the length of the desired motif as an extra input, we first guess the position of the motif for each sequences randomly. In the E-step, the proposed motif locations are used to estimate the probabilities of the bases appeared in the actives sites (a matrix that records the frequencies of the amino acid from each column and the background frequencies after alignment). In the M-step, the probabilities of the motif started at each position in the sequences are computed based on the results from E-step. The locations with the largest probability will be selected as new proposed motif sites so that new probability matrix can be generated again in E-step. These two steps perform consecutively until the motif converges and no longer changes. MEME [5] method uses this idea to build up its motif discovery algorithm.

Researchers also make use of machine learning techniques to annotate proteins. For example, Artificial Neural Network (ANN) is chosen to recognize sequence motifs in [55].

## 3.2 Structure-based annotation

In this section, we will examine annotation problems that are founded on structural motifs. For other possible non-motif related methods, please refer to [49]. Structural-based annotation methods can be grouped into three subproblems. First subproblem is how spatial and sequential information can be incorporated into a motif representation. Second subproblem is how to recognize common substructures where the motif pattern may or may not be given in advance. Third subproblem is how to increase the sensitivity and specificity of the motif such that the resulting motif can accurately represent the protein family. In this section, we will present some current developments for solving these problems.

### 3.2.1 Protein/Motif representation

They can be categorized into three main areas: 1) Point sets—represented by a set of points, which represent the 3D coordinates of the protein elements. These protein elements can be amino acids, specific atoms of residues, or even the centroids of proteins. Other chemical or biological attributes can be associated with these protein elements, such as charge, evolution information etc. 2) Point lists—represented by an ordered list of point set. The order of the point list should follow the primary sequence of the protein. 3) Graphs—represented by labeled graphs. A label graph contains nodes, representing the protein elements, and edges indicating the connections between two protein elements. This connection can be chemical bond, actual physical contact, or other physico-chemical interactions. Other possible representations can be seen in [22].

All of these representations have their advantages and disadvantages. Although complicated representations can encapsulate detail information about the proteins/motifs, the structure comparison algorithms will take longer to process the inputs. Moreover, complicated representations are vulnerable to noise, meaning that measurement errors would greatly reduce the quality of the representation. On the other hand, simple representations lose valuable structural information. After balancing the trade-offs, the structures of proteins and the initial motifs will be shown as labeled graphs. And the motif refinement process will be formalized as a graph problem. Detail definition of labeled graph used in this paper is listed in Chapter 2.

### 3.2.2 Structure comparison

Depends on the nature of comparisons, we can divide it into two categories: 1)pattern discovery—No pattern (or motif) is given in prior. Instead, the algorithm needs to find structure patterns that appear in all or most of the protein structures from a dataset. A number of sequence-independent and sequence-dependent methods have been developed, mainly differs in their structural representations and searching mechanisms. For example, TRILOGY [12] uses ordered lists of residues and their relative locations among residues to represent a structure, and its matching process involves gluing multiple triplets together from initial pattern discovery phase, PINTS [54], on the other hand, utilizes graphs for protein structures. Each node records the $C_\alpha$ and $C_\beta$ coordinates of an amino acid, while each bond measures the Euclidean distance between a pair of atoms. A depth-first method will try all possible sub-patterns of a structure. A pattern will match to a structure if the corresponding nodes are identical and their distance vectors fall between a predefined ranges. FFSM [35] also uses edge-based depth-first search to discover frequent patterns. However, it uses an unique graph representation known as CAM code, which is derived from the adjacency matrix of the protein graph structure. A pattern matches with a graph implies that the CAM of the pattern is a submatrix of the CAM of the graph (embedding). Two operations: CAM-join and CAM-extension are needed to generate the candidate patterns from the CAM search tree. To conclude, our algorithm is a complement of pattern discovery problem in the sense that the discovered motifs are further improved to obtain more remote homolog structures.

2) Pattern matching—Given a pattern (or motif), the algorithm determines if it appears in a protein structure. The category can be further divided into occurrence and probabilistic pattern matching. Several methods are designed for comparing protein structure, such as ASSAM [4], TESS [58], JESS [6] and Geometric Hashing [48], etc. Most of these approaches are occurrence matching methods, which report some or all the occurrences of a protein structure. In ASSAM, the occurrences between a protein and a motif are determined by subgraph matching. Their structures are represented as labeled graphs, but unlike our labeled graph definition, each residue has two nodes: the 'start' atom and the 'end' atom, which denoted as $(s, e)$. This atom pairs symbolize the location of the residue and the direction of its corresponding side chain. The pattern matches to a subset of a protein if there is one-to-one mapping between the amino acid nodes, and the distance vector differences are smaller than pre-defined thresholds. The distance vector between two resides $(s_1, e_1)$ and $(s_2, e_2)$ contains four numbers: $d(s_1, s_2), d(s_1, e_2), d(e_1, s_2), d(e_1, e_2)$ where $d$ is the Euclidean distance between two atoms. In geometric hashing based method TESS, the structures of proteins and patterns are represented using point set. To find out if the pattern occurs in a protein, the program maps all the coordinates of the nodes to a hash table. By counting the number of matched entries in the hash table, we can discover the largest set of coinciding points between the pattern and protein. Basically, it is a two-step process. In the preprocessing step, the program stores the patterns points from all possible reference frames into a hash table. A reference frame fixes the relative position of the pattern so that the coordinates can be calculated. Normally, two points in a pattern (assume that the pattern is ex-

pressed in 2D) can define a reference frame. The computation of the coordinates relative to the frame system requires geometric transformations such as rotations and translations. In the recognition step, a protein with a chosen frame will be mapped the same hash table obtained from the preprocess step. If there exists a pattern under certain reference frame which matches with one of the protein's entries, then the pattern occurs in that protein. Distinct from the occurrence pattern matching methods mentioned above, our algorithm utilizes statistical model to perform probabilistic pattern matching such that the probability of a motif appears in a protein structure is calculated.

### 3.2.3 Motif refinement/filtering

To further improve the sensitivity and specificity of an existing motif, researchers investigate the problem in two directions: 1) They introduce domain constraints to better identify functional homologs. For example, Geometric Sieving [13] compares the Least Root Mean Squared Distance (LRMSD) distributions between a candidate motif and an external protein set in order to select the optimal motif structure. LRMSD is a similarity measure between two point list representations (the candidate motif and a protein from the data set). The assumption behind Geometric Sieving is that an optimized motif should demonstrate the maximal geometric and chemical differences to all known protein structures. As a result, the researchers first generate a candidate motif set by considering all possible subsets of the original motif, and pick a candidate motif with the highest median LRMSD distribution as the refined motif. Cavity-aware motifs [17] combine structure motifs with a set of spheres known as C-spheres to imitate the protein's active site and its surrounding space for chemical reactions. Other information like structure energy level [1], electric charge or hydrophobicity can also be used for motif improvement. Mauer-Stroh *et al.* [44] refine the motif of carboxy-terminal enzymes by comparing the upstream linker region of the CaaX box. They notice that the compositions of this linker region often includes small or flexible hydrophilic amino acids. As a result, scoring functions are designed based on the differences of their linker profiles. W-AlignACE [18] also tries to improve the accuracy of the sequence motifs by utilizing information from the gene expression like Chip-chip data. The method assumes regions that cause huge change in expression are likely to be motifs. While calculating probability values in position weight matrices, the method puts more weight to the sequences with drastic fold of expression variations in order to achieve better motifs. 2) In data mining community, some studies start focusing on summarizing patterns using statistical models or machine learning algorithms. Yan *et al.* [63] construct pattern profile on a frequent itemsets based on Bernoulli distributions with clustering techniques. Wang *et al.* [59] reduce the number of frequent itemset patterns by building a model at each level iteratively. In protein analysis, Berger *et al.* [7] implement another iterative method that uses randomness and statistical techniques to improve the motif recognition on coiled coils proteins. Shah *et al.* [53] and Xiao *et al.* [62], on the other hand, use iterative method on top of multiple classifiers. They both tackle the imbalanced dataset problem by training the classifiers with unlabeled data. This unknown data is selected by the closeness of the positive/negiative data points (assuming that the classification results remain unchanged in areas of dense examples). During the re-training process at each iteration, the classifiers should become more accurate and more proteins will be annotated correctly.

# Chapter 4

# Algorithm Design

In this chapter, we will first introduce our structural motif refinement algorithm, which utilizes MRF to represent an optimized motif. Next, we will present two filter approaches to further improve the motif accuracy. The first filter makes use of the amino acid frequencies around the active site to identify the remote homologs. The second filter enlarges the original structural motif to nearby residues to increase its discriminative power. Also, results from different extended motifs can be combined to obtain better annotation quality. Figure 4.1 shows a high-level block diagram about the methods we proposed and their expected inputs and outputs. All of our methods are originated from approximate matching method with the initial motif (see sec 4.1.1). Note that these two filters can be incorporated into our motif refinement algorithm as well as being applied to the annotation results alone. In this paper, we combined the motif refinement algorithm and the environment filter for performance testing.

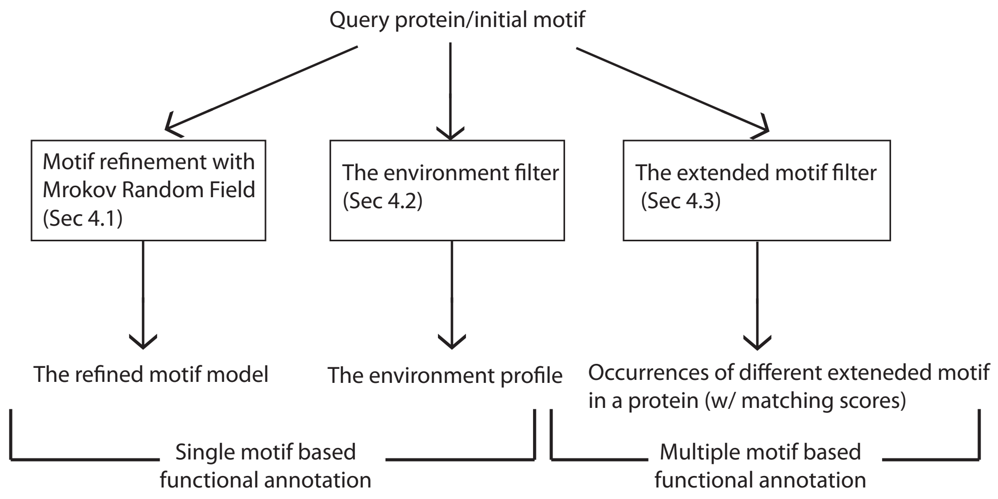

Query protein/initial motif
Motif refinement with Mrokov Random Field (Sec 4.1)
The environment filter (Sec 4.2)
The extended motif filter (Sec 4.3)
The refined motif model
The environment profile
Occurrences of different exteneded motif in a protein (w/ matching scores)
Single motif based functional annotation
Multiple motif based functional annotation

**Figure 4.1.** Block diagram of the algorithm design. Three proposed approaches are introduced. Note that outputs of the extended motif filter will be combined by the voting method and the feature vector method. The detail information of each method can be seen in corresponding sections indicated in each block.

## 4.1 Motif refinement with Markov Random Field motif model

Our algorithm takes a motif from a protein(known as initial motif) and a protein database as input. Our goal is to model the motif by a statistical model. The algorithm outputs a new model which can better describe the remote homologs matched from the database by repeatedly improving and verifying the initial motif. In other words, it is an optimized representation of the motif. Note that although this algorithm can refine or modify a given motif, this method cannot discover any new motif from the database. The initial (and unrefined) motif is either determined by other literature, or manually annotated from the domain experts. Nevertheless, in the section 6, we will propose an automatic approach to discover and refine motifs from a set of proteins (by using graph mining method) without human intervention.

Figure (4.2) shows an overview of our algorithm. In general, this algorithm can be divided into two stages:

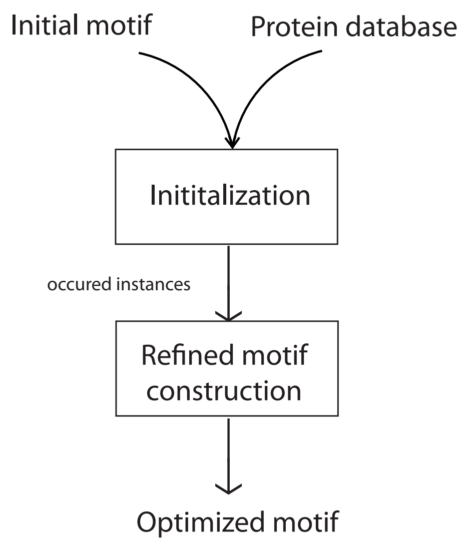


**Figure 4.2.** Overview of motif refinement algorithm. See Figure 4.3 for a detail break-down of this flowchart.

In the *initialization stage*, we first convert the inputs (i.e. the protein structures and the motif) into our graph representations. After the graphs are constructed, we match the initial motif to the protein database with certain criteria (see section 4.1.1). Proteins that are considered as matches with this motif will be passed onto the refined motif construction stage.

Given a set of matched proteins from initialization stage, the *Refined motif construction stage* first identifies all the instances from those proteins. An instance is a substructure of a protein which aligns with the motif. By 'studying' information about these instances, a new motif model is built. Specifically, a new motif that contains Markov Random Fields is constructed. This is called model building step. The resulting motif will be an improved version of the initial motif.

To further enhance the quality of the motif model, this new motif acquired from the model building step will match to the protein database again in order to collect a new set of instances. This new set of instances triggers the algorithm to go back to the refined motif construction stage again. This step is known as re-matching step. In refined motif construction stage, re-matching step and model building step will run iteratively until either 1) the number of proteins captured or 2) the number of the instances converges.

In general, our method is very similar to the well-known PSI-BLAST algorithm [3] in the sense that they both employ iterative approach to get its motif model. However, compare with PSI-BLAST, our method not only takes the protein structure into consideration, but also removes the assumption that each position is independent with each other, which is an underlying assumption for the position weight matrices. For the Geometric Sieving (GS), despite taking both structural and chemical information into account during refinement process, its motif model assumes equal distributions for all the node labels. Also, it only searches through subgraphs of the initial motif as the solution space. As a result, our model is superior to PSSM and GS. In the following sections, we will discuss our algorithm in detail.

### 4.1.1 Initial graph matching (also called approximate graph matching)

After the motif and proteins are represented using our labeled graph definition mentioned in section 2, we need to establish a mechanism for graph matching. In particular, we want to determine whether a motif graph $M$ *occurs* in a graph $G$ in a flexible manner while still be able maintain the relevance to the data set. In order to achieve this, we first introduce the idea of the subgraph isomorphism. Traditionally, a motif graph $M$ occurs in a graph $G$ if there exists a 1-1 mapping ($f : V_M \rightarrow V_{G'}\ where\ V_{G'} \subseteq V_G$) between $M$ and $G$ such that $M$ is a subgraph of $G$. However, this definition is too restrictive and it is not suitable for our use. Hence, we introduce a scoring matrix to the node mapping to quantify the degree of the similarity between two graphs. Formally speaking,

**Definition 2** ***(Initial Matching)*** *Graph* $G = (V, E, \Sigma_V, \Sigma_E, \lambda)$ *is* subgraph isomorphic *to* $G' = (V', E', \Sigma_{V'}, \Sigma_{E'}, \lambda')$ *if there exists a 1-1 mapping* $f : V \to V'$ *such that ,*

$$\sum_{u \in V} S(\lambda(u), \lambda'(f(u))) \geq T_1, and \tag{4.1}$$

$$dRMSD(E, E') = \sqrt{\frac{\sum_{(u,v)\in E} [\lambda(u,v) - \lambda'(f(u), f(v))]^2}{|V|(|V|-1)/2}} \leq T_1' \tag{4.2}$$

*where $S$ is a scoring function that penalizes a node label mismatch, $|V|$ is the size of the graph (i.e. total number of nodes), $T_1$ is a threshold for node label mismatch, and $T_1'$ is a threshold for structural differences.*

*Formula 4.2 is defined as distance root-mean-square deviation(dRMSD) between G and $G'$. It is a well-known standard for structural comparison [64]— Larger dRMSD means more diverse protein structures. In this paper, we set $T_1'$ to be* $0.8\AA$.

In practice, the function $S$ is usually provided by substitution matrices such as BLOSUM and PAM. In our experimental study section, we used BLOSUM62 [31] for our scoring function.

### 4.1.2 Refined motif construction

This process can be divided into two substages: Model building stage and Re-matching stage. The end products of this stage not only include the refined motif model, a new set of protein annotations will also be generated during this iterative process.

#### 4.1.2.1 Building refined motif

Our new motif model is defined as follows,

**Definition 3** ***(Pattern Statistical model)****Our new motif model is a triple* $(\Theta, \Sigma_E, \lambda)$*, where* $\Theta$ *is a Markov Random Field:* $\Theta(n) \rightarrow \Re^+$ *with* $n \in N$*.* $\Sigma_E$ *is a set of edge labels; and the* $\lambda$ *is a function that assigns the edge labels to the corresponding edges in the Markov Random Field graph.*

This model not only contains both labeled items and structure components, but also offers large (almost infinite) search space for our algorithm to optimize a motif. At the same time, our model enforces certain restrictions on the edges labels and nodes labels (potential functions) such that dependencies among neighboring elements can be preserved.

To estimate the potential functions of the maximal cliques in MRF, we apply Radim Jirousek's Iterative proportional fitting algorithm (IPF). Given a set of instances from the previous matching, IPF will try to modify the potential function for each clique $V(X_c)$ such that the marginal probability $p(X_c = x_c)$ equals to the maximum likelihood (ML) estimate (in this case ML is the empirical marginal). To summarize, it involves three iterative steps:

1. Initialization — assigns startup value of 1 to all the potential outputs, and calculates the normalization factor.

2. Marginal calculation — for each maximal clique $c$ in $C$, computes the marginal probability using the potential function with the following formula (at iteration t).

$$p^{(t)}(X_c = x_c) = \sum_{X,\ X_c=x_c} \left(\frac{1}{Z}\prod_{D\neq c} V_D(x_D)V_c^{(t)}(x_c)\right)$$

   where $C$ is the set of maximal cliques in the graph, $c \in C$, $x_c$ is one configuration of clique $c$

3. Proportional fitting — changes the potential values for c according to the quotient between the maximum likelihood and the marginal probability (at iteration t).

$$V_c^{(t+1)}(x_c) = V_c^{(t)}(x_c)\frac{P_{ML}(X_c = x_c)}{P^{(t)}(X_c = x_c)}$$

The last two steps will repeat until the algorithm converges (i.e. the maximum likelihood equals to the marginal probability). Figure 2.5 illustrates a sample instances obtained from the first stage, and the resulting MRF model after applying IPF. It has been proven in [51] that IPF always converges to a given ML estimation. Also, it guarantees to achieve the global maximum as it performs a coordinate ascent on the log-likelihood from the direction of each potential function. As a result, a refined MRF is generated for the next round of graph matching.

Because MRF requires clique identifications to set up the potential functions, we need to design a program which can identify all the maximal cliques from a given motif. This problem is also known as ‘maximal clique enumeration’ problem. This is a classical NP-hard graph problem and there are many exact and heuristic approaches available. For our solution, we simply do a depth first (brute force) search and go through all possible nodes with a particular order. Moreover, we keep track of the nodes we visited throughout the recursions. A list containing set of nodes stores a candidate of maximal clique. If no new node can be added to the list, a maximal clique is found. Although our program has a worse case complexity of $O(2^n)$ where n is the number of nodes, we justify that this is still feasible because most of the motifs are very short in length, and their structures are usually less complicated.

#### 4.1.2.2 Re-matching

In the *re-matching stage*, we generalize the matching criteria by utilizing our new motif model— Markov Random Fields for matching the node labels and edge labels for matching edges. The following conditions define if a motif model is subgraph isomorphic to a graph G,

**Definition 4** ***(Re-Matching)*** *The pattern model* $M\,(\Theta, \Sigma_E, \lambda)$ *matches with the graph* $G' = (V', E', \Sigma_{V'}, \Sigma_{E'}, \lambda')$ *if there exists a 1-1 mapping* $f$ *such that,*

$$\begin{aligned} Score_{node} &\geq T_2, and & (4.3)\\ dRMSD(\Theta_E, E') &\leq T_1' & (4.4)\\ for\ u_0 \ldots u_m \in V', && (4.5)\\ Score_{node} &= \Theta(\{f(u_0) \ldots f(u_m)\}) & (4.6) \end{aligned}$$

*where,*

*$T_2$ is the scoring threshold*

*$\Theta_E$ is set of edges in the MRF graph*

*dRMSD function is defined in formula 4.2*

*$T_1'$ is the threshold for structural difference, which is the same $T_1'$ in 4.1.1 (set to 0.8 Å in our experiments)*

To carry out this graph matching in both initialization (section 4.1.1) and rematching (section 4.1.2.2) stage, we employ J. R. Ullman's occurrence algorithm [57]. Regardless of numerous algorithms that deal with subgraph isomorphism problem, we pick this one due to its availability and the ability to adapt our flexible matching approach.

Ullman's subgraph isomorphism algorithm is originally designed for unlabeled graphs (neither node nor edge is labeled). It can find all the isomorphisms between a pattern graph $G_A$ and subgraphs of another graph $G_B$. Both graphs are represented by their adjacency matrices $A(m, m)$ and $B(n, n)$, given that the size of graph $G_A$ is $m$ and the size of graph $G_B$ is $n$ where $m \leq n$. Adjacency matrix indicates the existence of an edge between any two nodes. For instance, if there is an edge between node $i$ and $j$ in $G_A$, then $a(i, j) = 1$. Otherwise, $a(i, j) = 0$. The isomorphism algorithm then constructs another binary matrix $M$ which has $m$ rows and $n$ columns. For each configuration of $M$, each row contains exactly one 1 and no column contains more than one 1. These restrictions (called row/column restrictions) of $M$ make sure that there is always an one-to-one mapping between $G_A$ and $G_B$. To report a subgraph isomorphism between $G_A$ and $G_B$, we just need to identify all the 1's in the matrix $M$. For example, node $i$ in $G_A$ maps to node $j$ in $G_B$ if $m(i, j) = 1$.

Ullman's subgraph isomorphism algorithm uses depth first traversal and backtracking to enumerate all possible configurations of $M$. Starts from the first column, it recursively flips one 0 to 1 at each column, as long as it does not violate row/column restrictions mentioned before. In our application, because the graphs are labeled, extra criteria (see Def 2 and Def 4 for the rules) will be added along with the row/column restrictions. When all rows have exactly one 1 in $M$ (i.e. it reaches the base case), one subgraph isomorphism is found.

In terms of linear algebra, a matrix $C$ is defined as $C = M(MB)^T$ where T is the transposition of the matrix. If

$$\forall(i, j : 1 \leq i, j \leq m, a(i, j) = 1 \Longrightarrow c(i, j) = 1)$$

then $M$ shows an subgraph isomorphism between $G_A$ and $G_B$. For more detail about the proof and implementation of this algorithm, please refer to [57].

From the machine learning point of view, we investigate the motif refinement problem in a semi-supervised approach. In our algorithm, we assume the existence of a graph database containing a large set of protein graphs. We further assume that only portion of the graph database has class labels (proteins captured from the Initial graph matching stage) , while the majority of the graphs have no label (the rest of the dataset). Our semi-supervised method will take this labeled data, together with some of the unlabeled data (by iteratively scanning the database), to build a statistical motif model which can better summarize the remote homolog proteins.

### 4.1.3 Computational complexity

Figure 4.3 shows a detail step-by-step breakdown of the motif refinement algorithm. There are two main parameters needed to be set to ensure our refinement algorithm works effectively:

- The initial score threshold ($T_1$), a protein matches with the initial motif if the score is larger than the threshold.
- The model score threshold ($T_2$), a protein matches with the refined motif model if the score is larger than the threshold.

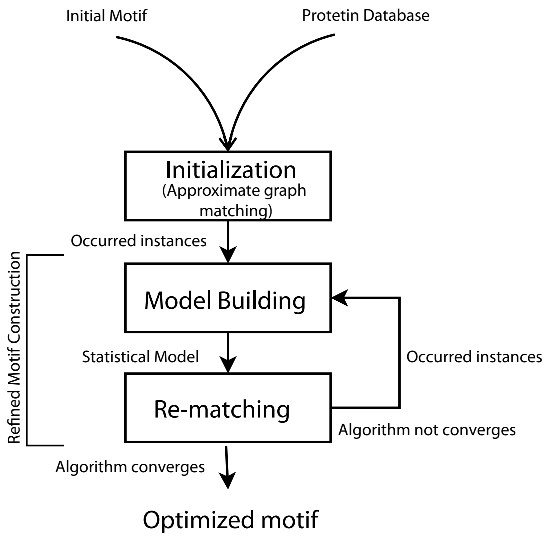


**Figure 4.3.** Flowchart of motif refinement algorithm.

So far determining these parameter values is a difficult task and it may require an extensive search to discover the optimal solution. However, it seems like these two parameters are interrelated. After performing a number of experiments, we found out that adjusting one parameter may have an equivalent effect on adjusting another parameter. For example, both decreasing the initial score threshold and the model threshold can reduce the number of instances matched to a motif. A detail investigation is needed in the future research.

Two NP-complete problems are addressed during the discussion, including clique enumeration problem, and subgraph isomorphism problem. Therefore, it is not surprising that our algorithm is also a NP-complete problem. However, due to the real-world constraint such as chemical properties and limited motif size, the complexity of the graph topology is restricted, thus allowing us to refine motifs within a reasonable amount of time. Moreover, the graph matching steps during the initialization and re-matching stage can be parallelized to speed up the matching process.

## 4.2 The environment filter

The environment filter assumes that the local environment around the active site is a determining factor for protein functions. If the surroundings of a probable motif location is very different from other proteins in the same family, then this site may not be functional, and thus having different functions. The environment filter works in two stages. In the first stage, a profile is generated for a particular protein family, which is known as the environment profile. In the second stage, the filter uses the environment profile to eliminate the false positives from the pattern matching results. The environment profile is defined as follows,

**Definition 5** *(**The environment profile**) The environment profile $P$ is an ordered list of 20 triples $[(a_1, \mu_1, \sigma_1), (a_2, \mu_2, \sigma_2), ..., (a_{20}, \mu_{20}, \sigma_{20})]$ where each element represents one amino acid, $a_i$ is the amino acid identifier $i$, $\mu_i$ is the mean frequency of amino acid $i$, and $\sigma_i$ is the standard deviation of frequency in amino acid $i$.*

To generate the environment profile in the first stage, it requires a set of proteins (protein family set) with the following requirements: 1) They should contain the same motif as the initial motif. In other words, those proteins should have the same function. 2) The location of the initial motif is known in prior. For each protein in the protein family set, it first collects the neighboring residues around the active site. The neighbors of a motif are described as follows,

**Definition 6** ***(Neighbors of a motif)****A node $v$ is considered as the neighbor of motif graph $G' = (V', E', \Sigma_{V'}, \Sigma_{E'}, \lambda')$ which is resided inside protein structural graph $G = (V, E, \Sigma_V, \Sigma_E, \lambda)$ if it satisfies the following conditions:*

$$v \in V \ and, v \notin V',$$
$$\exists w \in V' \ such \ that \ (v, w) \in E \ and \ \lambda(v, w) \leq 8.5$$

Simply speaking, all the nodes that are connected to the motif nodes with edge distance less than $8.5\mathring{A}$ are treated as its neighbors. The normalized frequency distribution is then computed, results in a tuple of twenty numbers. Fig 4.4 shows this frequency calculation with a sample graph. Finally, when the frequency distributions of protein family set are gathered, we can calculate the environment profile,

Given the normalized frequency distributions for $N$ proteins in the protein family set: $(d_{1,1}, ...d_{20,1}), (d_{1,2}, ...d_{20,2}), ..., (d_{1,N}, ...d_{20,N})$, the environment profile $P = [(a_1, \mu_1, \sigma_1), (a_2, \mu_2, \sigma_2), ..., (a_{20}, \mu_{20}, \sigma_{20})]$ is computed by the following formula,

$$\mu_i = \frac{\sum_{j=1}^{N} d_{i,j}}{N-1} \qquad \sigma_i = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}(d_{i,j} - \mu_i)}$$

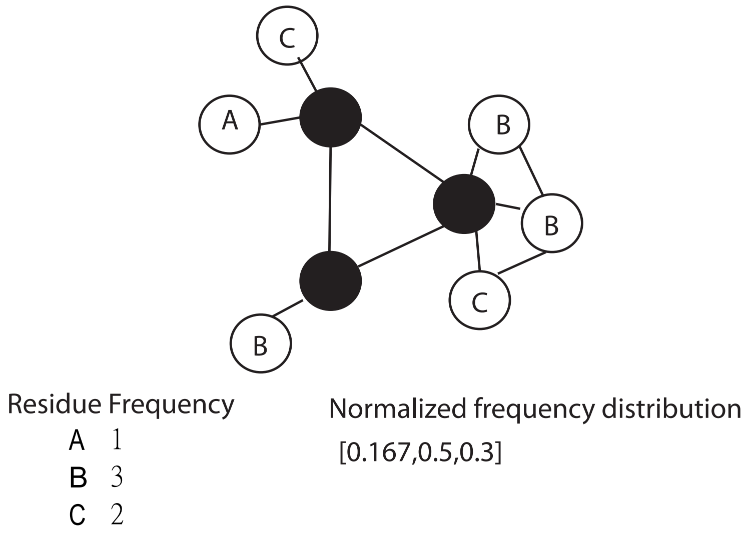


**Figure 4.4.** Calculating the neighbor frequency of a motif. The black nodes are the motif, and the white nodes are the neighbors of the motif. Other nodes that are neither motif nor its neighbors are omitted. In addition, the edge labels are not shown for simplicity. Assume that the node label domain only contains {A,B,C}, the neighbor frequency and its normalized distribution are shown here. To calculate the environment profile, we need to get the normalized frequency distributions for each protein in the protein family set. Then, we measure the mean and standard deviation of every amino acid frequency distribution.

where $i = \{1, 2, 3...20\}$

In the second stage, we apply the environment profile to filter the pattern matching results. In particular, a matched protein will be filtered by the environment profile $P = [(a_1, \mu_1, \sigma_1), (a_2, \mu_2, \sigma_2), ..., (a_{20}, \mu_{20}, \sigma_{20})]$ given that the normalized frequency distribution of that protein is $(d_1, ...d_{20})$, and,

$$\sum_{i=1}^{20} \frac{|d_i - \mu_i|}{\sigma_i} \geq T_3$$

where $T_3$ is called the filter threshold, which is an adjustable value for strictness of the filter (the larger the threshold, the looser the filter becomes).

In general, the environment profile can be applied to any pattern matching methods as long as their matching results include the active site locations. Actually, in our motif refinement algorithm, our approximate matching used in the first (initial graph matching) and the last (re-matching) stage (see section 4) can apply the environment filter to further refine the results. Therefore, we will incorporate them together to do the functional annotation in the experiment section.

## 4.3 The extended motif filter

Similar to the environment filter, the extended node filter also employs the neighbors of the active site. It embraces one of the neighboring nodes into the initial motif, thus enlarging the motif size by one. The definition of the motif neighbors is identical to Definition 6, and the extra node is chosen randomly in our experiment. Nonetheless, random selections of the nodes are not required. With domain experts, people can pick up some useful residues which may not close to the motif. To use this filter, we will apply the pattern matching method (e.g. our approximate graph matching algorithm) with this extended motif. The rationale behind this filter is that larger motifs often prevent random matches, so it may reduce the false positives.

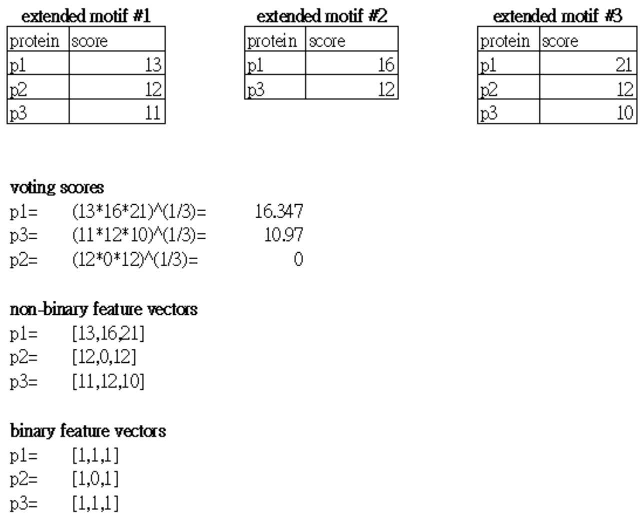
To further improve the quality of the results, we consult multiple extended motifs to obtain different filtered results. In this study, we try two different ensemble techniques in machine learning to combine their results: the voting method and the feature vector method.

Voting method: Given a set of extended motifs which is enlarged by different neighboring residues, we first apply our approximate graph matching method (see section 2) individually. The matching proteins results, along with their mismatch scores (as defined by formula 4.1), will be averaged by their geometric mean,

$$v_p = (\prod_{i=1}^{n} s_{i,p})^{1/n}$$

where $v_p$ is the voting score (averaged score) for protein $p$, $n$ is total number of extended motifs, and $s_{i,p}$ is the mismatch score for protein $p$ using extended motif $i$.

The matched proteins will be sorted according to the averaged voting scores. An extra parameter $T_4$ will be used to determine the number of top-scored results to pass the filter. Figure 4.5 illustrates the voting methods based on three extended motifs.



**extended motif #1**

| protein | score |
|---|---|
| p1 | 13 |
| p2 | 12 |
| p3 | 11 |

**extended motif #2**

| protein | score |
|---|---|
| p1 | 16 |
| p3 | 12 |

**extended motif #3**

| protein | score |
|---|---|
| p1 | 21 |
| p2 | 12 |
| p3 | 10 |

**voting scores**

|  |  |  |
|---|---|---|
| p1= | (13*16*21)^(1/3)= | 16.347 |
| p3= | (11*12*10)^(1/3)= | 10.97 |
| p2= | (12*0*12)^(1/3)= | 0 |

**non-binary feature vectors**

|  |  |
|---|---|
| p1= | [13,16,21] |
| p2= | [12,0,12] |
| p3= | [11,12,10] |

**binary feature vectors**

|  |  |
|---|---|
| p1= | [1,1,1] |
| p2= | [1,0,1] |
| p3= | [1,1,1] |

**Figure 4.5.** Examples of voting method and feature vector method using extended motifs. Three tables (on the top) indicate the pattern matched results from three extended motifs. Each table has the protein name and its mismatch score. The geometric means of the results are sorted and listed as voting results. The non-binary feature vectors rearrange the scores into an ordered list, whereas binary feature vectors only consider if that protein appears in the matching result or not. Note that extended motif # 2 does not match with p2, so its mismatch score is equal to zero.

Feature vector method: Given a set of extended motifs, we first apply our approximate graph matching method. Next, for every protein in the dataset, we construct a feature vector of length $n$ , which is the total number of extended motifs. The feature values will be the same as the scores they obtained from the corresponding motif. If the matching result does not contain that protein, that protein will have a zero for its feature value. After the feature vectors for all the proteins are constructed, machine learning approaches can be utilized to study underlying patterns of the features. Specifically, we use support vector machine (SVM) to do the job. Since this is a supervised learning problem, we supply the SVM with a training set, which is another set of proteins with known annotations (either the protein is in that family or not). The trained SVM can annotate new proteins with feature vectors as inputs. See the experiment section on how we collect the training and the testing samples for the SVM.

During preliminary testing, we tested the SVM with both binary feature vectors (1 when the protein appeared in the matching results, 0 otherwise) and non-binary feature vectors(use the mismatch scores as feature values). The non-binary feature vectors performed much better than the binary ones (about 30% precision improvement with the same recall level). As a result, all the experiments conduced in this paper use non-binary feature for the extended motif filter. Figure 4.5 illustrates an example of the feature vector method.

# Chapter 5

# Experimental Study

Each of the filter and motif model proposed in this paper undergoes a series of tests from the real-world protein data. In particular, five enzyme families will be used for functional annotation. The goal of this study is to evaluate the ability of distinguishing remote homologs from a set of functional-unrelated proteins given a query protein with its motif. To achieve this goal, two performance metrics are employed to measure the effectiveness of each experiment:

1) The number of proteins captured from the data set which falls under the same enzyme family as the query protein, i.e. the number of the true positives($TP$).

2) The number of proteins captured from the data set which should not be considered as the same enzyme family as the query protein, i.e. the number of the false positives ($FP$). This may include non-enzyme proteins and enzymes that have different or unknown functions.

On the contrary, metrics that evaluate the negative samples such as true negatives ($TN$, the number of functional-unrelated proteins that does not being tagged as remote homologs) or accuracy ($accuracy = [TP+TN]/[TP+FP+TN+FN]$)

will not be our focus of this experiment. This is because the dataset is highly unbalance— In the real-world protein database, the ratio between remote homologs from a protein family and the unrelated proteins is very small. Consequently, a method can obtain very high accuracy and high TNs even though it does not capture any proteins. The meaningful information comes from the true predicted ones since their functions can be associated with their family members. Based on the $TPs$ and $FPs$ collected from each method, we can compare the performance trade-off using receiver operating characteristic (ROC) analysis. ROC analysis is a popular evaluation tool in machine learning and data mining community [11, 36]. The relation between the true positive rate and false positive rate is plotted by varying the discrimination threshold, the resulting graph is known as a ROC curve . The area under the ROC curve (AUC) can quantify its annotation ability into a real number. Research has shown that AUC provides better measure than accuracy [20].

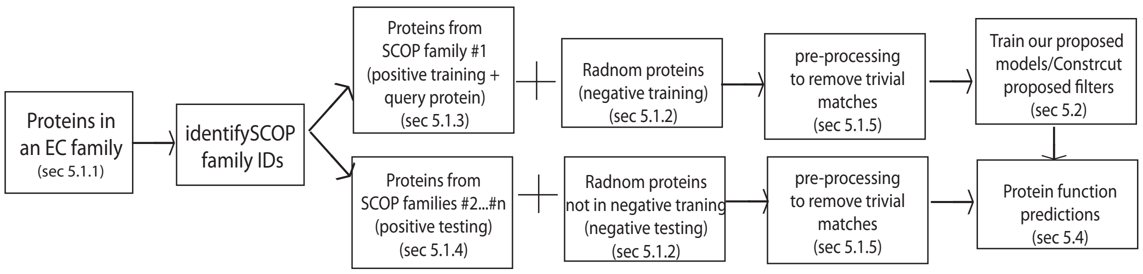

Proteins in an EC family (sec 5.1.1)
identifySCOP family IDs
Proteins from SCOP family #1 (positive training + query protein) (sec 5.1.3)
Radnom proteins (negative training) (sec 5.1.2)
pre-processing to remove trivial matches (sec 5.1.5)
Train our proposed models/Constrcut proposed filters (sec 5.2)
Proteins from SCOP families #2...#n (positive testing) (sec 5.1.4)
Radnom proteins not in negative traning (negative testing) (sec 5.1.2)
pre-processing to remove trivial matches (sec 5.1.5)
Protein function predictions (sec 5.4)

**Figure 5.1.** Block diagram of the experiment. Detail information can be seen in corresponding sections indicated in each block. This diagram will repeat for the other EC families.

Figure 5.1 provides an overview of these experiments. In the following sections, data collections and implementation specifics will be listed in detail. In addition, results from different experimental setups and their implications will also be discussed.

## 5.1 Data collection

All 3D coordinate information of the proteins and motifs in this study is obtained from the Protein Data Bank [1] (PDB). The SCOP database (version 1.71)[2] provides information about the protein structures. Since our motif model requires a query protein (the active site of the query protein is the initial motif) and a protein dataset as inputs, we will talk about how we gather this training data in detail in the upcoming sections. For the sake of performance evaluation, all of our dataset (training/testing) consists of two parts—*Remote homologs* contain all the positive samples, and *random proteins* consist of negative samples. Both parts need to go through some pre-processing steps to ensure the validity of the data, and it will be mentioned in sections 5.1.1 and 5.1.2.

### 5.1.1 Protein families selection for positive samples

For gathering training and testing dataset, it is necessary to setup a fixed benchmark so that positive and negative samples can be identified. For this reason, we decide to use the functional classification methods from the Enzyme Commission(EC) [61]. Enzymes are subsets of proteins which assist chemical reactions within an organism. And depend on the chemical reactions every enzyme catalyzes, it is grouped into a four-level hierarchical code, known as an EC number. Each level is represented by an integer. Lower level code in an EC number describes a finer level of chemical reaction category. In this paper, we assume that all functional annotations defined by the EC in ENZYME[3] database (version 11/2008) are correct, and two enzymes are considered as functionally-related if

[1]http://www.rcsb.org/pdb/home/home.do
[2]http://scop.mrc-lmb.cam.ac.uk/scop/
[3]http://ca.expasy.org/enzyme/

their **first three** levels of the EC numbers matched. For example, two proteins with PDB IDs 2LPR (EC number: 3.4.21.12) and 1HJA (EC number: 3.4.21.1) are related because they both involve in breaking down peptide bonds when interacting with water molecules (these proteins are known as Hydrolases). Moreover, they both contain a serine residue in their active sites (these proteins are known as serine peptidases). Therefore, their functions should be the same as other enzymes in serine peptidases family (EC 3.4.21)—blood clotting, complement activation, and digestion etc. [26]. In fact, the forth level of the EC number usually specifies the substrate of the reaction, which is not closely related to our motif model as we only focus on the structure of the active sites [24]. So, ignoring the last level in this experiment is a logical decision.

As mentioned above, we will test our approaches with five different enzyme families. Table 5.3 shows their EC numbers and the total number of proteins in each enzyme family. To gather the positive members from those families, we utilize the list provided by PDBSProtEC [4] mapping [42].

Note that since EC allows partial assignments, some enzymes may not receive full four-level EC numbers. Also, according to the documentation in the PDB-SProtEC list, three types of proteins are recorded: enzymes, non-enzyme proteins, and 'unknown' proteins (proteins that may/may not be enzymes). In this experiment, proteins will not be counted as positive samples if they are 'unknown' proteins or they do not have complete EC numbers up to third level.

Choosing only enzyme families for functional annotation is only for the ease of performance evaluations, our approaches can actually work on other types of annotation problems other than enzymatic reactions such as recognizing DNA

[4] Can be downloaded at http://www.bioinf.org.uk/pdbsprotec/

binding proteins.

### 5.1.2 Random proteins selection for negative samples

In order to introduce false samples, we randomly add functionally-unrelated proteins to our dataset. Proteins are considered functionally-unrelated if they do not come from any of those five EC families.

### 5.1.3 Training data construction

For each EC family, we first retrieve the 3D structures of positive samples (as defined in 5.1.1). Then, we manually collect one protein and treat it as the query protein. We search through the literature database like PDB, PubMed [5]or catalytic residue database Catalytic Site Atlas (CSA) [50] to obtain the active center location of the query protein, which would be the initial motif. To obtain the positive training samples, we retrieve the structure classification (i.e. the SCOP family ID) of the query protein, and collect positive samples with the same SCOP family ID. The size of negative training samples is the same as the positive training samples, and it consists of random proteins.

The selected query protein from each EC family, along with their active regions and their original sources are shown in table 5.1. The positive training samples, tagged with their SCOP family IDs, are listed in table 5.2. This training dataset will go through preprocessing step in section 5.1.5.

### 5.1.4 Testing data construction

For each EC family, positive samples that have different SCOP family IDs as the query protein would be considered as positive testing samples. About 1000

[5]http://www.ncbi.nlm.nih.gov/sites/entrez

| EC number | Query protein | Initial motif | Source |
|---|---|---|---|
| 3.4.21 | 1mcta | HIS57-GLY193-SER195 | CSA |
| 3.4.22 | 1pppa | CYS25-HIS159-ASN175 | CSA |
| 6.3.2 | 2dlna | GLU15-SER150-GLY276 | Fan *et al.* [23] |
| 1.1.1 | 7mdha | ASP201-ARG204-HIS229 | CSA |
| FAD binding (1.8.1 + 1.18.1) | 1q1ra | GLY11-GLY13-GLY16-ALA20 | Hanukonglu *et al.* [28] |

**Table 5.1.** Query proteins selected for each EC family. The *Initial motif* column shows the active sites of query proteins. The *source* column indicates where initial motifs are obtained (either from Catalytic Site Atlas (CSA) or from literature).

| EC num (SCOP family id) | PDB ids with chain number |
|---|---|
| 3.4.21 (50514) | 1ekb, 1elva, 1eq9a, 1fiwa, 1gvkb, 1gvza, 1m9ua, 1nn6a, 1pq7a, 2hlca |
| 3.2.22 (54002) | 1cv8_, 1cb5a, 1thea, 1ewla, 9papa, 1icfa, 1a6ra, 1cs8a |
| 6.3.2 (56602) | 2dlna, 1e4da, 1ehia, 1iova, 1iowa, 1glva, 1gsaa, 1gsha, 2glta |
| 1.1.1 (51848) | 1guza, 1hyea, 1hyha, 1mlda, 1o6za, 1ojua, 1t2da |
| FAD binding (51943) (1.8.1 + 1.18.1) | 1feca, 1onfa, 1trba, 3grsa, 1fl2a, 3lada |

**Table 5.2.** Proteins selected as positive training samples after preprocessing

random proteins are also added to the testing data to serve as negative testing samples. This testing dataset will go through preprocessing step in section 5.1.5.

### 5.1.5 Preprocessing of training and testing data

Both training and testing data will be filtered to exclude redundant structures from our analysis. First of all, we need make sure that there is no protein overlapping between the training and testing data. This action is to ensure no extra information is leaked during the evaluation process. All proteins with sequence identities $> 40\%$ will then be removed from the dataset. Finally, all the 'trivial' matches which can be done by sequence-based functional annotation method like PSI-Blast will also be removed. The resulting dataset after this preprocessing step should have low sequence identities and structural similarities — proteins that are considered to be in the 'twilight zone' [52].

Part of preprocessing step has already been done by the Protein Sequence Culling Server (PISCES) [60]. We download the pre-complied protein list provided by their server [6]. This list were then filtered by PSI-Blast results using query protein as input. Table 5.3 lists the number of samples in training and testing dataset after preprocessing.

| EC Family | Training | | Testing | |
|---|---|---|---|---|
| | # Positive | # Negative | # Positive | # Negative |
| 3.4.21 | 10 | 10 | 23 | 1000 |
| 3.4.22 | 8 | 8 | 16 | 1000 |
| 6.3.2 | 9 | 9 | 21 | 1000 |
| 1.1.1 | 7 | 7 | 13 | 1000 |
| FAD binding (1.8.1+1.18.1) | 6 | 6 | 14 | 1000 |

**Table 5.3.** Dataset statistics after preprocessing

## 5.2 Experiment procedures

The experiments will be conducted in the following order: 1) approximate matching with the initial motif for baseline comparison (no filter or refinement process applied) 2) approximate matching with environment filter 3) motif refinement algorithm with environment filter 4) voting with extended motif filter and 5) feature vectors method using extended motif filter and SVM. To select the optimal

[6]The list can be obtained from http://dunbrack.fccc.edu/PISCES.php , the parameters used in this list are: resolution=6.0, R factor=0.25, sequence similarity cutoff $< 40\%$.

parameters for the experiments, we use a heuristic approach, meaning that the best parameters attained in the previous experiments will be reused in the next experiment. The following sections detail the steps needed for each experiment. For the implementation of each approach, please refer to Chapter 4.

### 5.2.1 Approximate match with initial motif

In this experiment, we merely apply the active sites (i.e. the initial motifs) to the testing dataset using the approximate matching technique. For each enzyme family, we apply its corresponding initial motif and count the number of TPs and FPs from every matching result. The matching criteria are identical to the first stage of our motif refinement algorithm (see section 4.1.1). BLOSUM62 is used for the node mapping mismatch and dRMSD is used for for the edge mapping mismatch. To create a ROC curve, we vary the node label mismatch threshold ($T_1$) with increment of 2.

### 5.2.2 Approximate match with environment filter

In this experiment, we apply an environment filter on top of the approximate matching method. The environment profile is generated based on the protein family set from positive training samples. Because positive training samples have similar folding structures, we prevent cheated actions by leaking extra structural information to the environment filter. Table 5.2 lists the proteins used for the profile generation. The approximate matching results between the initial motif and the testing dataset will finally be filtered by the environment profile. The approximate score threshold is determined from the previous experiment. To generate the ROC curve, the environment filter threshold ($T_3$) is varied with increment of 5.

### 5.2.3 Motif refinement algorithm with the environment filter

In this experiment, we apply our motif refinement algorithm with the environment filter. The active site of the query protein from each enzyme family is the initial motif of our algorithm, and the protein database is the testing dataset we built for the experiment. Since the algorithm is semi-supervised, the entire refinement process will be operated on the testing data alone. The filter threshold ($T_3$) and the approximate score threshold ($T_1$) is determined from the previous experiments. To generate the ROC curve, the model threshold ($T_2$) is varied with increment of 2.5. For the termination condition our algorithm, the number of proteins obtained in the last two iterations have to be the same.

### 5.2.4 Voting method with extended motif filter

In this experiment, every initial motif derives four extended motifs. Each extended motif contains one additional node which is chosen randomly from the motif's neighbors in query protein. Table 5.4 shows all the resulting motifs and their specific nodes they enlarged. The approximate graph matching applies to these four extended motifs and produces four lists of proteins with their node mismatch (BLOSUM) scores. These scores are aggregated using geometric means and the proteins are re-ranked by their new voting scores. The approximate score threshold ($T_1$) is fixed to be the same as the parameter chosen in the first experiment. To generate the ROC curve, the voting threshold ($T_4$) is varied with increment of 10.

| EC number | Query protein | Extended motifs |
|---|---|---|
| 3.4.21 | 1mcta | **ASN95**-HIS57-GLY193-SER195 |
| | | **CYS42**-HIS57-GLY-193-SER195 |
| | | **CYS58**-HIS57-GLY-193-SER195 |
| | | **ILE212**-HIS57-GLY-193-SER195 |
| 3.4.22 | 1pppa | **ALA160**-CYS25-HIS159-ASN175 |
| | | **SER29**-CYS25-HIS159-ASN175 |
| | | **VAL157**-CYS25-HIS159-ASN175 |
| | | **ALA27**-CYS25-HIS159-ASN175 |
| 6.3.2 | 2dlna | **SER19**-GLU15-SER150-GLY276 |
| | | **HIS63**-GLU15-SER150-GLY276 |
| | | **LEU62**-GLU15-SER150-GLY276 |
| | | **THR278**-GLU15-SER150-GLY276 |
| 1.1.1 | 7mdha | **LEU200**-ASP201-ARG204-HIS229 |
| | | **VAL265**-ASP201-ARG204-HIS229 |
| | | **GLY227**-ASP201-ARG204-HIS229 |
| | | **ASN173**-ASP201-ARG204-HIS229 |
| FAD binding (1.8.1 + 1.18.1) | 1q1ra | **HIS43**-GLY11-GLY13-GLY16-ALA20 |
| | | **PRO42**-GLY11-GLY13-GLY16-ALA20 |
| | | **ALA38**-GLY11-GLY13-GLY16-ALA20 |
| | | **GLY111**-GLY11-GLY13-GLY16-ALA20 |

**Table 5.4.** The list of the extended motifs for different EC families, which are used for the voting method and the feature vector method. The bold residues indicate some additional nodes added to the initial motifs. Note that even though the additional nodes have the same type (e.g. both the second and third extended motifs in 3.4.21 include CYS residue), their edge labels can still be different. As a result, their approximate matching results will vary.

### 5.2.5 Feature vectors method with extended motif filter

In this experiment, the extended motifs from the previous experiment will be used to generate the feature vectors for the proteins in the dataset. Table 5.4 lists all the extended motifs and their configurations. The approximate score threshold ($T_1$) is fixed to be the same as threshold chosen in the first experiment. Since there are four extended motifs for each initial motif, the length of the feature vector will be four. Note that this is a binary classification problem— either a protein is from the same EC family or not, so we need to build a classifier for each protein family. Support vector machine (SVM) with the radial basis function (RBF) kernel is chosen to be the classifier. To select the suitable parameters ($C$ and $\gamma$) for the SVM, we will use 3-fold cross validation method. The entire training set is partitioned into three parts: one of them is used for training the SVM with particular parameters, the rest of them are used for performance evaluation (validation set). The validation set is rotated and the prediction errors are averaged. The trained SVM with the smallest prediction error will be used to predict protein functions for the testing dataset. To generate the ROC curve, we move the SVM decision boundary across the space to compute different TP and FP rates.

## 5.3 Implementation specifics

The Bioinformatics Cluster at ITTC is used to perform our experiments. It has total 128 nodes, 384 Intel Xeon processors and 640 GB of memory. Our core algorithms are implemented as a serial application using C++, while other peripheral programs such as graph structure generation and the driver scripts are written in Perl and Python. For the C++ compilation environment, we used gcc compiler with 03 optimization. The SVM implementation for the feature vectors method is from the LIBSVM package [16]. And for visualization of some proteins structure, we use a software package called VMD [37] with all default settings.

## 5.4 Experimental results

Figure 5.2 summarizes the performance differences of our five experiments using ROC analysis. The ROC graphs for the other four enzyme families are

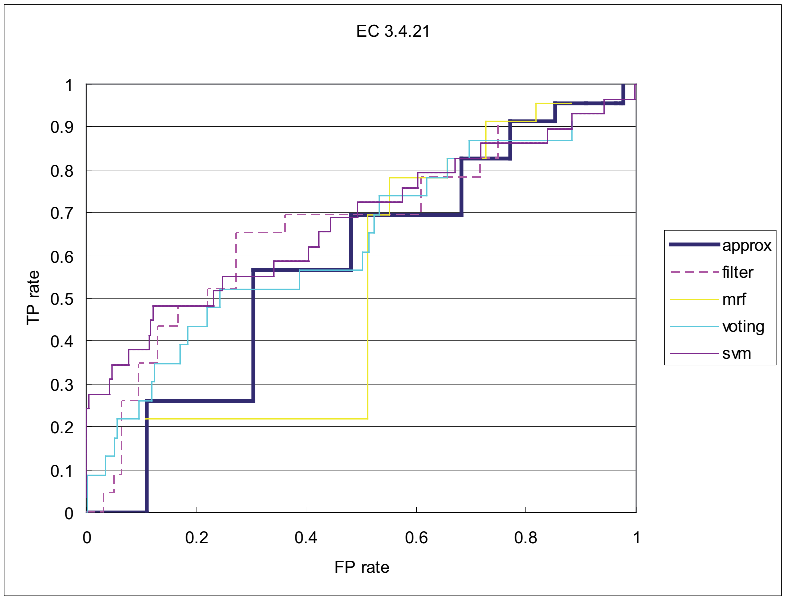


**Figure 5.2.** The ROC analysis of EC 3.4.21 using different proposed methods: approx.= Approximate match with initial motif; filter = Approximate match with environment filter; mrf = Motif refinement algorithm with the environment filter;voting = Voting method with extended motif filter; SVM = Feature vectors method with extended motif filter and support vector machine. See Appendix A for the ROC analysis of other enzyme families.

included in the Appendix section. In the ideal case, a perfect method should have a curve that passes through the coordinate (0,1) point, which reveals that it can achieve 100% TP rate (100% precision) and 0 % FP rate (100% recall). On the other hand, a method which forms a diagonal line from (0,0) to (1,1) would imply a performance of random guesses. In short, a good method should have a ROC curve close to the top left corner of the graph. Another performance measure related to ROC curve is called the area under the ROC curve (AUC). Table 5.5 lists the AUCs of the five methods testing five EC families. Larger area usually indicates better performance. Although not all of our methods performed better than the approximate graph matching technique (i.e. our baseline method) in all cases, some of our methods did produce substantial improvement to the functional annotations. In the following sections, we will discuss our observations of each method in detail.

| EC number | Approx. | Env filter | MRF + filter | voting | svm |
|---|---|---|---|---|---|
| 3.4.21 | 0.580 | 0.671 | 0.468 | 0.630 | **0.678** |
| 3.4.22 | 0.642 | 0.581 | 0.488 | **0.696** | 0.643 |
| 6.3.2 | 0.332 | 0.430 | 0.363 | **0.564** | 0.550 |
| 1.1.1 | 0.554 | 0.701 | 0.726 | **0.740** | 0.701 |
| FAD binding (1.8.1 + 1.18.1) | 0.559 | 0.420 | 0.442 | 0.523 | **0.719** |

**Table 5.5.** The area under curve (AUC) with five different methods testing five EC families. Those five methods are : Approx.= Approximate matching with initial motif; Env filter = Approximate matching with environment filter; MRF + filter = Motif refinement algorithm with the environment filter;voting = Voting method with extended motif filter; SVM = Feature vectors method with extended motif filter and support vector machine. The bold numbers are the largest number of each row, which indicates the best performance possible among all five methods.

### 5.4.1 Results of approximate match with initial motif

Tables 5.6 shows the number of true and false matches found in each enzyme family. All families are reported with reasonable amount of matches. And since we removed all the 'easy' matches from the dataset beforehand, the approximate matching technique obviously performed better than PSI-Blast in terms of the number of recovered remote homologs. The reason why the sequence-based annotation methods like PSI-Blast cannot detect those TPs is that other methods focus on the entire configuration of the proteins. For example, when Blast tries to align the query sequence to the database, protein which has a longer matches with the query sequence normally has higher probability to get picked during the statistical computation of e-values. The advantage of this approach is that it has very few false positives, as the matches are somehow similar to the query protein. Nonetheless, proteins (especially enzymes) can retain their functions largely due to their active regions, not the rest of the protein structures. As a result, although PSI-Blast can pick up homologs accurately, it has difficulties to recover remote homologs which have diverse structures. In short, the search space of sequence-based method is restricted, and in this case, it may stuck with the local optimal solution. Table 5.7 exactly shows a proof of this situation. Both methods were tested on the dataset before the PSI-Blast results were filtered, and their TPs (EC 3.4.21) together with their SCOP family ids are listed in the table. Approximate matching technique not only included all the results from PSI-Blast, but also recovered enzymes from different SCOP families. PSI-Blast, on the other hand, only found the proteins with SCOP id 50514. As SCOP database classifies proteins by the resemblance of their sequences and 3D structures, using active site as searching condition can effectively recover remote homologs while avoiding local optimal solutions.

| EC number | Number of true positives | Number of false positives |
|---|---|---|
| 3.4.21 | 21 | 557 |
| 3.2.22 | 14 | 507 |
| 6.3.2 | 18 | 642 |
| 1.1.1 | 12 | 502 |
| FAD binding (1.8.1 + 1.18.1) | 14 | 598 |

**Table 5.6.** The number of true positives and false positives captured for each enzyme family using approximate matching method (the baseline method).

Unfortunately, approximate matching method also has its shortcoming — it produces large amount of false positives. This is expected as the initial motif is usually pretty small (range from 1aa to 15aa), and the approximate matching method allows partial matches by introducing the scoring function. These two factors resulted in numerous random matches from unrelated proteins. In fact, we often found many matches within a single true positive in different locations, meaning that this method cannot even locate the active site correctly. Therefore, some filtering methods are needed to post-process the results we got from approximate matching method.

The AUC curves for the approximate matching method showed an interesting phenomenon. All of their curves formed staircase-like step functions. It means that large amount of matches obtained with the same mismatch score. When the score threshold is raised, large amount of TPs and FPs will be lost, thus having a huge jump between data points. If the score threshold was higher than the score of a perfect match (the initial motif matched with a site with the exact same amino acid composition), then no protein would be retrieved.

### 5.4.2 Results of approximate match with environment filter

Overall, the filter successfully eliminated large amount of unrelated proteins. Compared with the matching results without using the environment filter, as seen in table 5.5, three out of five EC families showed a positive response to the filter. And AUC increased from 15% to about 30%. All of these facts entail that at certain level, the surrounding distributions of residues are the determine factors for the emergence of active regions. However, the introduction of the environment filter also brought us another side effect—the reduction of the true positives. Both EC families 3.4.22 and FAD binding sites exhibited drops on their AUCs after the environment filter was applied. Fig 5.3 illustrates the decreasing trend of the TPs and FPs as we varied the threshold. Although both TP and FP curves have the tendency to decrease as we increase the environment threshold, the FP curve seems to go down more rapidly. In fact, the TP curve is on top of the FP curve most of the time, meaning that the remaining percentage of TPs is higher than

| PSI-blast TP results | | | Approx. matching TP results | | | |
|---|---|---|---|---|---|---|
| PDB ID | Chain ID | SCOP family ID | PDB ID | Chain ID | SCOP family ID | Shared by both methods |
| 1z6e | a | 50514 | 1z6e | a | 50514 | * |
| 2bz6 | h | 50514 | 2bz6 | h | 50514 | * |
| 1vzq | h | N/A | 1vzq | h | N/A | * |
| 2bvr | h | N/A | 2bvr | h | N/A | * |
| 1p57 | b | 50514 | 1p57 | b | 50514 | * |
| 1gj7 | b | 50514 | 1gj7 | b | 50514 | * |
| | | | 1gci | _ | 52744 | |
| | | | 1h2w | a | 50994 | |
| | | | 1i71 | a | 57441 | |
| | | | 1k32 | a | 68933 | |
| | | | 1lcy | a | 74933 | |
| | | | 1scj | a | 52744 | |
| | | | 1wpo | a | 50790 | |
| | | | 1xf1 | a | N/A | |
| | | | 2e7v | a | N/A | |
| | | | 2gef | a | N/A | |
| | | | 2h5c | a | 50495 | |
| | | | 1svp | a | 50596 | |
| | | | 1agj | a | 50495 | |
| | | | 1jhf | a | 51307 | |
| | | | 1q0p | a | 53301 | |
| | | | 1qzm | a | 81269 | |
| | | | 1umu | a | 51307 | |
| | | | 1yks | a | 52724 | |
| | | | 1z0w | a | N/A | |
| | | | 2fzs | a | 52097 | |
| | | | 1svb | _ | 81284 | |

**Table 5.7.** Comparison of the true positives obtained from PSI-Blast and approximate matching method. Both methods used the data set before the 'easy' matches are removed. Query protein used in this experiment is 1mct with EC no. 3.4.21. N/A in *SCOP family ID* indicates that no entry found in the SCOP database. If both methods share the same protein, a star will show up in the last column. Notice that the result from PSI-Blast is a proper subset of the results from approximate matching method.

the FPs at any given parameter values. This moving trend can also be seen in other enzyme families (their graphs are included in the Appendix B). Yet, if the approximate matching had a decreasing TP trend smaller than the environment filter, its AUC would end up being larger than the filter's AUC. That was exactly what happened to the EC families 3.4.22 and FAD binding site experiments.

The causes of diminishing true positives can be attributed to the diversity of the true samples and lack of proteins for profile generation. If a protein had a really different structure than the query protein, its active site environment might also be very different even though they both have the same function. This situation can be seen when a true protein was filtered through a high threshold. A solution to this problem is to include more proteins with distinct structures (e.g. different SCOP families) during the profile generation. Inaccurate distribution from the profile is another reason for the loss of the TPs. As seen in some figures from the Appendix such as EC 1.1.1, the curves never reach 100 % even when the threshold arrives at its maximum value in the graph. This is because some amino acids in the profile have very low standard deviation. During the profile difference calculation, the quotient became infinite (or a very large number, as we added some pseudo-counts before the calculation). Similar to the first problem, adding more proteins to the profile should result in larger standard deviations, which should alleviate the problem.

### 5.4.3 Results of motif refinement algorithm with environment filter

To demonstrate the convergence nature of our iterative refinement algorithm, we display the maximum potential change in IPF while constructing the refined motif for EC family 3.4.21 (see Figure 5.4). On average, our IPF executions

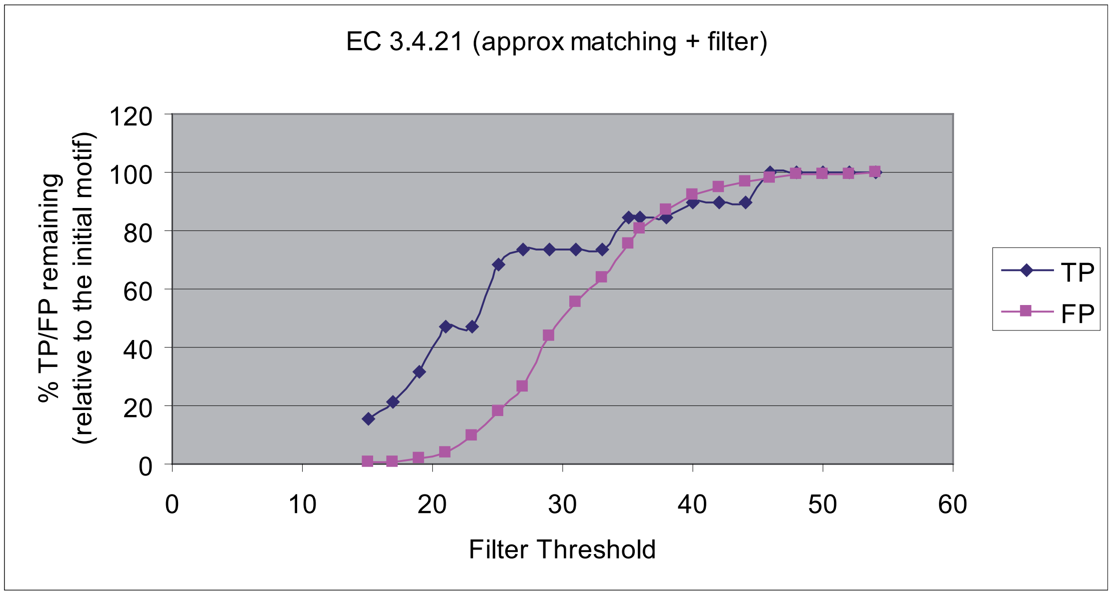


**Figure 5.3.** The effects of the environment filter on TP and FP from the enzyme family EC 3.2.21. The x axis indicates the filter threshold. Smaller filter threshold means stricter filters applied to the dataset. The y axis indications the percentage of TP and FP matches remaining with respect to the initial motif. The Appendix B shows the graphs for other enzyme families.

converged between two and three iterations. Overall, our algorithm finished within three to eight iterations as well. Figure 5.5 shows the number of proteins captured at each iteration for the EC family 3.4.21. Both graphs indicate the trend of increasing number of matched proteins as the model converged, which can also be seen in all the EC families we tested. This shows that MRF model becomes more generalize to a particular protein family as it converges.

Unfortunately, from the AUC and ROC analysis, the performance of the motif refinement algorithm is not as good as expected. Only two (EC 6.3.2 and 1.1.1) out of five experiments have AUC values larger than the baseline method (approximate matching). Also, compare with other methods we proposed in this paper, the refinement method has the smallest AUC in the first three experiments. There

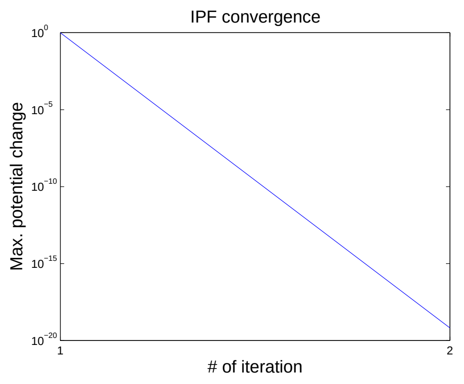


**Figure 5.4.** Convergence of the potential values when constructing a MRF. Throughout our experiments, the IPF normally finished within two to three iterations.

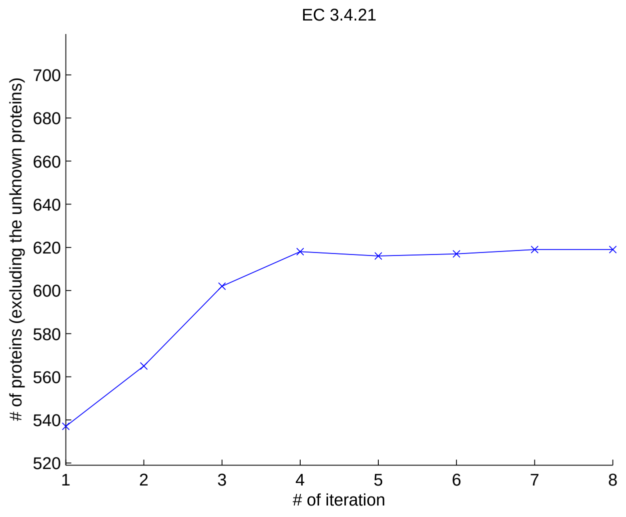


**Figure 5.5.** Convergence of our motif refinement algorithm on EC 3.4.21. It shows the number of proteins captured at each iteration. Similar converging trend also appeared in other enzyme families we tested.

are two possible reasons why it performed poorly. Actually, the refined motifs in the experiments did pick up additional remote homologs. With a proper model threshold of $T_2$, it would even pick up TPs that were originally eliminated in the initial stage. Unfortunately, during the initial stage, large amount of FPs got included, and those FPs remained as the algorithm iterated. To make the situation worse, those FPs would be used to construct a new MRF model, which would end up gathering more FPs in the re-matching stage. As a result, the proteins list contained a lot of junks as FPs accumulated at every iteration. This is called the propagation effect. This effect is very common in iterative algorithms such as PSI-Blast. To solve the problem, we have to make sure the quality of initial matching results so that FPs cannot retain and propagate. We have already applied the environment filter to filter out some of the FPs in the initial matching stage, but it was not powerful enough (as seen from the AUC table). One may either crate a better filter, or manually gather a list of TPs to bypass the first stage of the algorithm. The refined motif should have higher precision and recall.

Parameter selections can also be another factor of restricting the number of FPs during initial stage. Two thresholds: $T_1$ (initial score threshold) and $T_3$ (filter threshold) determine which proteins can match with the initial motif in the first stage. In this experiment, these two parameters were fixed based on the best results chosen from the previous experiments (the heuristic nature of our experimental study). However, since our goal now has shifted to minimize the FP, the optimal parameter from previous experiments do not necessary be optimal in this case. As a result, re-searching these three parameters may result in better (or optimal) results.

Another possible reason for the poor performance is the structure of the initial motif. Our motif model uses MRF, which highly depends on potential values from each maximal clique to compute the probability values. If an initial motif had only one maximal clique, the potential values for that clique would be exactly the same as the maximum likelihood values, which would not be useful. The true power of MRF comes from the variety of potential functions, since the combinations of those potential values can estimate the joint probability from different configurations. In other words, more maximal cliques in the motif make the resulting MRF more flexible. To achieve this goal, one may try to remove some of the edges in the MRF topology so that a large clique can be broken down into smaller cliques.

### 5.4.4 Results of voting method using extended motif filter

Among all the methods listed in the AUC analysis, voting method has the best performance in terms of the percentage of improvement. In four out of five experiments, the voting method always outperformed the baseline approximate matching technique (except for the FAD binding family). Its improvement rate can go up to 70% (EC 6.3.2). In addition, three experiments: EC 3.4.22, EC 6.3.2, and EC 1.1.1 show that voting method formed the largest AUC among all the proposed methods, meaning that voting method provides the best annotation results on those enzyme families. The significant changes on the results can be contributed to the average scheme voting method utilized. The geometric mean heavily penalizes the proteins to which the extended motifs disagree. As multiplication is used to aggregate the proteins' mismatch scores, a probable protein would have an average score of zero even if one of the extended motifs could not capture that protein — any proteins that did not include in their matching results would have a zero mismatch score. This effect of multiplication would potentially filter out all the FPs: only proteins that acquire the consensus from all the extended motifs are remained as functional homologs. This voting method actually makes biological sense because every extended motif includes different additional features from the active site environment. If a protein that satisfies all the characteristics described by the motifs, that protein will have a high probability to be related to the query protein. We also tried another voting strategy called the arithmetic mean. Arithmetic mean uses additions instead of multiplications to sum up the scores. However, the voting results were not as good as the geometric mean. The results of the arithmetic mean were not shown in this paper.

### 5.4.5 Results of feature vector method using extended motif filter

Among all the methods listed in the AUC analysis, feature vector method has the best performance in terms of the stability of improvement. In all of our experiments, the feature vector method always can provide some degree of improvement, ranges from 0.1% to 65%. Even in the FAD binding family experiment, where no proposed method so far could improve the baseline approximate matching result, the feature vector method can raise the AUC up to 28.5 %. The results shown in the ROC graphs and the AUC table are non-binary features using RBF as SVM kernel. As mentioned before, we tried different feature vector construction methods, including binary and non-binary feature values, along with different kernels for the SVM classifier like linear and Radial Basis Function (RBF) kernels. Nonetheless, all those combinational constructions (the results were not shown in this paper) did not provide any significant improvement comparing with the current construction method. This means that the mismatch scores for each extended motif do offer valuable information to the classifier. Also, the data points in the feature space are so complicated that using linear hyperplane cannot separate the remote homologs from the functionally unrelated proteins efficiently. Thus we needed to use the 'curved' decision boundary provided by the RBF kernel. To further enhance the annotation ability of the feature vector method, one may try other machine learning techniques such as feature selections, feature extractions, or different classifiers. One can also use more extended motifs to enlarge the size of the feature vector so that better results can be achieved. The main purpose of this experiment is just a proof-of-concept — we just want to show that it is feasible to use our extended motif to achieve better annotation results. People always can fine-tune our methods to suit their needs.

# Chapter 6

# Conclusion and Future Work

In this paper, we first discussed the limitations of the current sequential and structural motif representations for protein functional annotations. From that we came up with two goals that a good annotation method using motif models should achieve— 1) it should be sensitive to accommodate the variations of the active sites within a family and 2)it should be specific to pick up proteins from a protein family, especially if they are distantly related. To attain these two goals, we proposed four different approaches to refine the annotation results based on a query protein with its initial motif. One of which involves the reconstruction of the motif model using a statistical model MRF, and the rest of them utilize the surrounding environment of the active site to eliminate the false positives. These include the environment filter and the extended motif filter methods. The experiments on five sets of enzyme families demonstrated that our algorithms can have certain degree of performance improvement when compared with the baseline method. Some of our experiments can even get up to 70% increase in terms of the AUC improvement. This fact illustrates that our methods obtain remote homologs across diverse global structures (as they are from different SCOP families) using a single query protein. Among all of our approaches, voting method has the best performance in terms of the percentage of improvement, and feature vector method has the best performance in terms of the stability of improvement.

Nevertheless, methods like the motif refinement algorithm with MRF did not work well as expected. In some cases, they worked even worse than the baseline approximate matching method. The followings summarized the action can be taken to avoid their inferior performance:

- For the environment filter, one can add more proteins to generate the environment profile such that more information of the active site neighbors are recorded.
- For the motif refinement algorithm, one needs to make sure the proteins obtained from the initialization stage are correct to prevent propagation effect. This can be done by applying stricter filters or manually curated results to jumpstart our algorithm.
- The topology of the initial motif also affects the performance of motif refinement algorithm. One may remove the longer edges in the motif in order to construct a flexible MRF.
- For the extended motif filter, one can combine multiple extended motifs to enhance the functional annotation ability. The more extended motifs are included, the more powerful filter it will become. In addition, one can resort other machine learning techniques to fine tune the voting and feature vector methods.

For our future works, we will show that fully automatic approach of motif refinement is possible. In this study, all initial patterns were obtained from the literature/database. Instead of supplying an initial motif manually, we can first make use of a subgraph mining tool such as FFSM [35] to gather a set of initial motifs which occur in the input sets frequently. Our algorithm will then take over and refine each of the motif. Finally, these optimized models will be tested statistically to make sure they are not generated by chance. Motifs that have high significance will output to the user. By using this approach, we can truly perform a large-scale test to construct a more effective motif. The concept of statistical testing can also be employed to other approaches, for example, we can measure the significance of the filters by computing the signal-to-noise ratio from an external testing data. Also, we can unite all of our proposed methods together by voting or other ensemble means to attain better results.

# Appendix A

# Complete ROC Analysis from all Enzyme Families

The graphs shown in this section illustrate the ROC curves from different enzyme family using our methods. Fig A.1 in this section and Fig 5.2 in 5.4 are the same graph.

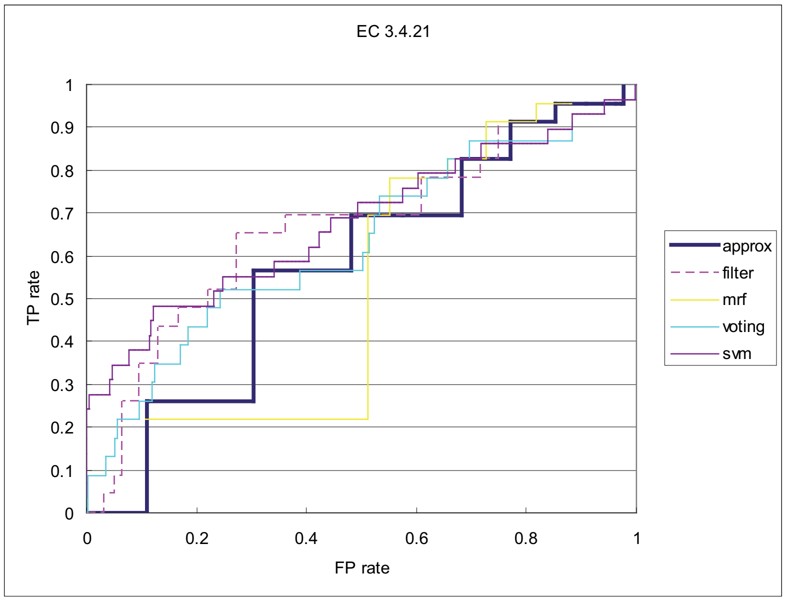


**Figure A.1.** ROC analysis of enzyme family EC 3.2.21.

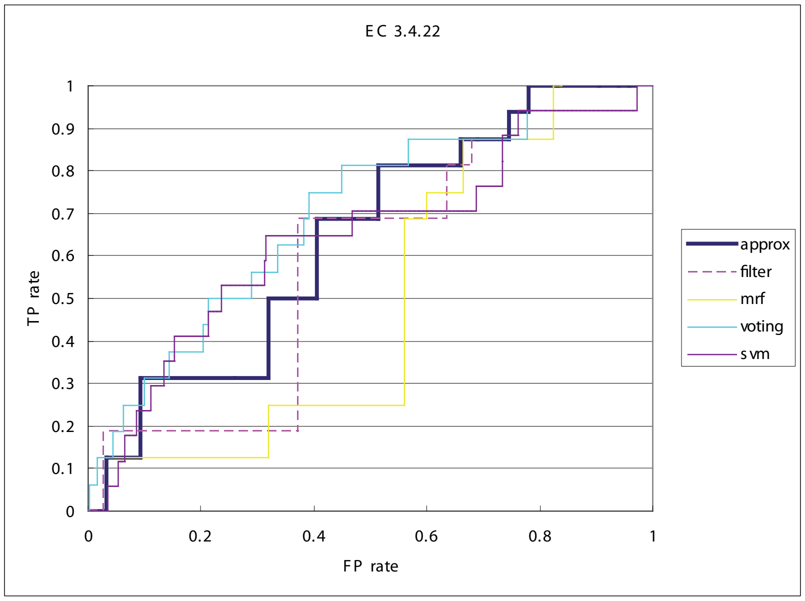


**Figure A.2.** ROC analysis of enzyme family EC 3.2.22.

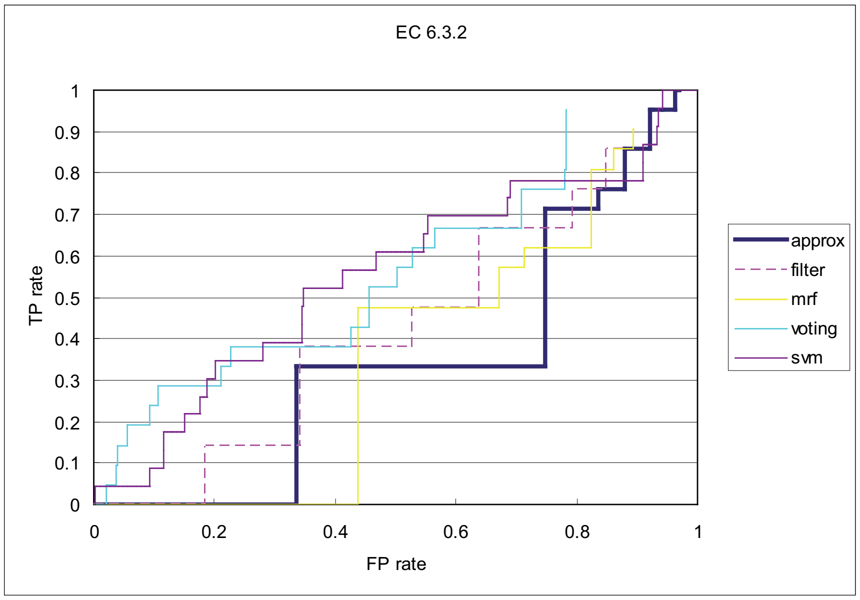


**Figure A.3.** ROC analysis of enzyme family EC 6.3.2.

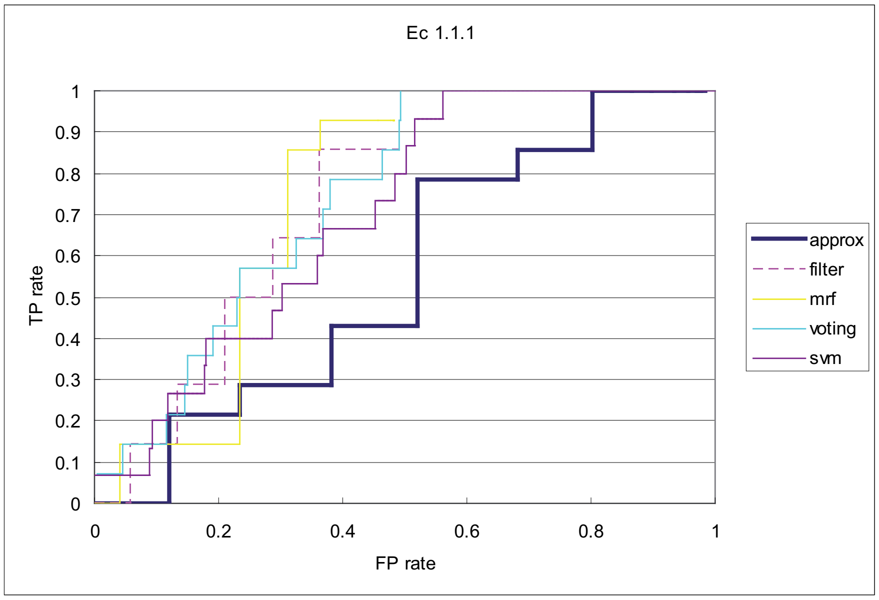


**Figure A.4.** ROC analysis of enzyme family EC 1.1.1.

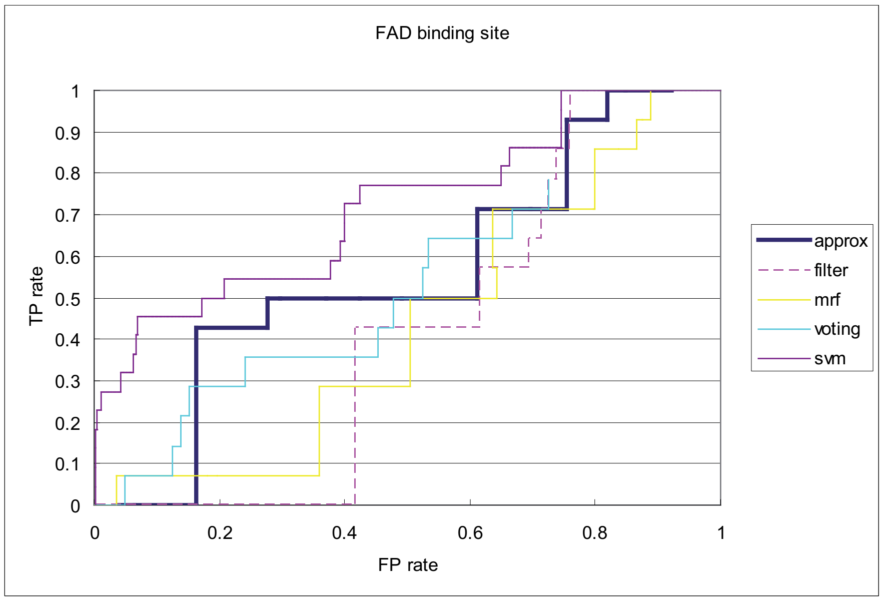


**Figure A.5.** ROC analysis of FAD binding families (1.8.1 + 1.18.1).

# Appendix B

# Complete Results from 5.4.2

The graphs shown in this section illustrate the changes of the TPs and FPs with different filter thresholds. Fig B.1 in this section and Fig 5.3 in 5.4.2 are the same graph.

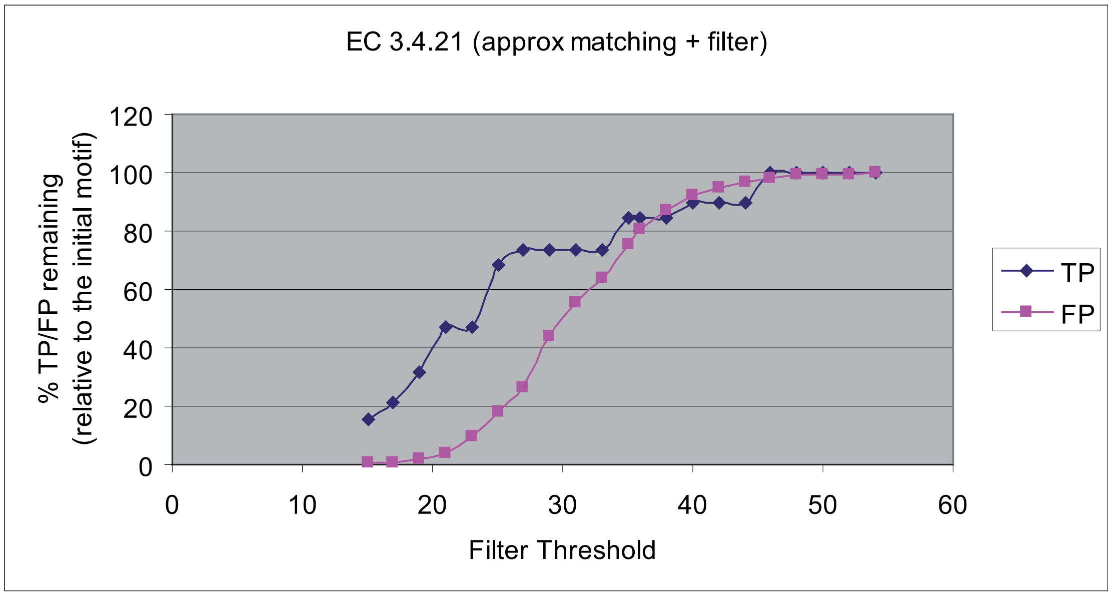


**Figure B.1.** The effect of the environment filter on TP and FP on enzyme family EC 3.2.21.

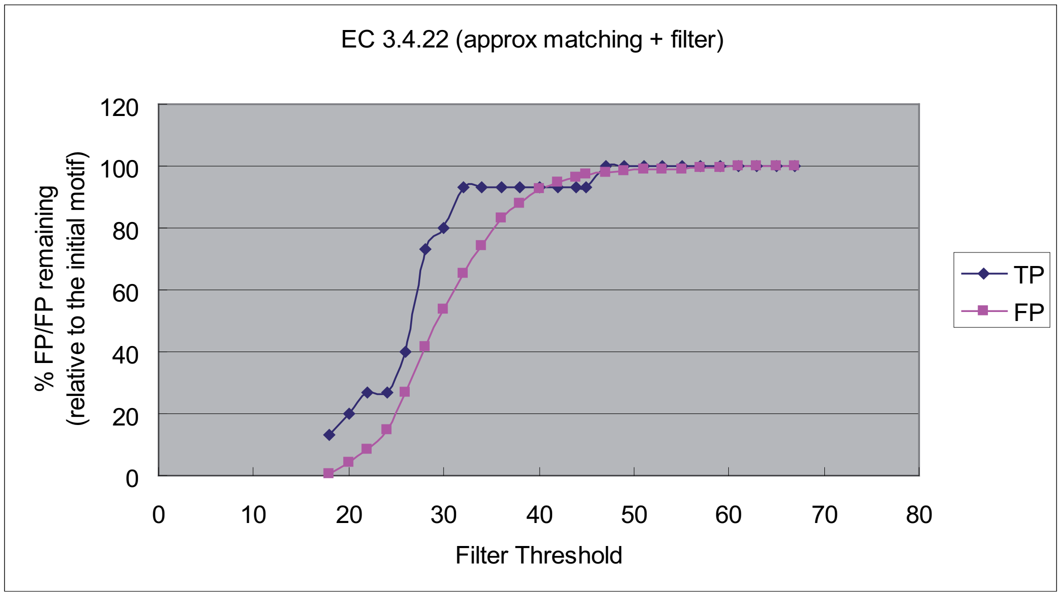


**Figure B.2.** The effect of the environment filter on TP and FP on enzyme family EC 3.2.22.

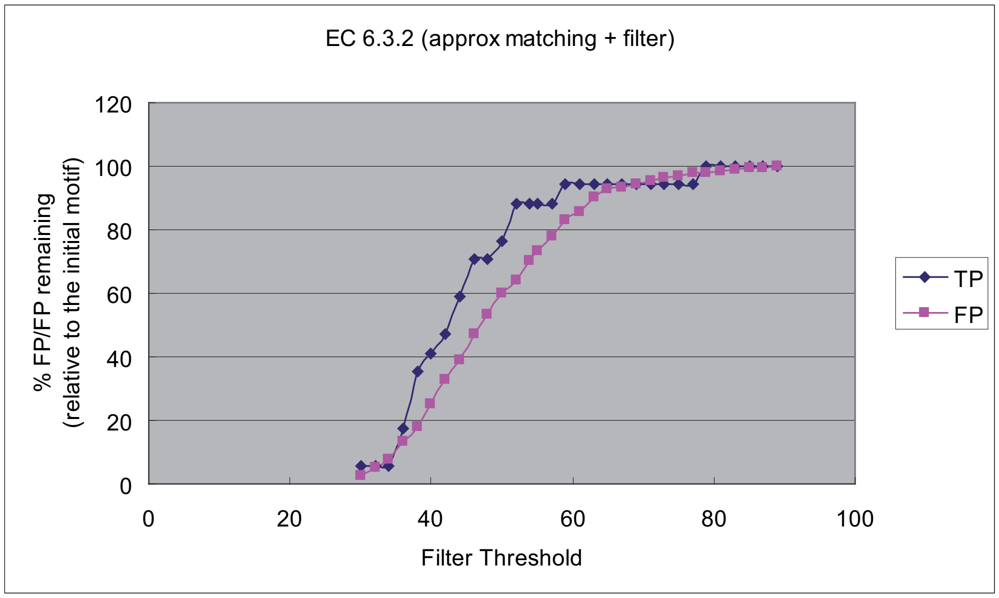


**Figure B.3.** The effect of the environment filter on TP and FP on enzyme family EC 6.3.2.

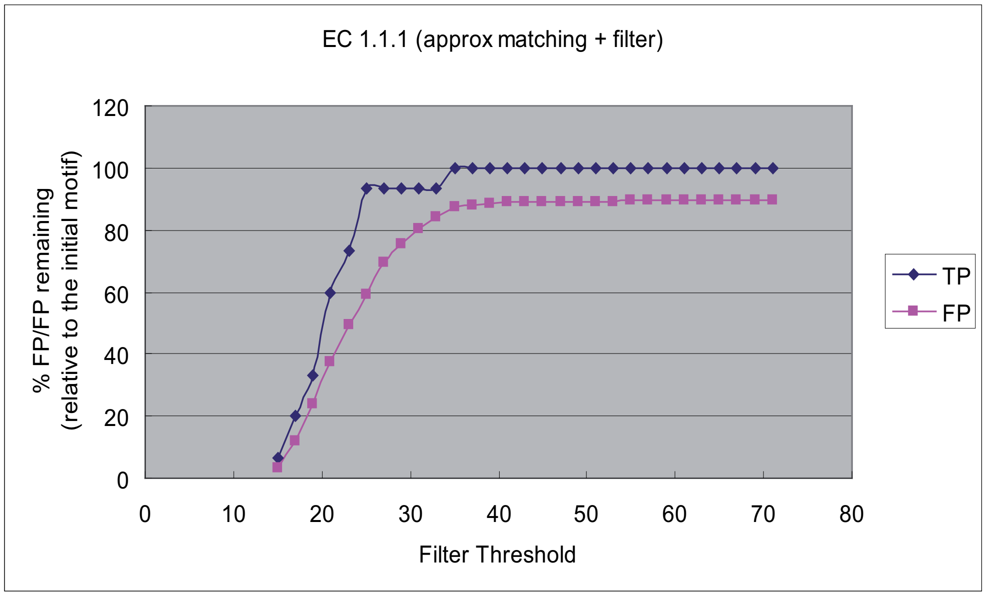


**Figure B.4.** The effect of the environment filter on TP and FP on enzyme family EC 1.1.1.

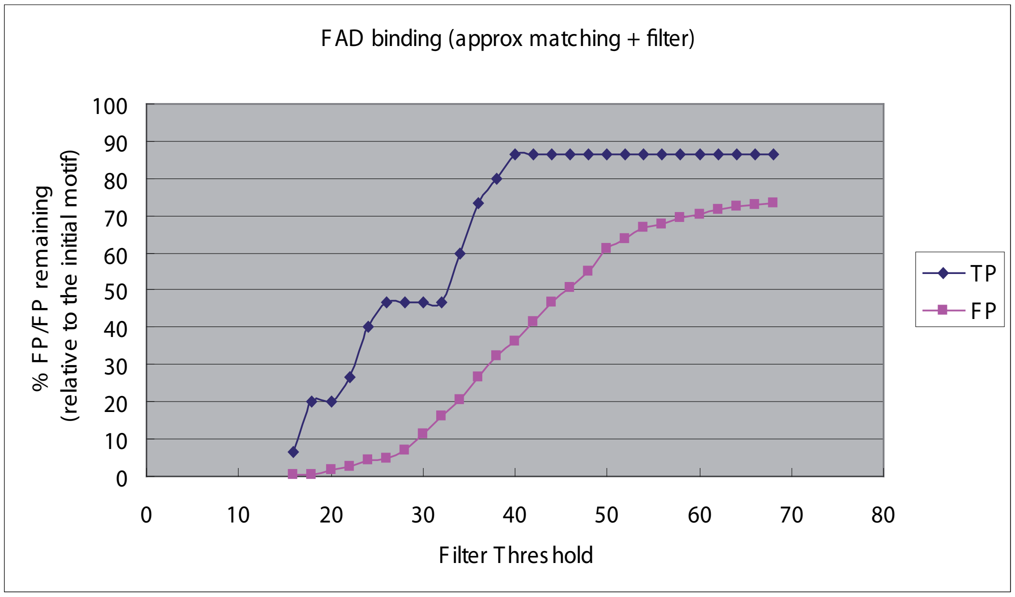


**Figure B.5.** The effect of the environment filter on TP and FP on FAD binding families (1.8.1 + 1.18.1).

# References

[1] D. K. P. R. J. S. A. Kolinski, M.R. Betancourt. Generalized comparative modeling (genecomp): A combination of sequence comparison, threading, and lattice modeling for protein structure prediction and refinement. *Proteins*, 44(2), 2001.

[2] E. Alpaydin. *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2004.

[3] S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucl. Acids Res.*, 25(17):3389–3402, 1997.

[4] P. J. Artymiuk, A. R. Poirrette, H. M. Grindley, D. W. Rice, and P. Willett. A graph-theoretic approach to the identification of three-dimensional patterns of amino acid side-chains in protein structures. *Journal of Molecular Biology*, 243:327–44, 94.

[5] T. Bailey, N. Williams, C. Misleh, and W. Li. MEME: discovering and analyzing DNA and protein sequence motifs. *Nucleic Acids Res.*, 34:W369–373, Jul 2006.

[6] J. Barker and J. Thornton. An algorithm for constraint-based structural template matching: application to 3d templates with statistical analysis. *Bioinformatics*, 19(13):1644–9, 2003.

[7] B. Berger and M. Singh. An iterative method for improved protein structural motif recognition. In *RECOMB '97: Proceedings of the first annual international conference on Computational molecular biology*, pages 37–46, 1997.

[8] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucl. Acids Res.*, 28(1):235–242, 2000.

[9] P. Bork. Shuffled domains in extracellular proteins. *FEBS Lett.*, 286:47–54, Jul 1991.

[10] P. Bork and T. Gibson. Applying motif and profile searches. *Meth. Enzymol.*, 266:162–184, 1996.

[11] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[12] P. Bradley, P. S. Kim, and B. Berger. TRILOGY: Discovery of sequence-structure patterns across diverse proteins. *Proceedings of the National Academy of Sciences*, 99(13):8500–8505, June 2002.

[13] D. H. B. Brian Y. Chen, Viacheslav Y. Fofanov. Geometric sieving: Automated distributed optimization of 3d motifs for protein articlefunction prediction. *Proceedings of The Tenth Annual International Conference on Computational Molecular Biology (RECOMB 2006)*, 2006.

[14] P. Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *J. Mol. Biol.*, 212:563–578, Apr 1990.

[15] M. J. Buck and J. D. Lieb. Chip-chip: considerations for the design, analysis, and application of genome-wide chromatin immunoprecipitation experiments. *Genomics*, 83(3):349–360, Mar 2004.

[16] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[17] B. Y. Chen, D. H. Bryant, V. Y. Fofanov, D. M. Kristensen, A. E. Cruess, M. Kimmel, O. Lichtarge, and L. E. Kavraki. Cavity-aware motifs reduce false positives in protein function prediction. *Comput Syst Bioinformatics Conf*, 2006.

[18] X. Chen, L. Guo, Z. Fan, and T. Jiang. W-AlignACE: an improved Gibbs sampling algorithm based on more accurate position weight matrices learned from sequence and gene expression/ChIP-chip data. *Bioinformatics*, 24:1121–1128, May 2008.

[19] F. Crick. Central dogma of molecular biology. *Nature*, 227:561–563, Aug 1970.

[20] J. H. C.X. Ling and H. Zhang. AUC: a Better Measure than Accuracy in Comparing Learning Algorithms. *Proceedings of 2003 Canadian Artificial Intelligence Conference*, 2003.

[21] W. Day and F. McMorris. Critical comparison of consensus methods for molecular sequences. *Nucleic Acids Res.*, 20:1093–1099, Mar 1992.

[22] I. Eidhammer, I. Jonassen, and W. R. Taylor. *Protein bioinformatics : an algorithmic approach to sequence and structure analysis.* J. Wiley & Sons, Chichester, West Sussex, England ;; Hoboken, NJ, 2004. ID: 54982066.

[23] C. Fan, P. Moews, C. Walsh, and J. Knox. Vancomycin resistance: structure of D-alanine:D-alanine ligase at 2.3 A resolution. *Science*, 266:439–443, Oct 1994.

[24] R. George, R. Spriggs, G. Bartlett, A. Gutteridge, M. MacArthur, C. Porter, B. Al-Lazikani, J. Thornton, and M. Swindells. Effective function annotation through residue conservation. *PNAS*, 102:12299–12304, 2005.

[25] R. A. George, R. V. Spriggs, J. M. Thornton, B. Al-Lazikani, and M. B. Swindells. SCOPEC: a database of protein catalytic domains. *Bioinformatics*, 20(suppl_1):i130–136, 2004.

[26] M. B. E. E. P. D. George M. Y., Ari D. K. Genomic overview of serine proteases. *Biochemical and Biophysical Research Communications*, 305(1):28–36.

[27] J. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. Unpublished manuscript, 1971.

[28] I. Hanukoglu and T. Gutfinger. cDNA sequence of adrenodoxin reductase. Identification of NADP-binding sites in oxidoreductases. *Eur. J. Biochem.*, 180:479–484, Mar 1989.

[29] H. Hegyi and M. Gerstein. The relationship between protein structure and function: a comprehensive survey with application to the yeast genome. *Journal of Molecular Biology*, 288(1):147–164, Apr 23 1999. LR: 20061115; PUBM: Print; CI: Copyright 1999; JID: 2985088R; 0 (Enzymes); 0 (Fungal Proteins); 0 (Proteins); ppublish.

[30] H. Hegyi and M. Gerstein. Annotation Transfer for Genomics: Measuring Functional Divergence in Multi-Domain Proteins. *Genome Res.*, 11(10):1632–1640, 2001.

[31] S. Henikoff and J. Henikoff. Amino Acid Substitution Matrices from Protein Blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[32] S. K. Hofmann, P. Bucher, L. Falquet, and A. Bairoch. The prosite database, its status in 1999. *Nucleic Acids Res*, 27(1):215–219, 1999.

[33] L. Holm and C. Sander. Dali/FSSP classification of three-dimensional protein folds. *Nucleic Acids Res.*, 25:231–234, Jan 1997.

[34] J. Huan, D. Bandyopadhyay, J. Prins, J. Snoeyink, A. Tropsha, and W. Wang. Distance-based identification of structure motifs in proteins using constrained frequent subgraph mining. *Computational systems bioinformatics / Life Sciences Society.Computational Systems Bioinformatics Conference*, pages 227–238, 2006.

[35] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraph in the presence of isomorphism. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM)*, pages 549–552, 2003.

[36] J. Huang, J. Lu, and C. X. Ling. Comparing naive bayes, decision trees, and svm with auc and accuracy. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 553, Washington, DC, USA, 2003. IEEE Computer Society.

[37] W. Humphrey, A. Dalke, and K. Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[38] K. Karplus, C. Barrett, and R. Hughey. Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, 14:846–856, 1998.

[39] L. Kelley, R. MacCallum, and M. Sternberg. Enhanced genome annotation using structural profiles in the program 3d-pssm. *J Mol Biol.*, 299(2):499–520, 2000.

[40] T. Lengauer and R. Zimmer. Protein structure prediction methods for drug design. *Brief. Bioinformatics*, 1:275–288, Sep 2000.

[41] H. Leung and F. Chin. Finding exact optimal motifs in matrix representation by partitioning. *Bioinformatics*, 21 Suppl 2:86–92, Sep 2005.

[42] A. C. Martin. PDBSprotEC: a Web-accessible database linking PDB chains to EC numbers via SwissProt. *Bioinformatics*, 20(6):986–988, 2004.

[43] B. Matthews. Structural and genetic analysis of the folding and function of t4 lysozyme. *The FASEB Journal*, 10:35–41, 1996.

[44] S. Maurer-Stroh and F. Eisenhaber. Refinement and prediction of protein prenylation motifs. *Genome Biol.*, 6:R55, 2005.

[45] A. Murzin, S. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–40, 1995.

[46] N. Nagano, C. A. Orengo, and J. M. Thornton. One fold with many functions: the evolutionary relationships between tim barrel families based on their sequences, structures and functions. *Journal of Molecular Biology*, 321(5):741–765, Aug 30 2002. LR: 20061115; PUBM: Print; JID: 2985088R; 0 (Enzymes); 0 (Ligands); 0 (Phosphates); EC 3.5.1.- (imidazole glycerol phosphate synthase); EC 3.5.4.- (Aminohydrolases); EC 5.3.1 (Aldose-Ketose Isomerases); EC 5.3.1.16 (Phosphoribosyl-5-amino-1-phosphoribosyl-4-imidazolecarboxiamide isomerase); RF: 44; ppublish.

[47] C. Orengo, A. Michie, S. Jones, D. Jones, M. Swindells, and J. Thornton. CATH - a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1108, 1997.

[48] X. Pennec and N. Ayache. A geometric algorithm to find small but highly similar 3d substructures in proteins. *Bioinformatics*, 14(6):516–22, 1998.

[49] J. Pevsner. *Bioinformatics and Functional Genomics.* J. Wiley & Sons, Chichester, West Sussex, England ;; Hoboken, NJ, 2003.

[50] C. T. Porter, G. J. Bartlett, and J. M. Thornton. The Catalytic Site Atlas: a resource of catalytic sites and residues identified in enzymes using structural data. *Nucl. Acids. Res.*, 32(90001):D129–133, 2004.

[51] L. R. Convergence of the iterative proportional fitting procedure. *The Annals of Statistics*, 23(4):1160–1174, 1995.

[52] D. RF. *Of URFs and ORFs: A Primer on How to Analyze Derived Amino Acid Sequences.*, volume 92. Mill Valley: University Science Books, 1986.

[53] A. Shah, C. Oehmen, and B. Webb-Robertson. SVM-HUSTLE–an iterative semi-supervised machine learning approach for pairwise protein remote homology detection. *Bioinformatics*, 24:783–790, Mar 2008.

[54] A. Stark and R. Russell. Annotation in three dimensions. pints: Patterns in non-homologous tertiary structures. *Nucleic Acids Res*, 31(13):3341–4, 2003.

[55] G. Stormo, T. Schneider, L. Gold, and A. Ehrenfeucht. Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Res.*, 10:2997–3011, May 1982.

[56] T. Tatusova and T. Madden. BLAST 2 Sequences, a new tool for comparing protein and nucleotide sequences. *FEMS Microbiol. Lett.*, 174:247–250, May 1999.

[57] J. R. Ullman. An algorithm for subgraph isomorphism. *Journal of the Association for Computing Machinery*, 23:31–42, 1976.

[58] A. Wallace, N. Borkakoti, and J. Thornton. Tess: a geometric hashing algorithm for deriving 3d coordinate templates for searching structural databases. application to enzyme active sites. *Protein Sci*, 6(11):2308–23, 1997.

[59] C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 730–735, 2006.

[60] G. Wang and R. L. Dunbrack. PISCES: a protein sequence culling server. *Bioinformatics*, 19:1589-1591, 2003.

[61] E. C. Webb. *Enzyme nomenclature 1992 : recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes.* Academic Press, 1992.

[62] Y. Xiao and M. Segal. Biological sequence classification utilizing positive and unlabeled data. *Bioinformatics*, 24:1198–1205, May 2008.

[63] X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: a profile-based approach. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 314–323, 2005.

[64] B. Zagrovic and V. Pande. How does averaging affect protein structure comparison on the ensemble level? *Biophys. J.*, 87:2240–2246, Oct 2004.