İSTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

PROTEIN FUNCTION PREDICTION USING HIDDEN MARKOV MODELS

M.Sc. Thesis by

Caner KÖMÜRLÜ, B.Sc.

Science Programme : Advanced Technologies
Programme: Computer Sciences

Supervisors : Assoc. Prof. Dr. Zehra Çataltepe

SEPTEMBER 2008

İSTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

PROTEIN FUNCTION PREDICTION USING HIDDEN MARKOV MODELS

M.Sc. Thesis by

Caner KÖMÜRLÜ, B.Sc.
704061004

Date of submission : 28 August 2008

Date of defence examination : 28 July 2008

Supervisor (Chairman): **Assoc. Prof. Dr. Zehra ÇATALTEPE**

Members of the Examining Committee **Prof.Dr. Muhittin GÖKMEN (İ.T.Ü.)**

Assoc. Prof. Dr. Özlem KESKİN (K.Ü.)

SEPTEMBER 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

SAKLI MARKOV MODELLERİ KULLANARAK PROTEİN FONKSİYON ÖNGÖRÜSÜ

YÜKSEK LİSANS TEZİ

Müh. Caner KÖMÜRLÜ
704061004

Tezin Enstitüye Verildiği Tarih : 28 Ağustos 2008

Tezin Savunulduğu Tarih : 28 Temmuz 2008

Tez Danışmanı : **Doc. Dr. Zehra ÇATALTEPE**

Diğer Jüri Üyeleri **Prof.Dr. Muhittin GÖKMEN (İ.T.Ü)**

Doç. Dr. Özlem KESKİN (K.Ü.)

EYLÜL 2008

ACKNOWLEDGEMENTS

First, I would like to thank my supervisor Assoc. Prof. Dr. Zehra Çataltepe who, among her all not less important deals than mine, spent whole her effort and her time during years for me. She never lost her hope and never reduced her expectations about my study and my future. She was not merely a supervisor for me.

Second, I would like to thank to my project colleagues, Aslı and Eser, for never grudging their knowledge, their help and also their warm friendship to me.

Third, I would like to thank Dr. Yücel Altunbaşak from Gatech and Dr. Özlem Keskin from Koç University for their contributions to our İTÜ Bioinformatics Project which was the right environment for my study and also Dr. Hakan Erdoğan from Sabancı University and Dr. Zafer Aydın from Gatech for their gratis contributions.

Next, I would like to thank to TÜBİTAK for their generous scholarship BİDEB-2228 which made certainly financial aspect of being an MS student much easier.

Finally, I would like to thank my family for their infinite trust in my attempts, tolerance to my defects and support to my dreams.

September 2008 Caner KÖMÜRLÜ

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....ii
TABLE OF CONTENTS.....iii
ABBREVIATIONS.....vi
TABLE LIST.....vii
FIGURE LIST.....ix
ÖZET.....x
SUMMARY.....xii

1. INTRODUCTION.....1

2. COMPUTATIONAL BIOLOGY AND BIOINFORMATICS.....3
2.1 Amino Acid Sequence.....3
2.2 Secondary Structure.....4
2.3 Similarity Measures.....6
2.4 Sequence Alignment.....7
2.4.1 Pairwise Alignment.....8
2.4.1.1 Global Alignment – Needleman-Wunsch Algorithmic.....8
2.4.1.2 Local Alignment – Smith-Waterman Algorithm.....9
2.4.2 Multiple Sequence Alignment.....9
2.5 Secondary Structure Prediction.....11
2.6 Profile Analysis.....12

3. PROTEIN FUNCTION PREDICTION.....14
3.1 Definition of Protein Function.....14
3.2 Diverse Approaches in Protein Function Prediction.....14
3.3 Homology and Residue Conservation.....16

4. PREVIOUS WORK.....17

5. HIDDEN MARKOV MODELS.....20

6. PROFILE HIDDEN MARKOV MODELS.....21
6.1 Hidden Markov Models in Biological Sequences.....21
6.2 HMMER – Biosequence Analysis Using Profile-HMM.....22
6.2.1 Introduction to HMMER.....22
6.2.2 Function Prediction Procedure Using HMMER.....24
6.2.2.1 Similarity Matrix Generation Via Neighborhood Based Sequence Profile Comparison.....24
6.2.2.2 Data Set Generation Via Class Based Profile Sequence Comparison 27

6.3 HHsearch – HMM-HMM Comparison for Homology Detection.....28
Pairwise Alignment of A Sequence and A Profile-HMM.....28
Pairwise alignment of profile-HMM's.....29
6.3.2 Protein Function Prediction Using HHsearch.....33
6.3.3 Alignment Score Estimation Using HMM.....34
6.3.3.1 Alignment Score Estimation with HMM Using Amino Acid Sequence Only.....35
6.3.3.2 Alignment Score Estimation with HMM Using Amino Acid Sequence and Secondary Structure.....40
6.3.3.2.1 Alignment Score Estimation with HMM Using Amino Acid Sequence and Predicted Secondary Structure.....40
6.3.3.2.2 Alignment Score Estimation with HMM Using Amino Acid Sequence and Actual Secondary Structure.....42
6.4 PRC – The Profile Comparer.....44
6.4.1 Introduction to PRC.....44
6.4.2 Function Prediction Using Procedures Using PRC.....45
6.4.2.1 Similarity Matrix Generation Via Neighborhood Based Sequence Profile Comparison.....46
6.4.2.2 Similarity Matrix Generation Via Class Based Profile-Profile Comparison.....47
6.5 Improved Profile Methods.....48
6.5.1 Sequence-Profile Comparison in Best-Match Approach using NR Data..50
6.5.2 Sequence-Profile Comparison in Best-Match Approach using 5-class Enrich Data.....52
6.5.3 Profile-Profile Comparison in Best-Match Approach using NR Data.....53
6.5.4 Profile-Profile Comparison in Best-Match Approach using 5-class Enrich Data.....54
6.6 Other Tools Working on Profile Hidden Markov Models.....55

7. PATTERN RECOGNITION METHODS.....57
7.1 Classification Algorithms.....57
7.1.1 KNN - k Nearest Neighbor.....57
7.1.2 tNN – Threshold Nearest Neighbor.....58
7.1.3 Best-Match Classification.....58
7.2 Evaluation Methods.....60
7.2.1 Accuracy.....61
7.2.1 ROC – Receiver Operating Characteristic Curve.....62
7.2.2 AUC – Area Under Curve.....63

8. EXPERIMENTS.....65
8.1 Data.....65
8.1.1 5-class GO Data.....65
8.1.2 27-class GO data.....69
8.1.3 NR data set.....73
8.1.4 5-class Enriched Data.....74
8.1.5 27-class Enriched Data.....77

8.2 Environments.....79

9. RESULTS AND EVALUATION.....81
9.1. Sequence-Profile Comparison with HMMER.....81
9.2 Profile-Profile Comparison with HHsearch.....82
9.3 Profile-Profile Comparison with PRC.....87
9.4 Improved Profile Methods.....89
9.4.1 Sequence-Profile Comparison with NR Data.....89
9.4.2 Sequence-Profile Comparison with Enriched 5-class Data.....92
9.4.3 Profile-Profile Comparison with NR Data.....94
9.4.4 Profile-Profile Comparison with Enriched 5-class Data.....97

10. CONCLUSION.....98

REFERENCES.....103

AUTOBIOGRAPHY.....109

ABBREVIATIONS

AUC	Area Under ROC Curve
AA	Amino Acid
AAS	Amino Acid Sequence
BLAST	Basic Local Alignment Search Tool
DSSP	Dictionary of Protein Secondary Structure Convention
GO	Gene Ontology
HEL	Helix Sheet Loop Convention
HMM	Hidden Markov Models
HHsearch	HMM-HMM Search
KNN	K Nearest Neigbor
MSA	Multiple Sequence Alignment
NR	Non-redundant
PDB	Protein Data Bank
PRC	Profile Comparer
PSI-BLAST	Position Specific Iterated Basic Local Alignment Search Tool
PSIPRED	Protein Structure Prediction Tool
ROC	Receiver Operating Characteristics
SAM	Sequence Alignment and Modelling Software System
SCOOP	Simple Comparison of Outputs Program
SS	Secondary Structure
SVM	Support Vector Machine
tNN	Threshold Nearest Neighbor

TABLE LIST

Page Number

Table 7.1: Confusion Matrix 61
Table 8.1: Selected molecular functions from GO annotation 66
Table 8.2: Protein labeling in 5-class GO data set 67
Table 8.3: Selected GO labels for 27-class data set with 40% sequence identity 69
Table 8.4: Selected GO labels for 27-class data set with 70% sequence identity 71
Table 9.1: AUC values obtained by search score and e-value score 82
Table 9.2: Mean AUC Actual and Predicted Secondary Structure Contribution at vasious weight in HHsearch method 87
Table 9.3: AUC values obtained by search score and e-value score 89
Table 9.4: Accuracy of Best-Match with n = 1, using NR data in sequence-profile comparison 90
Table 9.5: Accuracy of Best-Match with n = 3, using NR data in sequence-profile comparison 90
Table 9.6: Accuracy of Best-Match with n = 7, using NR data in sequence-profile comparison 91
Table 9.7: Accuracy of Best-Match with n = 11, using NR data in sequence-profile comparison 91
Table 9.8: Accuracy of Best-Match with n = all, using NR data in sequence-profile comparison 92
Table 9.9: Accuracy of Best-Match with n = 1, using enrich 5-class data in sequence-profile comparison 92
Table 9.10: Accuracy of Best-Match with n = 3, using enrich 5-class data in sequence-profile comparison 93
Table 9.11: Accuracy of Best-Match with n = 7, using enrich 5-class data in sequence-profile comparison 93
Table 9.12: Accuracy of Best-Match with n = 11, using enrich 5-class data in sequence-profile comparison 94
Table 9.13: Accuracy of Best-Match with n = All, using enrich 5-class data in sequence-profile comparison 94
Table 9.14: Accuracy of Best-Match with n = 1, using NR data in profile-profile comparison 95
Table 9.15: Accuracy of Best-Match with n = 3, using NR data in profile-profile comparison 95
Table 9.16: Accuracy of Best-Match with n = 7, using NR data in profile-profile

comparison 96
Table 9.17: Accuracy of Best-Match with n = 11, using NR data in profile-profile comparison 96
Table 9.18: Accuracy of Best-Match with n = All, using NR data in profile-profile comparison 96
Table 9.19: Accuracy of Best-Match with n = All, using NR data in profile-profile comparison 97

FIGURE LIST

Page number

Figure 6.1: HHsearch Procedure Chart, first 6 steps 38
Figure 6.2: HHsearch Procedure Chart, last 3 steps 39
Figure 6.3: Legend of HHsearch Procedure Chart 39
Figure 7.1: A sample ROC curve 63
Figure 8.1: Bar graph depicting number of sequences per selected GO classes 70
Figure 8.2: Bar graph depicting number of sequences per selected GO classes with 70% sequence identity 72
Figure 8.3: Venn diagram showing sequence identity variation with respect to sets, for 5-class GO data 76
Figure 8.4: Venn diagram showing sequence identity variation with respect to sets for 27-class GO data 79
Figure 9.1: ROC and AUC obtained by search score of HMMER output 81
Figure 9.2: ROC and AUC obtained by e-value score of HMMER output 81
Figure 9.3: HHsearch results with no SS contribution 83
Figure 9.4: HHsearch results with predicted SS contribution weight $\alpha = 0$ 83
Figure 9.5: HHsearch results with predicted SS contribution weight $\alpha = 0.25$ 83
Figure 9.6: HHsearch results with predicted SS contribution weight $\alpha = 0.50$ 83
Figure 9.7: HHsearch results with predicted SS contribution weight $\alpha = 0.75$ 84
Figure 9.8: HHsearch results with predicted SS contribution weight $\alpha = 1.00$ 84
Figure 9.9: HHsearch results with actual SS contribution weight $\alpha = 0$ 85
Figure 9.10: HHsearch results with actual SS contribution weight $\alpha = 0.25$ 85
Figure 9.11: HHsearch results with actual SS contribution weight $\alpha = 0.50$ 86
Figure 9.12: HHsearch results with actual SS contribution weight $\alpha = 0.75$ 86
Figure 9.13: HHsearch results with actual SS contribution weight $\alpha = 0.75$ 86
Figure 9.14: ROC curves and mean AUC values obtained using simple score of PRC in KNN classifier 88
Figure 9.15: ROC curves and mean AUC values obtained using coemission score of PRC in KNN classifier 88
Figure 9.16: ROC curves and mean AUC values obtained using reverse score of PRC in KNN classifier 88

SAKLI MARKOV MODELLERİ KULLANARAK PROTEİN FONKSİYON ÖNGÖRÜSÜ

ÖZET

Crick ve Watson'ın moleküler biyolojide devrim yaratan DNA'nın yapısı keşiflerinden bu yana genom ve proteom çalışmaları biyokimya ve genetiğin sınırlarını çoktan aştı. Araştırmalarda karşılaşılan problemlerin çözümünde laboratuar deneylerinin yanısıra hesaplamalı deneylerin gerekliliği son 25 yılda sıçrama yaptı. Dolayısı ile hesaplamalı bilimlerin temel yöntemleri, gerekli dönüşümlerle biyoinformatik başlığı altında yeni bir araştırma alanında ve disiplinde toplandı. Bugün biyoinformatiğin en popüler problemlerinden biri protein fonksiyon öngörüsüdür. Amino asit dizisine ve ikincil yapıya dayalı bu öngörü çalışmalarında, katar hizalama ve saklı Markov modelleri (HMM), anahtar rol üstlendi.

Saklı Markov modellerininin biyoinformatik alanında kullanılmaya başlanması ile üzerine düşülen konu HMM profilleri olmuştur. Saklı Markov modellerinden önce çoklu hizalama yöntemleri ile üretilen profiller, bu modellerin kullanılması ile daha başarılı ve yüksek doğrulukla üretilmeye başladı. Uzak homoloji kavramının bu modellerle çalışmalara dahil edilmesi bu sayede gerçekleşti. Uzak homoloji üzerine geliştirilen araçlar ve bu araçların kullandığı diğer araçların başında, HHsearch (HMM HMM search), PRC (Profile Comparer), SAM (Sequence Alignment Modelling), HMMER gelir. Bundan başka, saklı Markov modelleri ile doğrudan bağlantıları olmasa bile bu araçların kullandığı diğer önemli araçlar ise PSI-BLAST (Position Specific Iterative Basic Local Alignment Search Tool), ClustalW'dur.

Bu çalışmada HMMER, profil-dizi kıyaslaması yoluyla benzerlik matrisi üretiminde, HHsearch profil-profil kıyaslaması yoluyla benzerlik matrisi üretiminde, PRC yine profil-profil kıyaslaması yoluyla benzerlik matrisi üretiminde kullanıldı. Bu yöntemlerde gerekli yerlerde PSI-BLAST, ClustalW ve Kalign, hizalama ve demetleme yöntemleri için kullanıldı. Veri olarak Protein Data Bank veritabanınıdan Gene Ontology'ye bağlı olarak oluşturulan 5 sınıflı protein veritabanı, yine aynı veri kümesinin zenginleştirilmiş sürümü ve NR veri kümesi kullanıldı. Benzerlik matrislerinin üretiminin neticesinde elde edilen veri, örüntü tanıma tekniklerinde kullanıldı. Bu teknikler, kNN (k-nearest nereast neighbor), tNN (threshold Nearest Neighbor) ve En-iyi-uyuşma (Best-Match) yöntemleridir. Sıralanan tüm bu yöntemlerin birleşimi ile ortaya çıkan prosedürlerin başarımı ROC (Receiver Operating Characteristics), AUC (Area Under Curve) ve doğruluk ölçümleri ile değerlendirildi.

5 sınıflı veri kümesi için dizi-profil ve profil-profil kıyaslamasının katar hizlama yöntemlerinden daha kötü sonuç verdiği bulundu. İkincil yapının HMM'de hesaba katılmasının fonksiyon öngörüsünde faydalı olduğu görüldü. NR veri kümesi ile zenginleştirilmiş veri kümesinin profil üretiminde faydalı olduğu görüldü.

PROTEIN FUNCTION PREDICTION USING HIDDEN MARKOV MODELS

SUMMARY

Since Crick and Watson's revolutionary discovery of DNA structure in molecular biology, the genome and proteome studies have already overpassed the boundaries of biochemistry and genetics. The computational experimentation, beside laboratory experiments executed in solution, became a necessity within the last two decades. Consequently, basic methods of computational sciences studying molecular biology, genome and protein sciences are collected under the bioinformatics title. Today, one of the most popular problems of bioinformatics is the protein function prediction. In the study of this prediction problem, the key roles are assigned to use of string alignment techniques and hidden Markov models on amino acid sequence and secondary structure of proteins.

The profile-HMM's became popular with the use of hidden Markov models in bioinformatics. Profiles, which were conventionally produced using alignment methods, became more accurate and successfull by means of hidden Markov models. As a consequence, remote homolgs were included into function prediction studies with these models. Well known tools developed on remote homology are HHsearch, PRC, SAM and HMMER. In addition, PSI-BLAST and ClustalW are used by these tools in preprocessing steps.

In this study, HMMER is used in sequence-profile comparison, HHsearch is used in profile-profile comparison, PRC is used in profile-profile comparison for similarity matrix production. PSI-BLAST, ClustalW and Kalign are used in alignment and clustering steps. As the data set, 5-class protein database generated from Protein Data Bank database with respect to Gene Ontology Annotation is used. In addition, its variant, the enriched data set and NR data set are used. The similarity matrices produced by HMMER, HHSearch and PRC methods are used as inputs to machine learning techniques. These techniques are kNN, tNN and Best-Match. The accuracy of different similarity matrix production methods are evaluated using ROC, AUC and accuracy measures.

For the 5-class data set used, it is found out that sequence-HMM-profile and HMM profile-profile methods cannot perform as well as sequence aligment techniques. It is also found out that using secondary structure in addition to the amino acid sequence helps with protein function prediction. Enrichment of data set with NR data is found to help with function prediction.

1. INTRODUCTION

Protein is an organic molecule made by residues which are sequentially ordered in a chain, linked with peptide bonds. Proteins can contain varying number of residues, consequently there are infinitely numerous variations of protein. A residue is the unit element of a biological sequence, hence a residue in a protein means an amino acid, whilst a residue of a gene sequence is a nucleic acid. There are 20 types of amino acids which make up a protein sequence. Every amino acid, beside its chemical features, provides different geometrical fold to the sequence in which it is located **(Tramontano, 2007a).**

Proteins are the most elementary functional units. The function of a protein is the result of its 3D form and chemical features which are defined by its residues **(Tramontano, 2007a)**. A protein is studied in multiple structures: Primary structure which is amino acid sequence, secondary structure which is the most basic shape of the protein -can be described as shape at the chain level-, tertiary structure which is the overall 3D structure of a protein, and finally quaternary structure which is the shape of protein modified by its interactions with other functionally related proteins.

One of the easiest ways of structure detection of a protein is sequencing via mass spectrometry **(Tramontano, 2007a)**. Fast sequencing increases the knowledge about sequences of proteins but 3D knowledge does not grow as fast as that level. Thus, knowledge about most of the discovered proteins consists of only sequence data. Further information about a protein is obtained for a small subset of whole observed proteins. In addition, the functional knowledge of proteins is even rearer. Consequently, computational inference methods for prediction of secondary, 3D structure or function,

are even more important than they have ever been before **(Wallqvist et al., 2000)**. Computational methods are intended to find out experimental knowledge such as class, family, fold, superfamily, function, tertiary structure, secondary structure, using only original sequencing data which is quiet cheap to obtain.

The relation between different levels of protein structure, i.e. sequence and secondary structure or even secondary structure and tertiary structure, is highly complex and depends on a large set of conditions. Hence, an approximation of the higher structural information is used for computational discovery. The general idea of computational methods is to profit from experimental protein data as much as possible.

Similarity is one of the most often used features to find out relations between known and unknown proteins. Sequence alignment methods are conventionally used to measure the similarity between proteins. Different groups of methods are proposed to align sequences according to a local (e.g. Smith-Waterman), global (e.g. Needleman-Wunsch), pairwise (e.g. BLAST) or multiple alignment (e.g. ClustalW).

A further step is using profile Hidden Markov Models to detect similarities between proteins on the basis of protein family. Profile-HMM's, in contrast with former methods, includes features extracted not from a single protein but from a set of proteins. Hence, they are more capable of detecting common properties in protein set.

2. COMPUTATIONAL BIOLOGY AND BIOINFORMATICS

Computational biology and bioinformatics intend to solve problems of molecular biology and biochemistry using techniques and methods from computer sciences, engineering, mathematics, statistics, informatics and artificial intelligence **(Nair, 2007)**. As the subtopics of artificial intelligence, machine learning, data mining and pattern recognition techniques are mostly used.

The major study areas of computational biology and bioinformatics are amino acid sequence, secondary structure, similarity metrics and alignments. They aim to find relations using different similarity metrics, various alignment methods, on the amino acid sequences, secondary sequences and tertiary structures. Some of the contemporary problems dealt in bioinformatics are similarity measures, classification, family and superfamily detection, fold recognition, secondary structure prediction, function prediction and homology detection.

For the completeness of this thesis, we give more details on amino acid sequence, secondary structure, similarity metrics and sequence alignment below.

2.1 Amino Acid Sequence

Amino acid sequence (AAS) or peptide sequence is a macro molecule constructed by linearly connected amino acids, forming peptide chain, resulting in a protein. There are 20 types of amino acids. **(Petsko et al., 2004a)** Those are from Alanine, Arginine to Valine. There's no limit in the length of protein sequence, thus, theoretically, a sequence can be inifinitely long. In addition, each residue can follow any other residue. Hence a

subsequence of length n can have 20^n variations, and theoretically, there are infinite variations of a protein since no boundary exists for the length of the sequence. **(Petsko et al., 2004b)**

From the point of view of computer sciences, AAS is a string of characters that come from a 20 letter alphabet:{A,R,N,D,C,E,Q,G,H,I,L,K,M,F,P,S,T,W,Y,V}. Every information that can be extracted from the AAS must be related to the string of this 20 different characters.

In databases, which will be mentioned in section 8.1, AAS information is mostly stored in FASTA format. This format contains, sequence representation, with a single character per residue of the sequence, and a header information per sequence. **(Tramontano, 2007b)** The header of a sequence is restricted to a single line. It can contain, name, identity, source, etc. All lines in FASTA format is recommended to be limited with 80 characters. Hence there is no restriction in the number of lines reserved for a sequence **(NCBIa)**. A sample sequence in FASTA format is given below:

```
>1FBV:A
PPGTVDKKMVEKCWKLMDKVVRLCQNPKLALKNSPPYILDLLPDTYQHLRTILSRYEGKM
ETLGENEYFRVFMENLMKKTKQTISLFKEGKERMYEENSQPRRNLTKLSLIFSHMLAELK
GIFPSGLFQGDTFRITKADAAEFWRKAFGEKTIVPWKSFRQALHEVHPISSGLEAMALKS
TIDLTCNDYISVFEFDIFTRLFQPWSSLLRNWNSLAVTHPGYMAFLTYDEVKARLQKFIH
KPGSYIFRLSCTRLGQWAIGYVTADGNILQTIPHNKPLFQALIDGFREGFYLFPDGRNQN
PDLTGLCEPTPQDHIKVTQEQYELYCEMGSTFQLCKICAENDKDVKIEPCGHLMCTSCLT
SWQESEGQGCPFCRCEIKGTEPIVVDPF
```

2.2 Secondary Structure

Secondary structure[1] is a general expression used for biological sequences such as DNA, RNA or protein. For each residue, it defines the position in 3D space in a small

[1] SS will be used henceforth instead of secondary structure

neighborhood, i.e. relative to preceding or following residues, not with respect to the whole sequence **(Kabsch and Sander, 1983).** The absolute position in 3D space is the information held in tertiary structure.

Secondary structure is expressed as a string, just like AA, but comprising elements from a different alphabet. There are two major SS conventions: DSSP and HEL.

DSSP is proposed by Kabsch and Sander in 1983. Three different helix state is represented by G, H and I. 2 Different states are represented by B and E and finally the rest of the states are represented by S and T **(Kabsch and Sander, 1983).**

Another major convention is HEL which is also mentioned by Kabsch and Sander in **(Kabsch and Sander, 1983)**. In this convention, the character H represents all kind of helix regions, whereas E represents sheet areas and L represents loop sections.

For each residue in AAS, a representing SS character is placed in SS sequence. Hence, each residues SS can be found in SS sequence with the same index as in AAS. Consequently, AAS and SS are two sequences of the same length.

Although SS can be obtained experimentally as the most definitive way; there are some approximation methods to extract SS. One of the most successful tools is PSIPRED which requires a multiple local alignment, and a query sequence in FASTA format. It requires alignment input since it works with an approximation method.

To store SS data, FASTA format is also commonly used. In ITU Bioinformatics Project, some experiments required SS in FASTA format. The SS information in FASTA format, is analyzed using tools written in Matlab, Python, Java and C, by project members E. Aygun, A. Filiz and C. Komurlu **(Aygun et al., 2008) (Filiz et al., 2008).** A sample secondary structure in HEL and FASTA format is given below:

```
>1E12:A
CCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHTTCCTTHHHHHHHHHHHHHHHHHHHHHHHHHHHHTTTCEEEEB
```

```
CTTSTTBTCEEEECHHHHHHHHHHHHHHHHHHHHHHHTCCHHHHHHHHHHHHHHHHHHHHHHCCSCHHHH
HHHHHHHHHHHHHHHHHHHHTHHHHHHHHHTCHHHHHHHHHHHHHHHHHHHHHHHHSTTTTCSSCCHHHHH
HHHHHHHHHHTHHHHHHHHHHHHHTHHHHHCCCCCCCCCCCCC
```

2.3 Similarity Measures

All structures and features of a protein, basically, depend on its AAS. They are also affected by environmental conditions such as pH, temperature, interactions with other proteins. But the major parameter is AAS. Yet, these relations with AAS are much further complex than they could be modeled and let scientists acquire further information using only AAS.

From an abstract point of view, since almost all features of a protein are functions of AAS, theoretically, proteins that have similar AAS, are expected to have similar structures, such as SS or tertiary structure, which means, the closer are AAS of two proteins the closer should be their 3D shapes. Consequently, measuring the similarity between an unknown -only AAS is known- protein and known proteins, results in approximate information about higher structures of the first protein.

There are two major methods to measure similarities of proteins: Sequence alignment methods and hidden Markov models. The oldest is the first one which relies on string comparison and scoring using substitution costs. The second one which is now as famous as the alignment approach, uses hidden Markov models and compares proteins or protein clusters via these models.

Sequence alignment methods are the most used sequence comparison techniques. The basic idea behind alignment is to match residues, one-to-one. According to compatibility of the two subject sequences, a score is obtained. This score evaluates how similar these two sequences are with respect to the alignment scheme **(Durbin et al., 1998a)**. Another dominating idea in the basis of sequence similarity is that, no protein appeared on itself

out of nowhere. Indeed, every protein is evolved from common ancestors with other proteins. This evolution comprehends morphing in the level of residues via insertion, deletion and substitution. Hence, these modification behaviors are taken into account in alignments. They have also emphasized roles in homology studies **(Durbin et al., 1998b)**.

2.4 Sequence Alignment

Sequence alignment is a procedure applied to modify sequences with algorithmic methods in the purpose of uncovering similar or corresponding sections pointing to evolutionary, structural or functional relations between proteins. There are various approaches in sequence alignment. The two general distinction of methods are pairwise and multiple alignment. Pairwise alignment methods operate on a pair of sequence, and try to extract similarity between these two sequences, referring to only these two sequences' residues. There is no external sequence information, or other structural information about these two sequences. Multiple sequence alignment methods try to discover sequence regions that are important for a set of proteins and to ignore similar regions that are not widespread. In this second approach, the idea of having common ancestors which is indeed homology, similarity regions that are common in multiple proteins points to functional or structural relations, since the sites resulting in these common functions or structure properties are thought to be kept during evolution process **(Chan et al., 1992)**.

Another sequence alignment categorization is local or global alignment approach. In the global alignment, each sequence's start and end are included in the alignment. Hence, a region's effect in alignment depends on the relative position with respect to start residue. On the other hand, in local alignment, this is not the case. A region's content in the meaning of residue, and these residues' order does matter in alignment scoring **(Chan et al., 1992).** These approaches are covered in the following subsections.

2.4.1 Pairwise Alignment

2.4.1.1 Global Alignment – Needleman-Wunsch Algorithmic

Given two amino acid sequences, this alignment method matches residues of these sequences such that, the alignment is continuous, the order of residues is never violated and gaps are allowed. Continuity means that, there's no interruption in alignment, each residue is matched either with a residue of the other sequence or a gap. What is meant by order of residues is that, for each aligned pair of residues x_i and y_j, where x_i is i^{th} residue from sequence x and y_j is the j^{th} residue from sequence y, any residue located at the left section of x_i is allowed to be aligned with residues of y located at left section of y_j or with a gap, and vice versa. The same rule is valid for right section also. The most common algorithm as global alignment is Needleman-Wunsch. The algorithm is built upon a score matrix. The rows of the matrix are identified by residues of one sequence of two, and the columns are identified by residues of the other sequence. The matrix is filled recursively starting from top left to the bottom right. Each cell F(i,j) is filled with the function given below:

$$F(i,j)=max\{F(i-1,j-1)+s(x_i,y_i),F(i-1,j)-d,F(i,j-1)-d\} \quad \textbf{(2.1)}$$

where $s(x_i,y_i)$ is the substitution score of residue i from sequence x and residue j from sequence y; and d is gap penalty. Assigning $F(i\text{-}1,j\text{-}1)+s(x_i,y_j)$ to $F(i,j)$ means that residues x_i and y_j are aligned. Or, if $F(i,j)$ is assigned $F(i,j\text{-}1)\text{-}d$ means that residue x_i is aligned with a gap, likewise if $F(i,j)$ is assigned $F(i\text{-}1,j)\text{-}d$ means that residue y_j is aligned with a gap. When all cells from top left to bottom right are filled with maximum scores possible, all the way back from bottom right to top left is followed. This gives the best global alignment solution **(Durbin et al., 1998b)**.

2.4.1.2 Local Alignment – Smith-Waterman Algorithm

In the previous section, Needleman-Wunsch algorithm is described for global alignment.

For local alignment, most common algorithm is Smith-Waterman. There are three basic differences in this algorithm with respect to Needleman-Wunsch algorithm: The first difference is that, partial alignment is allowed, i.e. subsequences can be aligned. Secondly, the constraint of keeping the order of residues along the sequence where they are located is omitted. As a consequence, for instance let s_i and s_j be two subsequences of sequence x located respectively and t_k and t_l are two subsequences of sequence y located respectively. Note that in a global alignment, if s_i is aligned with t_l, alignment of s_j with t_k is not allowed. But in local alignment, this option is not forbidden. The second difference is made in function *F*(*i,j*).

$$F(i,j)=max\{F(i-1,j-1)+s(x_i,y_i),F(i-1,j)-d,F(i,j-1)-d,0\} \quad \textbf{(2.2)}$$

Note that a new 4th case is added in max() function. If an alignment reaches a negative value, it can be assigned zero. This means that if a partial alignment's score gets negative, it can be stopped. A new local alignment can be initiated.

The third difference is that, the alignment does not start at bottom right cell, it starts at the best scoring cell. Then it goes until the score gets 0 towards top left cell. Then it restarts with a positive score again **(Durbin et al., 1998b)**.

2.4.2 Multiple Sequence Alignment

In the previous section, the problem of how much two sequences are related or similar, is dealt. The defect in this approach is that comparing to sequences does not give enough information about structural or functional similarity, since the similar sections can be functionally important or not. A different approach to this problem is that if a subsection or close variants are commonly found in sequences, it can be functionally or structurally important. Multiple sequence alignment tries to align sequences at once. This way, it can find sites which are common in many sequences **(Lindahl,2000)**.

Multiple sequence alignment is the method of similarity detection in a set of proteins

having more than two members **(Chan et al., 1992)**. This procedure let find out common conserved regions in a group of proteins, guessed to be related by evolution. This method also allows to build phylogenetic trees. Generally the problem of multiple sequence alignment problems are considered as difficult and various techniques gathered under this method are classified as NP-complete optimization problems **(Wang and Jiang, 1994)**.

There are 5 major approaches in multiple sequence alignment techniques. The first is Dynamic Approach. This method applies pairwise alignment method to multiple sequences directly. Remember that there is a 2 dimensional alignment matrix inside which the alignment path is detected, and each dimension refers to one of the sequences. In multiple sequence alignment, a n dimensional matrix is used for alignment path, of which each dimensionality refers to one of the n sequences. Note that high time and space complexity makes these techniques quite expensive to apply. However, they guarantee the global optimal solution **(Lipman et al., 1989)**.

The second approach is progressive methods. This approach, instead of processing the whole sequence set at once, it starts with aligning two most similar sequence by means of a pairwise alignment method. Once a pairwise alignment is obtained, then rest of the sequence set is added to this alignment, one by one. The main handicap of this approach is that it is fairly dependent on the accuracy of initial pairwise alignment. In a case of low accuracy, the whole procedure is affected. There are many successful studies on the variation of Clustal as a variation of progressive method such as **(Higgins and Sharp, 1988),(Thompson et al., 1994)**, **(Chenna et al., 2003)** .

The third approach is iterative methods. This approach tries to recover the deficient point of the previous approach. It forms an objective function over initial global alignment. and tries to optimize it by means of iterating over that function. It realigns sequence subsets at each step **(Hirosawa et al., 1995)**.

The fourth approach is motif finding or profile analysis. It starts with a global multiple alignment over a subset of sequence. Therefore, it detects important regions which are indeed motifs, and it builds a motif matrix. The substitution matrix is reformed according to distribution of residues in motifs but not according to general distribution. Then finally, these motifs are searched in the rest of the alignment set.

The fifth partition is in fact a group of diverse approaches. This group is formed of computer science originated techniques. Mostly those are optimization methods like simulated annealing or genetic algorithms. Another techniques is Hidden Markov Model which underperformed in early applications but now has successful derivations **(Karplus et al., 1998)**.

2.5 Secondary Structure Prediction

Secondary Structure of a protein is described in section 2.2. The main idea is that, secondary structure of a protein is a function of amino acid sequence, i.e. given an amino acid sequence, an exact secondary structure exists and depends on amino acid sequence. Hence most of the secondary structure detection studies are based on this approached and called prediction. Two different main approaches. The earliest is based on predicting the secondary structure of a protein, only with respect to its own amino acid sequence. This approach has 60% accuracy in HEL convention. The second approach profits from multiple sequence alignment which gives evolutionary information thus structure intention of proteins. It gives better results with accuracies at 80% **(Kabsch and Sander, 1983)**, **(Branden and Tooze, 1999)**.

There are 3 computational methods in secondary structure prediction: Neural Networks, Hidden Markov Models and Support Vector Machines. The most commonly used tool, PSIPRED uses artificial neural networks approach. It produces two outputs. First is the predicted secondary structure and the second is the confidence array. This array contains natural numbers between 0 and 9. Each number corresponds to a residue in predicted

secondary structure sequence, and it represents how reliable the prediction is. The predicted secondary structure residues are computed using 3 outputs of neural network. In best case only one of the three outputs is expected to be 1 and the rest are 0. Each output is dedicated to one of 3 structures: H, E, L. In general cases, the greatest output is less than 1 and others are greater than 0. Even though, the greatest outputs corresponding structure is assigned to the current residue. The confidence value is computed as the difference of two greatest output of neural network. The higher the difference, the higher the confidence. PSIPRED works in two different modes which are two prediction approaches described recently. In the first mode, the accuracy is fairly low. In the second case where it uses a multiple sequence alignment, the accuracy is satisfying **(Jones, 1999)**.

2.6 Profile Analysis

Profile Analysis is a method developed to detect distant relationships between protein sequences via sequence comparison. The comparison is not made using only mutational distance matrix but also using alignments of proteins from the same families. A profile is a position specific scoring matrix which is of dimension $N \times 21$. Here 21 is the number of amino acid types and an additional column is for insertion/deletion penalty. N is the length of the MSA. This way, a specific score for each residue at each position and also for insertion and deletion can be defined in this matrix **(Gribskov et al., 1987)**.

The similarity between sequences can be valid on the level of subsection, according to some bioinformaticians. Some regions in proteins are quite common in protein families. Those regions are called, sequence motifs. A major approach to find these motifs is to represent them as alignment of sequences containing these motifs and then generate a sequence profile. The basis of the sequence profiling method is that, aligning sequences from the same family or sequences having same motifs can help extract information about this family or about those common motifs **(Lüthy et al., 1994)**. According to

Lüthy *et al*. **(1994)** a single sequence cannot provide these two major information which can be extracted using alignments: Conservation of residues and mutations. This two throughput are useful to check whether a new instance belongs to a protein family or contains sequence motifs. Note that a profile can be aligned to a sequence, using exactly sequence- sequence comparison methods. If the query sequence belongs to the family which forms the profile or if it contains similar motifs, the comparison results high scores, otherwise low **(Lüthy et al., 1994)**.

3. PROTEIN FUNCTION PREDICTION

3.1 Definition of Protein Function

As the first step, definition of protein function must be given. There are many levels for the definition of protein function from biochemical function as the lowest level to biological process, up to organ function and in addition, organism function **(Watson et al., 2005)**. According to Watson *et al.* **(2005)**, there are many approaches on function definition and unfortunately there are not exact criteria to verify these approaches. Two most famous schemes are Enzyme Commission and EcoCyc. A relatively new scheme is Gene Ontology **(Ashburner, 2000)**. The significant feature of Gene Ontology (GO) is machine readability. This feature makes it very common in Bioinformatics area **(Watson et al., 2005)**.

3.2 Diverse Approaches in Protein Function Prediction

According to Huynen *et al.*, genome sequencing provides an immense set of protein sequences but structural or functional information of a major subsection of the set is missing. Homology detection approach, as it uses the complete sequence of instances is appropriate for protein function prediction **(Huynen et al., 2003)**.

Watson *et al.* classifies, function prediction methods into three general approach. The first is sequence-based methods. Typical tools in this section are BLAST or FASTA which are direct sequence-sequence comparison tools. Following them, some profile methods are proposed like PSI-BLAST. Also, profile-HMM methods are also suggested like SAM and HMMER. Note that profile methods are built on the basis of information

retrieval using a set of protein. This leads to cases of using protein families. Hence the 3D structure, superfamily and remote homology searches are included in the scope of profile techniques. Since these techniques are sensitive to conservation of residues, they outperforms direct sequence-sequence comparison techniques. A better subsection is the section of profile-profile techniques such as HHsearch **(Watson et al., 2005)**.

The second category according to Watson *et al.* **(2005)** is formed by structure-based methods. Those methods can be involved when sequence-based methods are impotent and function-related information can be achieved via other ways. Note that structure-based prediction methods can vary according to structure level. Generally, those methods can be clustered under, fold-matching methods, surface clefts and binding pockets, residue template methods, DNA-binding methods and phylogenetic relationships, and machine learning techniques. Only the names are given about these subcategories except the last-one, since it is out of scope of this study. Under machine learning title, known statistical methods, data mining techniques and machine learning algorithms lay. In Watson *et al.*'s **(2005)** opinion, much endeavor has been spent for prediction from 3D structure. In these works, GO Annotation **(Camon et al., 2004)** is used due to its definite hierarchy. The problem about the machine learning techniques is that they perform well on the training data, however they are not as good in test data **(Watson et al., 2005)**. Watson *et al.* **(2005)** claims that, all methods, individually are not successful enough. There are still some cases that those methods are incompetent. Hence, combining those methods can solve more complex problems. The last category is diverse combinations of methods categorized so far **(Watson et al., 2005)**.

As stated above, for protein function prediction problem, GO Annotation hierarchy is used. The general description of the problem can be expressed as, generating a classifier model by means of machine learning techniques on protein data and classifying protein instances on GO Annotation classes.

3.3 Homology and Residue Conservation

Homology is a general term used in science, it is not specific to bioinformatics. As the general meaning, homology is the relationship of two elements which descend, diverging from a common ancestor element **(Fitch, 2000)**. In molecular biology, homology is understood as sequence similarity, beside its meaning of common ancestor. The general hypothesis is that, if two proteins have similar subsequence or sequence, it is likely that they are homologous. However, the sequence similarity does not always imply homology, since short sequences can have similarity by chance and long protein sequences can have similarity due to the fact that they share a common function in biological system **(Dewey and Patcher, 2006)**. During the evolution of species in nature, their genes and, as a consequence, their proteins have evolved. Some proteins are mostly conserved and some others are mostly deformed. Some amino acid positions in peptide sequence having important role in proteins 3D structure or in its function are under hard evolutionary conditions; which means that modification on these positions are fairly rare. Hence, the structural or functional importance of a residue can be related to its conservation during evolution process **(Landau et al., 2005)**. As its stated recently, the conservation and homology detection can help in accurate function prediction. As the functional sites are conserved during the evolution, they can be evidences of homology, i.e. if two proteins are homologous, we can detect this relationship with the help of these common conserved residues, and also we can assume that these conserved residues are functionally important. As a consequence, homology relations can help finding out functional similarities. There are some techniques to find remote homology between proteins. Mostly discussed methods are profile and profile-HMM techniques. Profile-HMM's are used most generally for fold recognition or family and superfamily detection, remote homology detection **(Lindahl and Eloffson, 2000)**. Profile-HMM is proven to be a good method to solve those problems. They can be used in function prediction, following the relation between homology and conservation.

4. PREVIOUS WORK

In this section, studies pioneering to this study are reviewed briefly. The review begins with sequence-based tools and goes on with profile and profile-HMM generating tools then finally adverts to profile-HMM comparison studies which are actually deeply held in section 6.

SSEARCH, written by William R. Pearson, is the initiating tool which makes pairwise alignment using Smith-Waterman algorithm. It can compare biological sequences which means that, it can handle both gene sequences and amino acid sequences. Although it is claimed to be the most sensitive tool, it is very slow **(Brenner et al., 1998)**.

FASTA is next alignment tool, written again by Pearson. It may run in two different modes. Either with high speed but less accuracy or slowly but with high accuracy **(Brenner et al., 1998)**.

Basic **L**ocal **A**lignment **S**earch **T**ool, or as known as BLAST, works with a new approach, written by Altschul *et al.* **(1990)**. First, it uses some heuristics to search a sequence database. Hence it does not guarantee the optimal solution, but guarantees a satisfying one in plausible time **(Altschul et al., 1990)**. The major contribution of BLAST is that it lets the user to search a database quicky. According to Brenner *et al.* **(1998)**, SSEARCH and FASTA results with better accuracies but BLAST works fairly fast. In this study, PSI-BLAST which is an improved version of BLAST is used.

ClustalW, is a multiple sequence alignment tool. It receives a set of sequences, does pairwise alignment with each pair of the set, builds a phylogenetic tree, then finally using this phylogenetic tree, it generates the multiple sequence alignment

(Chenna et al., 2003). In this study, ClustalW is never used independently. Neither it is assessed via evaluating its results. ClustalW supports the input format of HMMER. Hence, at a subset of methods, ClustalW is used as a former step before profile-HMM generation using HMMER.

Another multiple sequence alignment study is Kalign written by Lassman and Sonnhammer. It's mainly compared with ClustalW. It obtains better results and runs faster **(Lassmann and Sonnhammer,2005)**. In this study, Kalign is used instead of ClustalW, when it is noticed that ClustalW runs not fast enough for experiments. Since it's claimed that Kalign is faster and results better than ClustalW, with quick verification tests, Kalign is used instead of ClustalW in some profile-HMM methods where HMMER is used.

Up to now, sequence comparison tools are presented. A further step is generating profile-HMM using sequence sets. There are 3 major profile-HMM producing tools. Sean Eddy's HMMER, Johannes Söding's HHsearch and Karplus *et al.*'s SAM. HMMER is the most famous one. It produces profile-HMM's using plan 7 architecture. It requires a multiple sequence alignment output as input. It can also make sequence-profile comparison on a search database in both case, either given a sequnce it can do it over a profile-HMM datase or given a profile-HMM it can do over a sequence database. Further information about HMMER can be found in section 6.2.

The second tool is **S**equence **A**lignment and **M**odeling software: SAM. It's written by Karplus *et al*. . It builds profile-HMM's itself. In addition, it can use secondary structure information as a contribution on profile-HMM generation. According to Madera, SAM results better comparing with HMMER, given identical inputs **(Madera, 2002)**. Nevertheless, HMMER is preferred since it is referred more and it can make sequence-profile comparison.

Another profile-HMM extraction tool is Söding's HHsearch. It works with PSI-BLAST

to obtain amino acid sequence alignments and with PSIPRED when secondary structure information is considered as a contribution. HHsearch not only generates profile-HMM's but also compares 2 profile-HMM's **(Söding, 2005)**. It is a very good study, and it has been cited a lot. This study is detailed in section 6.3.

The last study included in this section is Martin Madera's **PR**ofile **C**omparer, PRC which is his PhD study. He released a tool named Profile Comparer at the end of his thesis study in 2002, and maintained the tool until the end of 2005. It's a profile-profile comparison tool based on Hidden Markov Model. The architecture is developed by Madera and called pair-HMM. It takes two profile-HMM and applies a pairwise HMM comparison then puts out some scores. It supports various data format such as FASTA format, PSI-BLAST output, HMMER and SAM also **(Madera, 2002)**. Further information is given in section 6.4.

5. HIDDEN MARKOV MODELS

Hidden Markov Model (HMM) is a model such that it simulates probabilistic rules included in a character sequence having a probabilistic behavior. Within this model, a finite state machine is produced, and for each transition, probability values are defined. In Markov process, the observed sequence is the real sequence itself. In this case, computation of transition probabilities is quite easy. In hidden Markov model, the observed sequence does not need to be, one-to-one, the real sequence; states can emit different symbols with respect to a defined probabilistic model. Taking this situation into consideration, beside transition probabilities, emission probabilities are added to Markov model, and the model is extended to hidden Markov model **(Eddy, 1996)**.

A very common problem in HMM's is given a model, what is the most probable path? Viterbi algorithm is suggested to solve it. The core relation of Viterbi is:

$$v_l(i+1) = e_l(x_{i+1}) \max_k (v_k(i) a_{kl}) \qquad \textbf{(5.1)}$$

Here, $v_k(i)$ is the probability value of most probable path from start to state k with emission i . Term a_{kl} is the transition probability from state k to l, and $e_l(x_{i+1})$ is the emission probability of symbol l in sequence element x_{i+1} **(Durbin et al., 1998b)**.

6. PROFILE HIDDEN MARKOV MODELS

6.1 Hidden Markov Models in Biological Sequences

HMM's are used for biological sequences in this way: Mostly agreed usage is modeling, insertion, deletion and matching cases with corresponding states in HMM. HMM's can be used for both pairwise similarity of proteins or profile-profile similarity measure. [Durbin1998]

For profile-profile similarity measure, sequence profiles indicating how frequent matching states are in a multiple sequence alignment. Basic principles in such a usage is established on occurrence of matching (M), unmatching (U), insertion (I) and deletion (D) events between two protein sequences. Coarsely, the number of M for given two sequences indicates how similar to each other, these two sequences are. Through HMM's ability of modeling insertions and deletions, HMM's performance were found out better than sequence- profiles and PFAM **(PFAM, 2002)** databases containing HMMER **(Eddy, 1998)** profiles were set up. There are some proposed tools such as HMMER **(Eddy, 1998)**, HMMSTR **(Bystroff et al., 2000)**, HHsearch **(Söding, 2005)**, that can be used to produce profiles. Among them, HHsearch is chosen in this work due to its performance of homology detection **(Cheng and Baldi, 2006)**,**(Söding, 2005)**, its ability to use actual SS or predicted SS obtained via PSIPRED and the fact that it's frequently updated by its writer Söding.

One of the most well-known software developed for the purpose of producing profile-HMM is Sean Eddy's HMMER **(Eddy, 1998)**. Another software which uses HMM's is Sequence Alignment and Modeling System (SAM) **(Hughey et al., 2003)**. It models,

matching, insertion, deletion and unmatching states, as mentioned above.

Another work related to profile-HMM is SCOOP software. In contrast with other profile comparison tools, this software does not compare profile-HMM's of two subject proteins, but it compares each profile-HMM through a complete profile-HMM database, and it computes the total common area matching both of the profiles separately. Finally it produces a similarity score based on the inference that it makes using comparison of two profile searches **(Bateman and Finn, 2007)**.

An additional work which uses profile-HMM's is Profile Comparer (PRC) which aligns profile-HMM's. In order to produce these profile-HMM's, it uses 2 null models. Then it models the cases of matching, deletion and insertion states which were mentioned above. **(Madera, 2005)**

Similar to the situation of Transductive SVM's **(Joachims, 2003a)**, **(Joachims, 2003b)**, it was decided to produce similarity matrix for proteins using AAS and SS informations.

6.2 HMMER – Biosequence Analysis Using Profile-HMM

6.2.1 Introduction to HMMER

HMMER written by Sean Eddy is another profiling tool using hidden Markov models. It produces profile-HMM's using multiple sequence alignment. Hence, this tool requires multiple sequence alignment tools. As it is expected, not with all but, HMMER works with a couple of MSA tools. It can work with output format of ClustalW, Wisconsin/GCG MSF format and Stockholm format. In this study, HMMER is used with ClustalW **(Eddy, 2003)**.

Using a multiple sequence alignment, HMMER can build one profile-HMM. For this task it has a component: `hmmbuild`. The output of this component is a human readable text file which contains transition probabilities for PLAN 7 **(Eddy, 2003)**.

Similar to HHsearch, Eddy suggests a calibration when a profile-HMM is created using `hmmbuild`. This procedure, again, corrects *e-values*. For the calibration of a profile-HMM, Eddy has added a component tool named `hmmcalibrate`. The output is again a HMM file which can be read by other components of HMMER **(Eddy, 2003)**.

Another skill of HMMER is to create a profile-HMM database using single profile-HMM's which HMM has created. For this task, HMMER has a special parameter that is given to `hmmbuild`. There is a second way to generate a profile-HMM library. Profile-HMM library of HMMER is designed in a way such that the database is formed of actually, concatenation of single profile-HMM files. `hmmbuild` with -A parameter, adds the new profile-HMM to the end of the file dedicated to be profile-HMM database. Hence, in this method, every profile-HMM created via `hmmbuild` is added immediately to the end of the profile-HMM database.

The second method is creating profile-HMM's, each separately, than generating the profile-HMM database by means of concatenating all single profile-HMM's. For this purpose, `cat` command of unix can be used.

HMMER's another skill is that HMMER can make sequence-profile comparison. It is able to do both a profile-HMM search over a sequence database or a sequence search over a profile database. It has two separate components for these processes: `hmmsearch` and `hmmpfam`. By definition given in HMMER user guide, hmmsearch is used for profile-HMM search over a sequence database. And again by definition, `hmmpfam` is used for sequence search over a profile-HMM database **(Eddy, 2003)**. Both of these tools produces a score file as the output. This file contains matchings which exceed a threshold. For each matching, the file contains two indicating values: *score* and *e-value*. *Score* indicates how well the subject sequence and the current profile-HMM are aligned. And *e-value* indicates the degree that the alignment is by chance but not by a homology relation. Hence both indicators can be extracted and used as similarity measure, since the higher the *score* is, the better the sequence and the profile-

HMM alignment is. Similarly, the lower the *e-value* is, the alignment of the sequence and the profile-HMM is weak. In this study, both tools are used but for only the purpose of sequence search over profile-HMM database. Their interesting result is analyzed in section 9, Results and Evaluation.

6.2.2 Function Prediction Procedure Using HMMER

Using the sequence-profile-HMM comparison, in this study two different approaches are developed for function prediction. In the first approach, this comparison is used to generate a similarity matrix by means of comparing each sequence with every profile produced for each sequence in the data set. In the second approach, the clustering idea is implemented. Every functional class is considered as a cluster, then each sequence's similarity to each cluster is computed. Finally this newly formed data set is used.

6.2.2.1 Similarity Matrix Generation Via Neighborhood Based Sequence Profile Comparison

As explained above, HMMER can align a sequence to a profile-HMM. In fact this produces a similarity result. If profile which represents a protein A is compared with a sequence of a protein B, the comparison results a similarity measure. The first issue is that how to generate a profile-HMM which represents a sequence. Remember that a profile-HMM is generated using a multiple sequence alignment. This means that, the profile-HMM generated using this multiple sequence alignment is same for all sequences participated into this multiple sequence alignment. To solve this problem, a method which lets a profile be special for a sequence must be found out. Thus a profile has to include characteristics of a sequence.

A proposition for the solution of this subproblem can be such as: Remember that, a profile-HMM is built using multiple sequence alignment and represents common characteristics of the proteins of the alignment set. The aim is building a profile-HMM given a sequence such that the profile-HMM represents the functional characteristics of

this sequence. Hence if the sequence set is formed of sequences which have generally the common functional features with a subject sequence, then the profile-HMM trained over this set also represents the functional features of the subject sequence. As mentioned in section 3.2, the homology detection is based on common functional sites of a sequence. By hypothesis, the conserved sites during the evolution are functional sites. Hence a multiple sequence alignment which is in fact a homology search results functionally common features of the set. To generate a profile-HMM for a given sequence, a multiple alignment set for this sequence must be formed. Hence, the most similar sequences to the subject sequence can be selected. Here, PSI-BLAST[2] can be used to detect most similar sequences to a subject sequence[3]. Once the similar sequences are detected, then the set is defined. This set, hence, can be used for multiple alignment.

In this procedure, for multiple alignment, ClustalW as a MSA method supported by HMMER **(Eddy, 2003)** is preferred since it was formerly used in other problems of ITU-Bioinformatics project. After building a multiple sequence alignment, the output can be used to form a profile-HMM. Since the members of multiple sequence alignment are most similar to a subject sequence, the profile-HMM represents this sequence.

The steps of the procedure can be resumed in a list:

1. For each sequence in protein data set, search similar sequences using PSI-BLAST. This forms N local alignment search output for the data set of cardinality N. The output of alignment can be called as bla file since the output is stored in bla extended file.

2. Detect the neighborhood of each sequence by means of extracting the ID's from bla files.

---

[2] PSI-BLAST is described in section 2.4.2

[3] Remember that similarity is not a binary concept, but continuous. Consequently, saying most similar sequences means sequences that exceeds an arbitrary threshold.

3. For each neighborhood, build the multiple sequence alignment with ClustalW. The output is stored into a file of which the extension is aln. The file will be called as aln file. At the end of this step, each sequence has a personal aln file.

4. For each sequence, generate a profile-HMM using HMMER's `hmmbuild` component with sequences' aln files. The output is profile-HMM files with extension hmm. At the end of this step, for a data set of N sequence, N hmm files are generated.

5. Using concatenation ability of HMMER[4], a profile-HMM database is formed with individual profile-HMM's. This lets a database search for a batch sequence-profile comparison.

6. For each sequence in data set of cardinality N , do a sequence-profile search over profile-HMM database. Each comparison search consists of N comparison. The database search is executed for each sequence in the data set. This makes $N \times N$ sequence-profile comparison. This step produces, as a consequence, $N \times N$ comparison value.

7. Using comparison values, form a similarity matrix of dimension $N \times N$. Since HMMER's components `hmmsearch` and `hmmpfam` compute *e-value* and score each of them, at the end of procedure, 4 similarity matrices are expected to be obtained.

At the end of this procedure, a similarity matrix is obtained for a protein set of cardinality N . This similarity matrix can be used as a standard data set, in known classifiers such as bayes classifier or SVM, or classifier that will be mentioned in section 7.1 such as k-Nearest-Neighbor or threshold-Nearest-Neighbor.

---

[4] Concatenation is not directly done by HMMER, since it can be done using cat command of Unix. But, saying as ability of HMMER, it's intended to emphasize that HMMER supports such a concatenation, and user can form a profile-HMM database.

6.2.2.2 Data Set Generation Via Class Based Profile Sequence Comparison

This method is simpler than the neighborhood based method in the meaning of theory. Briefly, in the previous method, the profile-HMM's are generated such that each profile-HMM represents a sequence. This time, a profile-HMM represents a class. Hence, a sequence profile comparison generates similarity of a sequence with a class but not two sequences. This way, a vector can be obtained for a given sequence. The number of features that the vector contains is equal to number of class in the data set, since each feature is similarity of the sequence to the corresponding class. In the direction of this idea, a profile-HMM is generated for each class. Hence, if the data set contains M class, the number of profile generated in this method is also M . Then for a data set that the number of sequences it contains is N , the number of sequence profile comparison is $N \times M$.

Note that, the set of sequences which is going to be used for profile-HMM generation is already defined. Unlike the previous method, there is no need to execute a neighborhood detection step. Consequently, multiple sequence alignment procedure is executed as soon as class based clusters are generated. For multiple sequence alignment, like the previous procedure, due to same reasons, ClustalW is used. The steps of this method can be listed as:

1. Cluster all sequences according to their class labels. This forms M clusters where M is the number of class in the data set.

2. Execute multiple sequence alignment for each class. This results M number of aln files at the end of step.

3. Using HMMER's `hmmbuild` component, generate profile-HMM for each aln file, i.e. for each alignment.

4. Using HMMER's `hmmsearch` and `hmmpfam` components, for each sequence with each profile, do sequence profile comparison.

5. Extract e-values and scores obtained via execution of `hmmsearch` and *e-values* and scores obtained via this time execution of `hmmpfam`. Form data set matrix. AT the end of this step, 4 data set matrices are expected to be formed.

6.3 HHsearch – HMM-HMM Comparison for Homology Detection

HHsearch detects homology by means of profile-HMM's pairwise alignment. A profile-HMM is a template representing how well conserved residues are, in their own location. In case of proteins having similar structure and/or having similar function, observing some amino acids in some specific locations is more probable. While in some locations, deletion of some amino acids in some specific locations is more frequent, observing some amino acids is less likely. A profile-HMM detects those patterns using multiple sequence alignment.

In a profile-HMM, M and I states produce expected amino acids in specific locations and states D produces multiple alignment model's gaps, by means of modeling deletions. Consequently, the version of each sequence aligned in an multiple alignment block can be represented by a path in profile-HMM.

Pairwise Alignment of A Sequence and A Profile-HMM

Log-odds score: Log-odds score is computed to compare (or align) a sequence to a profile, or a sequence to a profile-HMM. A log-odds score similar to one given in Equation 6.1 is computed for every path which can produce a specific sequence. The term at the numerator of right hand side is the probability of observation that the AAS $\left(x_{1,}x_{2,}\ldots,x_{L}\right)$ is generated by a specific path on the current profile-HMM. The term in the denominator is the probability of the observation that the same sequence is generated by a random model or null model. Note that the AAS, here, does not include

any gap, i.e. this case is not a part of multiple sequence alignment procedure. The query sequence is simply an AAS and the aim is assigning this AAS to a specific path on the profile-HMM. Assigning an AAS to a path is equivalent to aligning a sequence to a HMM **(Söding, 2005)**.

$$S_{LO}=\log\frac{P\left(x_{1,}\ldots,x_{L}|A\right)}{P\left(x_{1,}\ldots,x_{L}|B\right)} \quad \text{A: coemmission on the path,} \tag{6.1}$$

Best alignment: The alignment which has the best log-odds score and which can be found via dynamic programming.

Pairwise alignment of profile-HMM's

Log-sum-of-odds score: In case of comparing (i.e. aligning) a pair of HMM, we need a generalized form of log-odds score. Defining log-sum-of-odds score is possible proposition. Similar to the case of aligning a sequence to a HMM, the alignment of a pair of HMM can be represented by a path, which is indeed a combination of two paths, each private to a HMM. Since two HMM's are aligned, each possible AAS that can be generated by both of HMM's must be taken into account. This point is held by the equation (6.2) which computes a specific alignment score between a pair of HMM. In this equation, the probability that a AAS is generated by both of 2 HMM's, coemmission probability, is divided by the probability that the AAS is emitted by a null model. The same procedure is applied for all possible AAS and the scores for each AAS is summed up and given to the log function. Consequently, the contribution of each AAS that can be coemmitted by the pair of HMM is taken into consideration **(Söding, 2005)**.

$$S_{LSO}=\log\sum\frac{P\left(x_{1,}\ldots,x_{L}|A\right)}{P\left(x_{1,}\ldots,x_{L}|B\right)} \quad \text{A: coemmission on the path,} \tag{6.2}$$

Best alignment: It's computed with dynamic programming which is in fact Viterbi decoding method applied to feasible alignment space.

Using secondary structure information: In order to align a pair secondary structure symbol, we need to derive a scoring function. HHsearch considers two cases: (1) one of the symbols was predicted and the other was obtained via experimentation; (2) both of the secondary structure symbol was predicted.

1. The case of predicted against known: The parameters are used as given below:

σ : The residue's known secondary structure symbol which is being aligned (or HMM column). This information is retrieved from DSSP database.

ρ : The other residue's predicted secondary structure symbol which is being aligned (or HMM column). The predictions can be obtained using state-of-the-art prediction tools.

c : The prediction confidence value. It's an integer that can have values from {0,1,...,9}. Its task is to indicate how confident the prediction is. The prediction get more confident as this parameter gets higher values. Briefly, the higher c is, the more confident prediction is.

$P(\sigma;\rho,c)$: This is a parameter of null model. A matching between known secondary structure expressed with σ and predicted secondary structure expressed with ρ, having c as confidence value.

$P(\sigma)$: The *apriori* probability of the known secondary structure symbol.

$P(\rho,c)$: The *apriori* probability of the secondary structure symbol ρ predicted with confidence value c .

$M_{SS}(\sigma;\rho,c)$: The substitution matrix for the alignment of a predicted secondary structure symbol expressed with σ and a known secondary structure expressed with σ having c as confidence value. It's computed as shown in equation (6.3). The matching probability is divided by the product of probabilities obtained by null model.

Then the log function is applied.

$$M_{SS}(\sigma;\rho,c)=\log\frac{P(\sigma;\rho,c)}{P(\sigma)P(\rho,c)} \quad \textbf{(6.3)}$$

$S_{SS}(q_i,p_j)$: The score of the alignment of i[th] column of the HMM q and j[th] column of the HMM p using only secondary structure information. Here, q_i has the predicted secondary structure ρ_i^q with confidence value c_i^q and p_j has the known secondary structure σ_j^p . The equation is obtained as given in equation (6.4) **(Söding, 2005)**.

$$S_{SS}(q_i,p_j)=w_{SS}M_{SS}(\sigma_i^q;\rho_j^p,c_i^q) \quad \textbf{(6.4)}$$

In order to compute the substitution matrix $M_{SS}(\sigma;\rho,c)$, the database SCOP 1.63 is filtered so that the sequence identity is 20% at most. True secondary structure labeling of sequences in this database is fetched from DSSP, and the predicted secondary structures are obtained using PSIPRED software. Then, for each amino acid, predicted and known secondary structures are compared and the result is used in probability estimation. The basic idea behind this approach is so: in the case where a sequence is aligned against itself, identical residues are matched to themselves (i.e. the matching states are real matching states.), probability of matching of known and predicted secondary structures can be definitely estimated. Probabilities of null model can be estimated as marginal probabilities using the term $P(\sigma;\rho,c)$.

The contribution of secondary structure score is obtained via adding the term $S_{SS}(q_i,p_j)$ to the score computed with AAS. As a result, the SS information is added to algorithm which finds the best alignment. Note that, a columns secondary structure state is considered as the secondary structure state of HMM's seed sequence **(Söding, 2005)**.

2. The case of predicted against predicted: The parameters are defined the

way as given below:

ρ_i^q : Secondary structure symbol observed in ith column of the HMM q.

c_i^q : The value of prediction confidence of ith column of the HMM q.

ρ_j^p : Secondary structure symbol observed in jth column of the HMM q.

c_i^p : The value of prediction confidence of ith column of the HMM p.

σ : The actual (known) SS symbol of the HMM column which is subject to alignment. In the formulation, Söding does not make an explicit distiction between actual secondary structure symbols of HMM's p and q.

$P\left(\rho_i^q, c_i^q; \rho_j^p, c_j^p\right)$: The probability of matching of a predicted secondary structure of type ρ_i^q with confidence value c_i^q and a predicted secondary structure of type ρ_j^p with confidence value c_j^p .

$P\left(\rho_i^q, c_i^q\right)$: The *apriori* probability of predicted secondary structure ρ_i^q with confidence value c_i^q . It's a parameter of null model.

$P\left(\rho_j^p, c_j^p\right)$: The *apriori* probability of predicted secondary structure ρ_j^p with confidence value c_j^p . It's a parameter of null model.

$M_{SS}\left(\rho_i^q, c_i^q; \rho_j^p, c_j^p\right)$: The substitution matrix (or score) of a predicted secondary structure of type ρ_i^q with confidence value c_i^q and a predicted secondary structure of type ρ_j^p with confidence value c_j^p . In score estimation, all possible cases of actual secondary structure σ are taken into account. While estimating this score, the assumption given below is admitted:

$$P\left(\rho_i^q, c_i^q; \rho_j^p, c_j^p | \sigma\right) = P\left(\rho_i^q, c_i^q | \sigma\right) P\left(\rho_j^p, c_j^p | \sigma\right) \quad \textbf{(6.5)}$$

$$S_{SS}\left(q_i, p_j\right) = w_{SS} M_{SS}\left(\rho_i^q, c_i^q; \rho_j^p, c_j^p\right) \quad \textbf{(6.6)}$$

The score obtained in right handside of the equation (6.6) is added to the score obtained via the alignment of AAS **(Söding, 2005)**.

6.3.2 Protein Function Prediction Using HHsearch

HHsearch produces a profile-HMM for each protein. Recall that, a profile-HMM is obtained with the alignment of an input protein with database proteins. HHsearch uses PSI-BLAST with up to 8 iterations in order to obtain multiple sequence alignments. In addition, it applies some filters for the purpose of guaranteeing that only homolog sequences are allowed into the alignment.

HHsearch aligns the profile-HMM of a query sequence to profile-HMM's of the data set proteins. Pairs over a threshold are assigned homolog whilst those under the threshold are labeled as non-homolog.

Some amino acid substitutions affect on alignment score while they keeps the function of the protein. Hence, the result that amino acids kept in a group of proteins that have the same function are necessary for that protein function and that varying amino acids do not cause a modification on protein function, can be deduced. As a consequence, the functions kept during the evolution are more likely to be detected by multiple alignment methods than by pairwise alignment. Hence, algorithms using profile-sequence similarity such as PSI-BLAST results better than those using sequence-sequence similarity such as FASTA or BLAST **(Söding, 2005)**. Then, a step further of these methods, profile-profile similarity is tried **(Sadreyev and Grishin, 2003)**, **(Yona and Lewitt, 2002)** and proven that profile-profile comparison methods outperforms sequence-sequence and profile-sequence comparison methods in problems

of secondary structure prediction **(Ohsen et al., 2004)** as well as SCOP class labelling **(Soeding, 2005)**, **(Cheng and Baldi, 2006).**

6.3.3 Alignment Score Estimation Using HMM

Although HHsearch is detailed recently, it would be useful to outline its principle ideas. HHsearch, given a query protein, firstly, detects its homologs in the search data set. More clearly, HHsearch estimates scores for homology of a given protein with other proteins in the data set. The most homologous protein from data set to the query protein is assigned the highest score whilst those relatively less homologous to the query protein are assigned lower scores. In this case, this output can be used indeed as a similarity measure. Actually homology is a clustering that labels proteins derived from the same ancestor during the evolution. Theoretically during the evolution, functionally important sites of protein sequences are kept unchanged whereas sites that have less structural importance or less functionality were corrupted via insertion, deletion or substitution of amino acids. Consequently, since roughly the homology means the similarity between specific sites of two protein sequences, proteins which are detected as homologous, i.e. which are inferred that they have sequentially similar sites, can be admitted to have similar functions. As a result, homology information can be helpful in function prediction. Using this hypothesis, the path that should be followed, is briefly, homologs of a protein which is subject to function prediction must be searched on data set which is formed by proteins that functions are experimentally known, and detecting proteins with highest homology score obtained on this search, then predicting the function of the query protein with respect to these detected ones.

Describing with further details, each protein is held as subject to homology search over the rest of the data set, and pairwise homology score is computed for each protein pair. This actually means that, for a data set containing N proteins, N^2 homology scores are obtained. Using these values, a matrix of size $N \times N$ can be formed. There, a row i is formed of homology score values of i^{th} protein with other proteins in the data set, even

with itself. Hence, the first diagonal of the matrix is formed of homology scores of each protein with itself. Finally, this similarity matrix can be used in a arbitrarily chosen classifier.

As the next step, the method of preparation of homology matrix using HHsearch can be explained. The data set, as mentioned in section 7.1, is a data set which contains FASTA formatted amino acid sequences of proteins annontated according to GO and with 40% sequence identity at most. It has 785 proteins. Additionally, the data set contains actual secondary structure information which is in FASTA format, obtained from DSSP database. Finally, predicted secondary structure informations is obtained via PSIPRED software. Consequently, alignment using amino acid only in HHsearch, also, alignment using both amino acid sequence and predicted secondary structure in HHsearch, and finally alignment using amino acid sequence and actual secondary structure in HHsearch were done and scores are computed. These experiments are respectively described in following sections.

6.3.3.1 Alignment Score Estimation with HMM Using Amino Acid Sequence Only

Steps of alignment score estimation using only amino acid sequence is given below. First, a local alignment with PSI-BLAST for each protein is done. For this, at first, a local alignment search database is formed using 'formatdb' command of PSI-BLAST. This database is built with contribution of each protein in the database. Recall that, the homology matrix is obtained using each protein in the data set. Until this matrix is obtained, there's no distinction of training and test set. The data set is processed as a unit set. Over this alignment database prepared for PSI-BLAST, an alignment search is done for each protein, and every alignment exceeding the arbitrary threshold are recorded in a file that the name is given referring to the protein for which the search is done. Using these files, HHsearch produces HMM's. HHsearch, again, forms a database using the HMM that it has recently built. Then for each protein, a homology search along the database is executed. The steps followed are listed below:

At the first 3 steps of the procedure, PSI-BLAST tool is used:

1. The whole protein data set that is stored in one file in FASTA format is fragmented so that a file per protein is produced. Each file contains the amino acid sequence of the protein to which it belongs. These files are also in FASTA format. Files are named with ID's of the proteins, of which they hold the amino acid sequence. E.g. 10A2:A.fa, .fa is the extension of the file which reminds that this is a FASTA formatted sequence file and 10A2:A is the ID of the protein of which the amino acid sequence is stored in the file.

2. Using PSI-BLAST's `formatdb` tool, local alignment search database at batch format is created from fasta formated sequence file which is also in batch format.

3. With `blastpgp` tool, each individual protein file created in step 1 is used for a local alignment batch search over blast database created in step 2.

HHsearch tool is used in next 5 steps.

4. For the purpose that HHsearch produces HMM's using the alignment files produced with PSI-BLAST software, PSI-BLAST output files must be transformed. This task is accomplished by `alignblast.pl` tool available in HHsearch distribution package. This procedure is applied to each alignment file individually, and each time an output file is obtained. E.g. for the alignment file 10A2:A.bla, `alignblast.pl` produces 10A2:A.a2m file.

5. Using the transformation files produced in the previous step, a HMM per protein is produced using `hhmake` command which is actually another tool included in HHsearch package. E.g. 10A2:A.hmm

6. In order to produce a profile-HMM, a HMM search database must be created using HMM's obtained in the previous step. This database is obtained via direct concatenation of the HMM files. For this purpose, Unix's `cat` command can be used.

7. The e-value which is used as a threshold value while producing HMM's, is

defined experimentally by Johannes Söding with respect to SCOP data set **(Söding, 2005)**. In the case of using a different data set, HMM's produced by HHsearch must be calibrated with respect to SCOP data set. For this purpose, Söding has prepared a database named as cal.hmm which contains only data which is necessary to calibrate e-values. Before producing its profile-HMM, each HMM must have been calibrated on this *ad hoc* database file. This procedure is done using `hhsearch` tool with parameter `-cal`, of HHsearch software. After the calibration, each hmm file is ready for producing profile-HMM.

8. In this step, the procedure of homology estimation with profile-HMM method is conducted. `hhsearch` tool of HHsearch software, both produces profile-HMM's and estimates the homology scores. Each calibrated HMM data which was obtained in the last step is queried over the HMM database and its profile-HMM is produced. In the homology score file produced for each protein separately, the pairwise homology scores of this protein with each of other proteins. In addition, other homology describing supplementary information such as e-value and probability value which describes in which degree this homology is valid, are written in this file. At the same time, the alignment regions of the owner protein of the file with other proteins are exhibited.
9. Being the last step of the procedure, it's not included in HHsearch algorithm. None of the HHsearch tools is used, here in. This step only contains the decomposition of the homology result file, fetching of the homology scores, e-value and probability values and storing them into three separate matrix which is going to be used then as similarity matrices. Briefly, it is consisted of an easy but complicated decomposition and recomposition procedure.

The procedure described above is depicted graphically below:

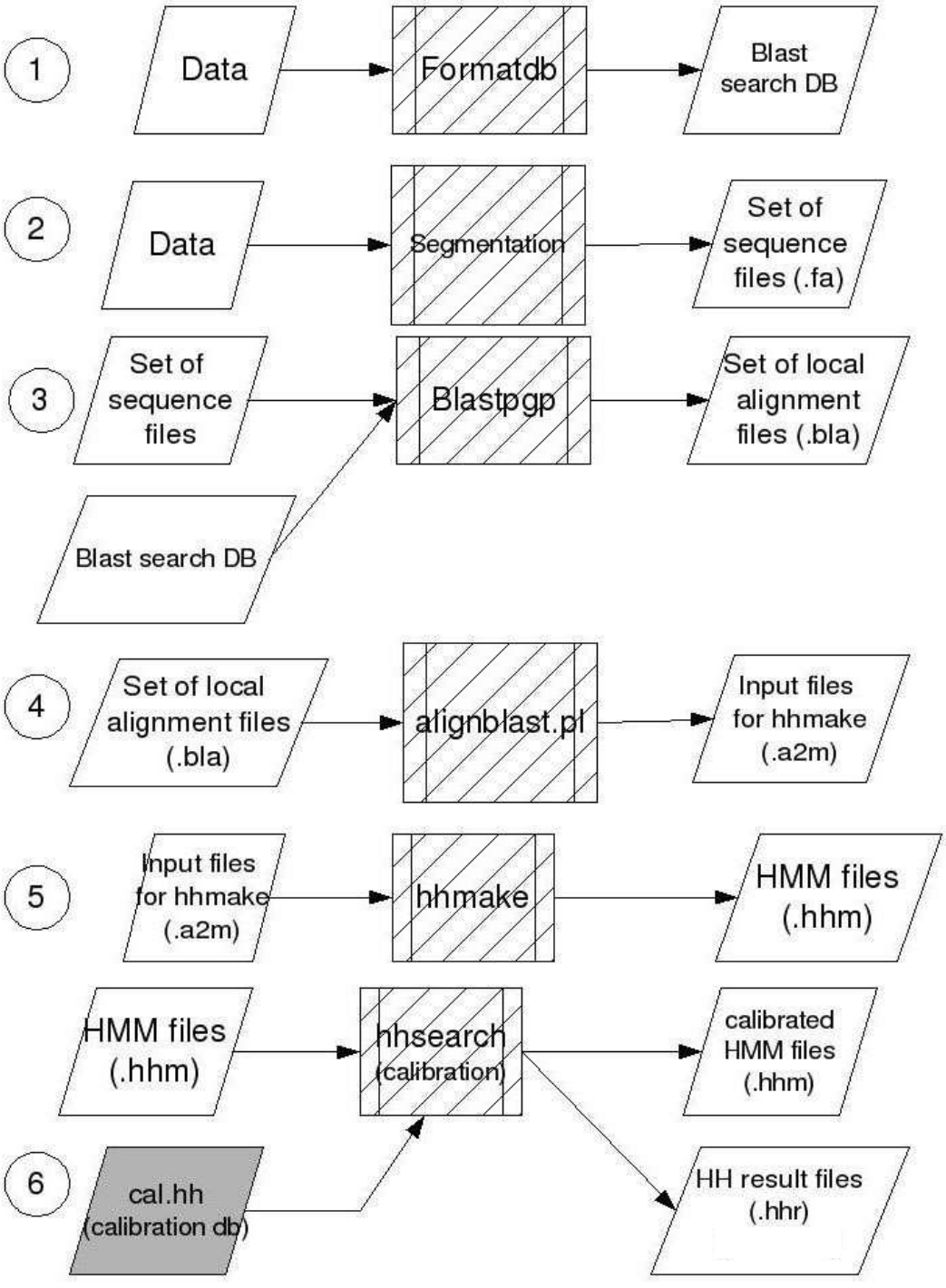

Figure 6.1: HHsearch Procedure Chart, first 6 steps

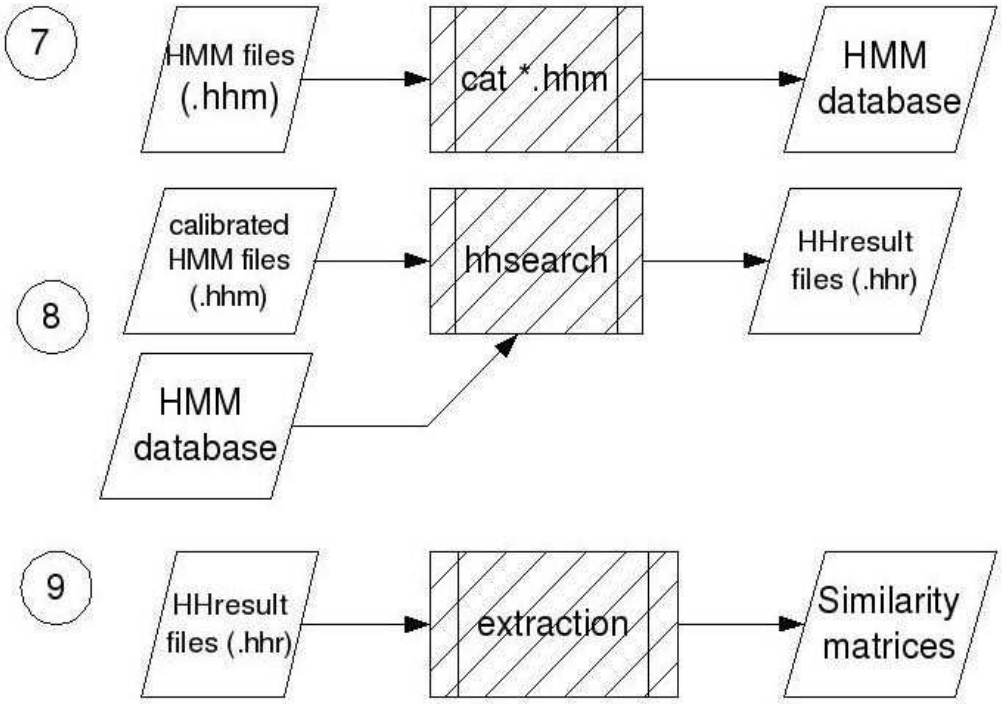

Figure 6.2: HHsearch Procedure Chart, last 3 steps

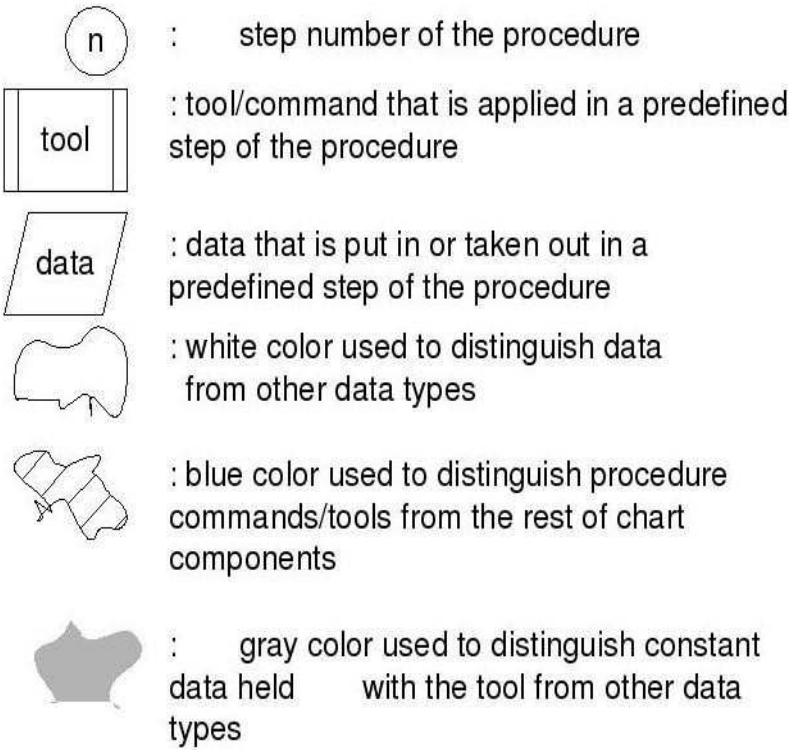

Figure 6.3: Legend of HHsearch Procedure Chart

6.3.3.2 Alignment Score Estimation with HMM Using Amino Acid Sequence and Secondary Structure

In this second way of estimation, as a difference from the previous section, secondary structure information of the proteins in the data set is used in homology detection. Since HHsearch software can do homology estimation using secondary structure information as mentioned in section 6.1, to apply this procedure, it's enough to set up the tools that HHsearch uses for secondary structure prediction. Actually, HHsearch obtains the predicted secondary structure using PSIPRED software which is mentioned in section 2, and can profit from secondary structure only this way. It adds the secondary structure information into amino acid information files using its own tools, and computes HMM's using this file again. Additionally, using actual secondary structure information in HHsearch procedure which is not a default usage, was included into the scope of this study. The procedure of HHsearch using secondary structure can then be observed under two subtitles: Predicted secondary structure and actual secondary structure.

6.3.3.2.1 Alignment Score Estimation with HMM Using Amino Acid Sequence and Predicted Secondary Structure

As described in the previous section, HHsearch can join secondary structure information into homology estimation. For the purpose, it uses one of the most famous softwares, PSIPRED. PSIPRED must be in the same file system with HHsearch in order to be used by HHSerach. In a brief description, some functions of PSIPRED can be outlined: PSIPRED is developed to predict secondary structures of proteins. It uses two methods. In the first method, it requires only an amino acid sequence in FASTA format, and makes prediction, taking into consideration only residue types in the sequence. This method is relatively unsuccessful, its error rate is much higher than the second method of PSIPRED. The second method is based on doing prediction using PSI-BLAST. In this procedure, as explained as a part of HHsearch procedure, a batch local alignment is executed over an amino acid database, and predicts secondary structure for the given

input amino acid sequence. This method has a higher accuracy and HHsearch works with this method **(Söding, 2005)**.

For the purpose that HHsearch uses this method, a tool that the name is `addpsipred.pl` was included in the package. This is a script written in perl. If the required settings could be done, HHsearch, accessing PSIPRED set up into the system, can run it and make it produce predicted secondary structure for an amino acid sequence. Recall that HHsearch could not use the output of PSI-BLAST and it had a post-processor script to extract necessary data; the same way HHsearch again can not use the output of PSIPRED. It has a file processing script to extract data from the output files of PSIPRED. Hence, the configuration of `addpsipred.pl` script is quite delicate in this situation. Just after the configuration is applied on processing script, HHsearch can be run directly. It will run and obtain the secondary structure results itself **(Söding, 2005)**.

HHsearch, does the secondary structure prediction between 4^{th} and 5^{th} steps of the operation sequence given in the previous section. The reason is that, HHsearch can give the a2m file which is in fact the input element in HMM estimation of HHsearch, can be given to PSIPRED as input data. HHsearch combines the information of amino acid sequence alignment with secondary structure information put out by PSIPRED and generates a3m file. The difference between this a3m file and the original a2m file is two character sequences. The first is the secondary structure residue sequence. The format of the secondary structure is HEC[5]. The second character sequence is formed of integers from the set {0,1,...,9}. Each element of this sequence indicates the confidence of the prediction at the corresponding residue **(Jones, 1999)**. The way that the confidence values are computed was explained in section 2.5.

The `hhmake` tool which is used in step 5 of HHsearch procedure, can process on a2m

[5] It is exactly the same format of HEL except that, instead of L, C is used. Remember that **L** stands for **L**oop and **C** stands for **C**oil.

which contains only amino acid alignment information as well as on a3m file which contains additionally secondary structure prediction information. In the second case it is supposed to estimate homology with a higher accuracy.

As it was described in **(Söding, 2005)**, the alignment score obtained from secondary structure information is added to the alignment score obtained from amino acid sequence information with an arbitrary weight as indicated in section 6.2. Since this weight is used while producing the profile-HMM which is the 8th step of the HHsearch procedure, it's given as a parameter to the command of `hhsearch`[6]. The default value of this weight is 0.15. In this study, different values are assigned for this weight. Let α be the contribution weight told recently. The similarity matrices in secondary structure contribution experiments are produced for $\alpha = 0, 0.20, 0.40, 0.70, 1$. Consequently, the similarity matrices are computed 5 times. In most of studies the contribution α is different from HHsearch's relation. In most studies the score expression is

$$S = (1-\alpha) S_{AA} + \alpha S_{SS} \qquad \textbf{(6.7)}$$

In HHsearch, the expression is

$$S = S_{AA} + \alpha S_{SS} \qquad \textbf{(6.8)}$$

6.3.3.2.2 Alignment Score Estimation with HMM Using Amino Acid Sequence and Actual Secondary Structure

In this method, actual secondary structure information is tried to be added into the HHsearch procedure and the results are tried to be observed with commenting the effects of actual secondary structure information on homology detection.

As shortly explained in the previous section, HHsearch can have higher accuracies by

---

[6] Actually, `hhsearch` is not a command, it's a component tool of HHsearch software. The expression command is used since `hhsearch` is used as a system command on the terminal.

means of adding secondary structure information into similarity measuring, in case of using good configuration. In order to assess the effect of secondary structure contribution, this information must be added into HHsearch procedure. Officially, the unique to add secondary structure information into HHsearch procedure is using PSIPRED which produces only predicted secondary structure as it is supposed to. Consequently, actual secondary structure could be added in artificial ways.

The method followed thus: using a script, a3m files which are obtained in HHsearch procedure with predicted secondary structure, are modified and actual secondary structure information is added. As mentioned in the previous subsection, a3m files, as the unique difference, has secondary structure sequence in HEC format and confidence value sequence in the first few lines. Once HHsearch obtains this file, it never checks the origin of these files. As a consequence, if actual secondary structure data is written instead of predicted secondary structure produced by PSIPRED, HHsearch reads this information and estimates the homology. The second issue that requires a decision is the way the confidence values for secondary structure are defined.

Recall the method of confidence values computation which is described in section 2.5 . Since, with a brief naming, the confidence values shows the uncertainty and since the actual secondary structure can be considered as certain[7], the confidence values can be selected highest, i.e. 9. Considering these last decisions, the a3m files produced in predicted secondary structure method can be manipulated. The modification is that, all a3m files secondary structure character sequences are substituted by actual secondary structure character sequences, and all confidence values are assigned 9. The rest of the procedure is exactly same as the previous procedure which is homology detection with HMM using predicted secondary structure.

---

[7] *Experimentally* doesn't mean that no error can occur, but means that possible errors due to physical measuring discalibration are out of this study's error scop and compensation. Thus, the possible errors in this case can be ommitted.

6.4 PRC – The Profile Comparer

6.4.1 Introduction to PRC

Theoretically, from primitive to advanced, comparison methods can be listed thus. Sequence-sequence, sequence-profile, profile-profile. Hence PRC, being a profile-profile comparison method, can be classified as advanced sequence comparison method.

PRC is a computational biology tool written by Madera, to compare profile-HMM's generated from biological sequences. The main idea behind PRC is that, comparing/alignment[8] two profile-HMM's and generate a score, measuring how much related/similar these two profile-HMM's.

Madera suggests that, two profile-HMM's similarity can be measured regarding a set of sequences. Remind that the relation between a sequence and a profile-HMM can be detected by means of computing the emission probability of the profile-HMM for the sequence. Given a set of sequences, each profile-HMM's emission for each sequence is computed. Then, the probability estimated by both of profile-HMM's are compared for each sequence, the closer these two profile- HMM's produces emission probabilities for each sequence, the more they are similar **(Madera, 2005)**.

$$P\left(HMM_1 \wedge HMM_2\right) = \sum_{i=1}^{N} P\left(S_i | HMM_1\right) P\left(S_i | HMM_2\right) \qquad \textbf{(6.9)}$$

The alternative method suggested by Madera is forming a new HMM using the two profile-HMM's subject to comparison. The new HMM will be called, pair HMM. It's not clearly a profile-HMM. States are formed of pairwise matching of states of original HMM's. Hence, if $state_i$ of the first profile-HMM is *matching* and $state_j$ from second profile-HMM is again *matching*, then a new state MM is added to pair HMM. For each

[8] Obviously, for a pair of HMM, the term *alignment* is not meaningful. But in bioinformatics, since the HMM *comparison* is a further step following sequence *alignment*, this term continued to be used, just by tradition.

coupling from the states of original HMM's, a new state is added to pair HMM. At this step, topologically, the pair HMM is fully connected. The transition probability of a state in pair HMM is computed as product of transition probabilities of states forming the new state. The emission probability of a state in pair HMM is computed as product of emission probabilities of original states **(Madera, 2005)**.

In order to make profile comparison, PRC requires prebuilt profile-HMM's since it does not generate profile-HMM's itself. It supports, two well known profile-HMM builder software: SAM **(Hughey et al., 2003)** and HMMER.

As a part of this study, SAM is intended to be used. But for whole thesis, HMMER was widely used and known better. Hence First experimentations are done using HMMER, although Madera stated in his thesis that, SAM outperforms HMMER in profile-HMM building **(Madera, 2005)**.

6.4.2 Function Prediction Using Procedures Using PRC

Unlike HMMER and like HHsearch, PRC does HMM-HMM comparison. This comparison can be used in similar ways to HHsearch, and also let to go further than HMMER ended. Remember that HMMER ended with sequence profile comparison.

In this study, PRC is used in two different methods. First is known and already described method. Very briefly, obtain profile-HMM for each sequence, do pairwise comparisons using profile-HMM's. Second is very similar to HMMER's second procedure. Cluster sequences with respect to class labels. Produce profile-HMM's restricted to clusters. Than produce similarities. Use it as a data set.

Details are given in following sections.

6.4.2.1 Similarity Matrix Generation Via Neighborhood Based Sequence Profile Comparison

This procedure is very similar to both HHsearch's method and HMMER's first procedure. The main idea is outlined in above. Recall that in HMMER's first method, the profile-HMM's are generated using a neighborhood approach. For a sequence i, the neighborhood is detected. This means actually that the set of sequences, of which the similarities to the subject sequence is over a threshold is detected. Again in this step, PSI-BLAST is used. From whole sequence data set, the sequences such that the pairwise local alignment with the subject sequence exceeds the default e-value, are listed in the blast output file of the subject sequences. This list forms the multiple sequence alignment set to that sequence. Once the neighborhood list is generated, then a multiple sequence alignment can be applied. Note that, the subject sequence is also included in the multiple sequence alignment process dedicated to that sequence. For the multiple sequence alignment, ClustalW is preferred for the reasons given section 6.2.2.1, HMMER's first procedure. Once the MSA is obtained, than a profile-HMM can be trained on this MSA output. Notice that, the process of profile-HMM training is not the task which can be accomplished by PRC, but by HMMER. HMMER can build profile-HMM's but cannot compare them. However, PRC can compare them but not build a profile-HMM. HMMER, then, is used to create profile-HMM for each sequence. Note that for a sequence i, the profile-HMM is generated using the multiple sequence alignment output for sequence i. When profile-HMM's for whole data set are obtained, as the final step, all profile-HMM's are compared one-to-one and using all comparison scores, a similarity matrix of dimension $N \times N$ is formed. This matrix can be used in classifiers then. Briefly, steps of this procedure can be listed as:

1. For each sequence, run PSI-BLAST over the data set and detect the most similar sequences to the query sequence.

2. For each sequence's similarity group, execute multiple sequence alignment and

obtain the output file aln, using ClustalW.

3. For each sequence, using the output of ClustalW, execute HMMER's `hmmbuild` component to generate profile-HMM.

4. For each pair of sequence in data set, do profile-profile comparison using PRC. Extract the scores and form the similarity matrix.

6.4.2.2 Similarity Matrix Generation Via Class Based Profile-Profile Comparison

This method is based on the hypothesis that for a better performance, a profile-HMM representing a sequence can contain features related to this class. In other words, recall that a profile-HMM represents the functional features of a sequence. In former procedures, while extracting profile-HMM's, class labels were omitted. As a consequence, in a profile-HMM, a feature either related to class label[9] of the owner or an irrelevant feature, can be represented if generally seen in neighborhood of the owner class. This can mislead the profile-profile comparison, an irrelevant feature can dominate the comparison. However, if the neighborhood of a sequence, of which the profile-HMM is built using this neighborhood, is reduced to same class sequences, then the profile-HMM is trained with respect to common functional sites seen in this neighborhood.

The procedure is as following: Using the neighborhood files generated in previous procedure (section 6.2.4.1) regenerate neighborhood files with respect to class labels. More clearly, for instance let sequence i has class label L. Scan the neighborhood list, eliminate all sequences which do not have class label L. In the neighborhood, the list contains only sequences of label L. Then execute the multiple sequence alignment over this list. Again the MSA tool used in this procedure is ClustalW. After the MSA output is obtained, HMMER's `hmmbuild` tool is applied and for the subject sequence i. And

[9] In the meaning that, if this feature is related to protein function or common for sequences having the same class label.

profile-HMM is built. Up to now, the unique difference from the previous procedure is neighborhood boundaries. The major difference starts here. The sequence i, recently mentioned, is not a random sequence from whole data set, but a sequence from training set. Hence all steps explained above are valid for training sequences. Hence the neighborhood of sequences are selected from training sequences. For the test sequences the same procedure is applied but the neighborhood is not defined over test set but again over train set. It means that, if the profile of a sequence j is to be built, the neighborhood of the sequence j is formed from training sequence. Then the multiple sequence alignment is executed over the neighborhood including the sequence j. The rest of the steps are same. Now, a profile-HMM for each sequence is created. Similar to previous method, a similarity matrix can be created.

Note that, according to GOA, a protein can have many GO labeling in molecular function. The data used in this study is soft-labeled, which means that, in the case of this study, a protein can have more than one class labels. This fact leads to the result as, a sequence can have different labeled sequences in its neighborhood. The all neighbors of a sequence are not supposed to have the same label. Even though, they all join to the same profile-HMM.

6.5 Improved Profile Methods

In the previous profile methods, the performance of different approaches are tested on our basic data which is 5-class GO data with 40% sequence identity[10]. The results[11] which are not as good as they were supposed to be, led us to better solution propositions giving us significant clues. We have followed two different road to improve the results: First, we charged on data and second, we tackled on classification method.

In the first improvement, we tried to ameliorate the data, as Assist. Prof. Dr. Hakan

[10] Check section 8.1 for a detailed description.

[11] All experimentation results are discussed in section 9, Results and Evaluation.

Erdoğan noticed that the basic data, containing 785 sequences, might not be enough to extract HMM's. So he proposed to use NR data which is deeply analyzed in section 8.1.3, to train HMM's. Recall that, to train HMM, we do not refer to class information, unless we train HMM's over label based clusters as in method of section 6.2.2.2 - Data Set Generation Via Class Based Profile Sequence Comparison, or as in method of section 6.2.4.2 - Similarity Matrix Generation Via Class Based Profile-Profile Comparison. Note that NR data's major disadvantage is that we do not have GO Annotation information for sequences within. As a consequence, we can use NR data only in steps which do not require label information such as profile-HMM training over a sequence set or neighborhood detection. For further information about NR data please refer to section 8.1.3.

As the second step of data amelioration, we can mention about, data enrichment which is a data extension procedure proposed by Assoc. Dr. Zehra Çataltepe. The major defect of NR data, as mentioned recently and stated by Zehra Çataltepe, is that sequence of NR data are not binded to GO Annotations, thus, we cannot use them as labeled data. This is indeed quite annoying. Hence, she proposed a different improvement approach. We have different level of data according to sequence identity. We work on 40% data, do not pass to higher data. Very briefly, the important condition in 5-class GO data of 40% sequence similarity is constraint of maximum sequence similarity between a test sequence and a train sequence and again maximum sequence similarity between a pair of test sequence which is fixed at 40%. Actually, this leads to the proposition that there's no need to hold the same constraint inside train set. Hence, keeping the constraint for the first two cases, we imported sequences from 5-class GO data with 95% sequence identity[12] (Please see Figure 8.3).

The second improvement consists of manipulation in classifier algorithm which is used once profile-HMM's are generated. In previous methods, k-NN algorithm, which is

[12] Please refer to section 8.1.4, 5-class Enriched Data. This section details the steps in generation of this data, describing it deeply.

described in section 7.1.1, is applied using $N \times N$ similarity matrix. Remind that, similarity matrix is formed of similarity vectors for each of the sequences data set. This means that a vector v_i of dimension N is generated by means of computation of similarity of sequence s_i to each sequence in data set. Using similarity vectors in k-NN algorithm, distance between each pair of sequence is computed. Actually, similarity and distance contains same information. Hence, using similarity to compute distance might blur the present information, even, it might cause loss of information. Hence, Assist. Prof. Dr. Hakan Erdoğan suggested to use similarity vectors directly in k-NN. He proposed **Best-Match** as a temporary distinctive name. This classification method is deeply explained in section 7.1.3.

To sum up, here are three modification approaches presented. In following subsections, four derived profile-HMM procedures using new data sets and new classifier algorithm, are described.

6.5.1 Sequence-Profile Comparison in Best-Match Approach using NR Data

This method is rather similar to the method exhibited in section 6.2.2.1 – Similarity Matrix Generation Via Neighborhood Based Sequence Profile Comparison. Recall that in that method, 5-class GO data is used. The blast search is applied on that data. Hence, the neighborhood detection and multiple sequence alignment using those neighborhoods are applied on that data too. As a consequence, that data is material of also profile-HMM's which are generated with HMMER. First modification on this approach is neighborhood search space. In contrast with the former procedure in the derived method, the search space is chosen as NR data. Each of the 785 sequences of 5-class GO data which are subject to the profile-HMM generation, is queried on NR cloud[13], close sequences from NR data to a query sequence forms the alignment neighborhood of that sequence. Just like the neighborhood detection step of the recently referred method, PSI-

[13] Cloud metaphor is used due to the fact that, in NR data, sequences are stored quite unorganized. No categorization, no pairwise relation, just sequences in FASTA format and ID's of millions of records.

BLAST search outputs are parsed to gather sequences of neighborhood in FASTA format so that multiple sequence alignment can be applied. As another difference from the previously referred method, not ClustalW but Kalign is used as the multiple sequence alignment tool. Since the neighborhood search space is selected as NR data, population in neighborhoods are much greater than neighborhoods of reference method. Hence, multiple sequence alignment procedure takes much longer. ClustalW keeps slow on this data. Thus, Dr. Zehra Çataltepe recommended Kalign which she had found after that unexpected problem. Kaling, claimed by Lassmann and Sonnhammer, is faster and more accurate than ClustalW **(Lassmann and Sonnhammer, 2005)**. Based on some quick trials, we experienced that Kalign can produce in same output format as ClustalW. So, substituting ClustalW with Kaling was feasible and did not cost much. After obtaining multiple sequence alignment outputs using Kalign, which are in fact an output per sequence, profile-HMM generation procedure is applied. In the end we obtained 785 new profile-HMM's. There's a trick which deserves to be mentioned. 785 sequences form in fact the whole data set. There's no 10-fold partition, so far. This means that, train and test sets are still mixed. Even though, there's no problem in profile-HMM extraction, since each sequence has no role in generation of another sequence's profile-HMM. Remember that, profile-HMM's are produced using only sequences from NR data and sequence of the profile-HMM itself. Once profile-HMM's are obtained, now is the time to execute 10-fold partitioning and to distinguish train and test sets. Recall that in Best Match approach, we need to compare similarities of each test sequence with each train sequence. There's no need to compute similarity between two train sequence. As a consequence, we compared each test sequence with each train profile-HMM. And we used the output in Best-Match algorithm. To compute similarity between a test sequence and a train profile-HMM, HMMER is used.

6.5.2 Sequence-Profile Comparison in Best-Match Approach using 5-class Enrich Data

In the method described recently, NR data is used as a source in order to generate profile-HMM's. As it has already been indicated, NR data does not contain any information but protein ID and amino acid sequence. This circumstance makes the solution candidate rather annoying since we can not do enough assumption. Dr. Zehra Çataltepe proposed enriched 5-class GO data which is generated as an alternative to NR data due to the problem stated above.

As a consequence, the unique manipulation on the previous method is substituting the data used to produce the model. Remind that the significant innovation applied by the previous method is using NR data as a profile-HMM search space. It doesn't mean that any data else is not used. Actually, 5-class GO data with 40% sequence identity is basically used, but NR data is profited as neighborhood source for each sequence from 5-class GO data.

In this method, enriched 5-class data replaced both of these data. It is used as a source of sequences which are subject to profile-HMM generation, and also used as a source of neighborhood which are detected for those subject sequences. Note that since the same sequence set is used for both of purposes, the train and test set separation is vital. In a case where sequences from both test and train set are mixed up in profile-HMM generation, the classifier model loses its validity since a classifier model can contain no information about test instances. Train and test separation was *de facto* obtained in data generation procedure (Please see section 8.1.4). Remember that 10-fold partition is made before sequence import process. At the end of the procedure, we obtained 50 enriched train sets and 50 corresponding test sets, one per each train set[14].

Given a train set of fold i and label j, profile-HMM's are produced for each sequence of

[14] 10-fold partition for each of 5 labels.

the train set, of which the neighborhood is detected on the same train set. First the member sequences of the neighborhood is detected using PSI-BLAST search for each train set. Then multiple sequence alignment is applied on the neighborhood sequences. Next profile-HMM is trained using this multiple sequence alignment output. Once the profile-HMM's are produced for train set, they form the HMM database for Best-Match inquiry database, over which test sequences are searched in FASTA format. Note that this method consists of sequence-profile comparison, hence there is no need to produce profile-HMM's for test sequences. For this comparison, HMMER is used.

6.5.3 Profile-Profile Comparison in Best-Match Approach using NR Data

This method is exactly same as “Sequence-Profile Comparison in Best-Match Approach using NR Data” method (section 6.5.1), except similarity computation in Best-Match algorithm. Note that in the first improved profile method, in Best-Match classification, sequence-profile comparison is used. Each test sequence is compared with each profile-HMM of train set. In this method, as a difference, profile-HMM generation is applied also for test sequences. In order to provide consistency in comparison, for profile-HMM generation of test sequences same conditions are held as profile-HMM generation of train sequences. As the source data in profile-HMM generation is NR data for train set, NR data again is used as the data source in profile-HMM generation of test sequences. In addition profile-HMM's of test sequences are generated using HMMER. Once profile-HMM's are obtained for test sequences, in Best-Match method, they are compared with profile-HMM's of train sequences. Remember that, PRC is the tool which is used for profile-profile comparison. There's another tool used in this study: HHsearch which is described in section 6.3. User can decide only data which is processed, parameter values in procedure steps and decides whether secondary structure will be involved in HHsearch. Remember that HHsearch does all former steps until profile-profile comparison on its own. On the other hand, PRC can use HMMER output, SAM output, even data in FASTA format. This makes PRC more flexible than

HHsearch.

More importantly, note that the goal of this process is comparing the method of sequence-profile comparison and the method of profile-profile comparison. For a better comparison, the procedures followed in both methods are preferred as similar as possible. Note that profile-profile comparison with PRC differs only at an additional step from sequence-profile comparison with HMMER method, if PRC runs with HMMER outputs. However, profile-profile comparison with HHsearch is totally different. As a consequence, in this method, PRC is preferred as profile-profile comparer.

6.5.4 Profile-Profile Comparison in Best-Match Approach using 5-class Enrich Data

It was stated that, the previous method was quite similar to method of sequence-profile comparison described in section 6.5.1 and also added that it differs only in similarity computation method of Best-Match algorithm. Same similarity is valid between this method and the method of `Sequence-Profile Comparison in Best-Match Approach using 5-class Enrich Data´. The data used in procedures, the tools used in procedure steps, the classifier algorithm which is Best-Match are all same, except similarity computation in Best-Match algorithm. In this procedure, like the one described in previous section, profile-profile comparison approach is applied instead of sequence-profile approach. Again PRC is used due to same reasons listed in the previous section. The tricky point in this modification is the source data for profile-HMM generation of test sequences. Note that, in NR data case, since no sequence from NR data is located in test set, profile-HMM's which are produced before train and test sets partition can be used after partitioning. For example, let sequence s_i is selected as train instance in fold j and test instance in fold k. Hence, profile-HMM hmm_i, can be used as train HMM for fold j and test HMM for fold k, without any modification, due to the fact all sequences joint in generation of profile-HMM hmm_i are in NR data set which is

independent from 5-class GO data.

This is not the case for this method since the data is already parititioned in folds and, test and train sets. Hence, profile-HMM of each sequence s at fold i, s is either in test set or train set, must be trained over train set of fold i. Hence the computation cost jumps immediately. For NR data method, neighborhood detection costs much since the search space is much greater. But there are 785 sequences which are subject to profile-HMM extraction. But in the case of enriched 5-class data, there are 50 train-test set pairs from 10 fold for 6 labels. And at each train set, there are approximately 1350 sequences and at each test set there are approximately 80 sequences. According to the equation 6.9:

$$(N_{seq\,in\,test\,set} + N_{seq\,in\,train\,set}) \times N_{fold\,per\,label} \times N_{label} = N_{profile-HMM} \quad \textbf{(6.10)}$$

This makes 71500 profile-HMM's to be computed.

The difference of computation cost in profile-profile comparison is much more significant. Each sequence in NR data method is compared 785 sequences which form the data set. However, in enriched data case, each test set must be compared with corresponding train set and this is supposed to be done for each fold at each label. According to the equation 6.10:

$$N_{seq\,in\,test\,set} \times N_{seq\,in\,train\,set} \times N_{fold\,per\,label} \times N_{label} = N_{profile-HMM} \quad \textbf{(6.11)}$$

This makes 5,400,000 comparisons to be computed.

6.6 Other Tools Working on Profile Hidden Markov Models

There are other different tools already taken places in the spotlight of this study. Those are SCOOP, SAM. SCOOP (Simple Comparison of Outputs Program) is a profile-profile comparison tool. SAM (Sequence Alignment Modeling Software) is a profile-HMM building tool.

SCOOP is written by Alex Batemen and Robert D. Finn. The basic idea of the method is that, pairwise comparison of profile-HMM's can miss some common sites indicating that the couple belongs to same superfamily. Another generalizing and less restricted method can catch weak relations according to **(Bateman and Finn, 2007)**. This method is searching both of the profiles along a superfamily database and compare matching set of database areas instead of query profiles themselves. If these two profiles have a common area greater than a threshold, than query profiles can be labeled in the same superfamily **(Bateman and Finn, 2007)**.

The second tool that deserves to be introduced is SAM. It's a software released by Karplus and his project team in University of California at Santa Cruz. Quite similar to HMMER, also SAM builds profile-HMM. Two basic differences SAM from HMMER are in front of eyes. First is that multiple sequence alignment is accomplished by SAM itself, whilst HMMER uses other tools such as ClustalW. And second is that, SAM does not calibrate e-values for profiles unlike HMMER, it computes via a theoretical function **(Madera, 2005)**,**(Hughey et al., 2003)**.

7. PATTERN RECOGNITION METHODS

As a part of this study, statistical methods are used. Since the aim is protein function prediction but not protein function computation, pattern recognition and classification techniques are used. The classification algorithms are preferred as simple and successful as possible. Hence KNN and tNN is used. For evaluation methods, ROC curve and AUC measure is used instead of accuracy which can mislead in some cases.

7.1 Classification Algorithms

7.1.1 KNN - k Nearest Neighbor

K nearest neighbor algorithm, as a classifier is quite simple and surprisingly has shown good accuracies. For these two basic reasons, it was chosen as the main algorithm of classification in this study.

Very briefly, KNN algorithm works this way: Each sample from test set is compared with train samples and the k train samples closest to the subject test sequence is selected. Here the distance metric is arbitrary. It depends on the characteristic of data set. Once the k nearest neighbor is detected, the class label having the most sample inside this neighborhood is assigned to test sample **(Alpaydın, 2004)**.

K is another parameter to be defined arbitrarily. In this study, K is chosen as 1, since it is quite effective and has low complexity **(Filiz et al., 2008)**.

7.1.2 tNN – Threshold Nearest Neighbor

This is a variant of K-Nearest Neighbor algorithm. Note that, a major problem in protein function prediction is that, if a sample has a class label i, it certain that the sample belongs to that class, however, even if it has not a class label i, it's not necessarily outside of that class. This means that, the sample can perfectly show this class i's features but these behaviors are not yet experimentally observed. Since in this study, only positive samples are certainly meaningful. The negative samples are uncertain.

The algorithm works this way: For a given test sample, if the greatest distance between the subject sample and training sample inside a class j is less than a threshold than, the test sample is assigned with this class label **(Filiz et al., 2008)**.

7.1.3 Best-Match Classification

This classification algorithm is another variant of K-Nearest Neighbor algorithm. In the previous classification methods which are mentioned recently, the algorithm requires an ordinary data set which is in fact in the format of a feature vector per sample, there the dimension of the vector is constant at n. At k-NN and t-NN, the distance/similarity between two samples is calculated using their feature vectors. The metric is arbitrarily chosen according to problems nature.

In methods which use profile-HMM's, the similarity between two sample proteins can be computed by means of features which are extracted from profiles to be used in distance metric. Briefly, up to now, in each method, profile-HMM's were used to compute a NxN similarity matrix where each row is a feature vector. Each member of a feature vector from similarity matrix is a similarity value between the sample to which the vector belongs and the sample to which the index of that member corresponds. For instance the j^{th} member of the i^{th} vector gives similarity measure between sample i and sample j. Once, the similarity matrix of NxN dimension is computed, then it can be used in Nearest-Neighbor methods where the distance/similarity between two sample is

calculated using feature vectors of those samples.

Actually, this adds an indirection since the feature vectors which are used exactly as a position vector in a space of dimension n which is the dimension of the feature vector also, since every component of the vector is in fact a distance value between two samples. Consequently, similarity which is computed using profile-HMM can be used as similarity/distance measure between two sample proteins. At this point, the sole drawback is that, indeed this measure cannot be a formal distance metric since it is not symmetrical. For instance, for a pair of sequence s_i and s_j , the similarity does not satisfy as $\mathrm{sim}(s_i, s_j) \neq \mathrm{sim}(s_j, s_i)$, while this equality is satisfied in indirect NxN similarity matrix method if the distance measure is a metric such as euclidean or manhattan.

As a consequence, in a Nearest-Neighbor based method, we need a routine to compute similarity between two given sequence samples. Hence, profile-HMM comparison methods[15] can be used as similarity computation routine. So, for each sequence in training set, profile-HMM is computed. Then, in order to compute similarity between a training sequence s_i and test sequence s_j , the profile-HMM of s_i (hmm_i) is compared with either the amino acid sequence s_j or with profile-HMM of s_j (hmm_j). In the first case, this makes sequence-profile comparison, in the second case, it makes profile-profile comparison.

With this subroutine, the rest of the algorithm is quite similar to k-NN. For a given test sample, by means of profile-comparison methods[10], similarity with each training sample is computed. Then n **best-match**ing training samples are selected and weighted voting is applied which is:

[15] Both sequence-profile comparison and profile-profile comparison methods

$$P(s_j|C_m)=\frac{\sum_{i,\, s_i \in C_m} \text{sim}(s_i, s_j)}{\sum_{k=0}^{n} \text{sim}(s_k, s_j)} \qquad \textbf{(7.1)}$$

This probability value is computed for each class in class set. Then, given a threshold, considering the probability value corresponding a class, the test sample can be decided whether it can be labeled with this class or not. If the probability value is above threshold, it can be labeled as positive $s_j \in C_m$, otherwise it can be labeled as negative.

The main problem leading us to develop this method is that in NxN similarity matrix method, the computation complexity is $O(n^2)$, plus k-NN complexity. However, in best-match method, for a data set of n element, the complexity is $O(k \times l)$ where $k < n$ and $l < n$. In most of our experiments, 10-fold cross validation is applied where the cardinality of train set is $\frac{9n}{10}$ and the cardinality of test set is $\frac{n}{١٠}$ where n is the cardinality of data set. As a consequence, the complexity of best-match is $O\left(\frac{9n^2}{10}\right)$ which is indeed $O(n^2)$ in theory. But it is a significant reduction in practice.

7.2 Evaluation Methods

Accuracy, recall and precision are conventional methods that are widely used **(Hamilton, 2007)**. These methods, although they are easy to understand and to compute, can be feinting in some cases. Hence, ROC curve and AUC value which are more robust against variety of cases get more popular in last studies. In early experimentation cases, ROC and AUC's are computed to evaluate models. In latter cases, accuracy of the models which are easier to understand which leads to have an idea

quickly is computed.

All evaluation methods mentioned above are deduced from *confusion matrix.* A confusion matrix is a report formed of test results of a model. It has 4 values: True Positive, True Negative, False Positive, False Negative.

True Positive (TP) is the number of positive test samples which are labeled as positive by the classification model.

True Negative (TN) is, by the same way, the number of negative test samples which are labeled as negative by the classification model.

False Positive (FP) is the number of negative test samples which are, contrarily this time, labeled as positive by the classification model.

Finally, False Negative (FN) is the number of positive test samples which are labeled as positive by the classification model **(Hamilton, 2007)**,**(Alpaydın, 2004)**.

Table 7.1: Confusion Matrix

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negiative	FP	TN

7.2.1 Accuracy

Accuracy is a measure which gives the ratio of correct labeling to all labeling, i.e. the number of correctly labeled test samples divided by the number of total labeled test

samples. This measure is not a reliable in some cases. For instance, in problems where the number of positive test samples is much greater than negative samples, even if the classifier model labels all test samples as positive, then the error rate is still small and the accuracy is high. Consider the case where the ratio of the number of positive test samples to the negative test samples is 95%. In this case, a classifier which assigns positive label for each test sample will have 95% accuracy, which is in fact a meaningless measure. However, accuracy can give basic idea about the success of the classifier **(Hamilton, 2007)**.

$$Accuracy = \frac{TP + FN}{TP + TN + FP + FN} \quad \textbf{(7.2)}$$

7.2.1 ROC – Receiver Operating Characteristic Curve

This a planar curve which depicts the behavior of *hit rate* with respect to *false alarm.* Here

$$Hit\ rate = \frac{TP}{TP + FN} \quad \textbf{(7.3)}$$

and

$$False\ alarm = \frac{FP}{TN + FP} \quad \textbf{(7.4)}$$

where TP is true positive, FN is false negative, FP is false positive and finally TN is true negative **(Alpaydın, 2004)**.

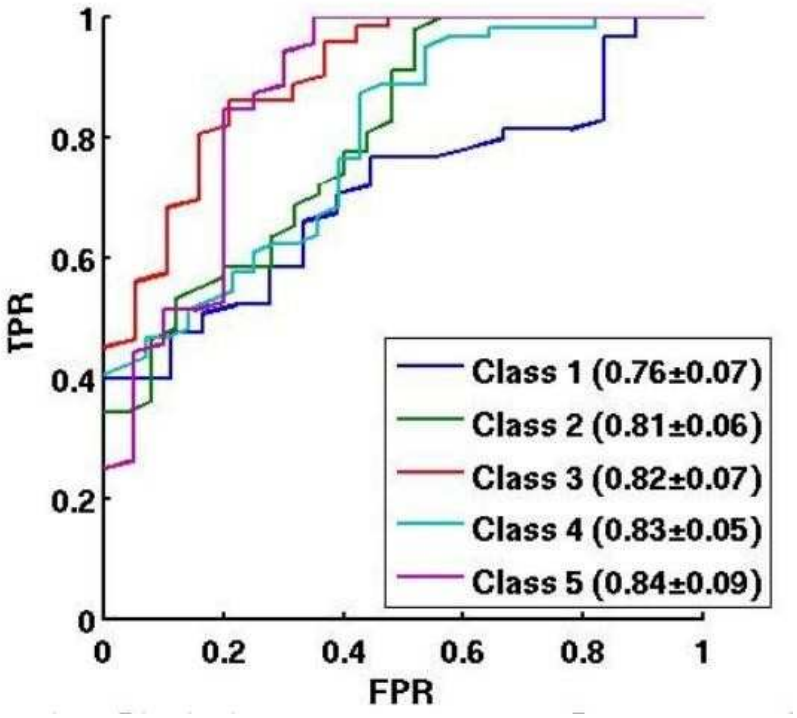

Figure 7.1: A sample ROC curve

In the best case, the ROC curve is a unit step function[16]. The random labeling results identity function[17]. Hence, a classifier is expected to have a concave curve greater than identity function and less than unit step function, as close to unit step function as possible.

7.2.2 AUC – Area Under Curve

In general cases, ROC curves let observer to decide which model outperforms. But in some rare cases, two ROC curves can not be better in whole definition interval. One can be better in a subinterval and the other can be in the rest part. Hence another measure is

[16] Horizontal line from the point (0,1) to the point (1,1)

[17] Straight line from the point (0,0) to the point (1,1)

needed.

Since in the best case, ROC curve is a step function and the area under its curve is 1 in [0,1], the area under curve can be a measure. The greater the AUC is the better the performance of the model is. As a consequence, if ROC curves do not give enough information to compare models, then AUC curve can be used.

8. EXPERIMENTS

8.1 Data

In this study, various data are used in experiments. They were prepared with respect to the test cases, classification methods which are defined in section 6 and also with respect to the problems occurred in experiments.

All types of data are described in following sections. Briefly, those are, 5-class GO Data, 27-class GO data, NR data, 5-class enriched data and 27-class enriched data.

8.1.1 5-class GO Data

For the purpose of training and testing of the classifiers, Gene Ontology Annotation (GOA) database is referred **(Camon et al., 2004)**. In this database which contains human, mouse, rat, arabidopsis, zebra fish, chicken and cow proteins fetched from UniProt Knowledge Base (UniProtKB) **(Wu et al., 2006)** and International Protein Index (IPI) **(Kersey et al., 2004)**, it is possible to access protein ID's in other databases, if they exist. Hence, for the purpose of querying amino acid sequence and secondary structure of the selected proteins in Protein Data Bank (PDB) **(Berman et al., 2000)**, only, proteins which are assigned ID from PDB, are used.

Just after, the proteins which are assigned in GOA **(Ashburner, 1998)(Ashburner, 2000)** and given ID in PDB, have been retrieved, in order to select the classes which will be used, low level GO classes' projections to the first level have been detected. Since the inclusion relation in GO hierarchy may be multiple-to-multiple, a term can have

multiple projections at first level. In addition, a protein which has been proven as multi-functional[18], is expected to be assigned multiple GO labels. In protein labeling, both cases were taken into account, each protein is matched every label at first level which includes at least one of the subject protein. Finally, five of all obtained GO labels, which contained neither too many nor too few matches, were selected as protein classes. Five classes are given in table (8.1)

Table 8.1: Selected molecular functions from GO annotation

GO Label	Number of Matching Proteins
GO:0005198 (structural molecular activity)	171
GO:0005215 (transporter activity)	214
GO:0030234 (enzyme regulator activity)	127
GO:0030528 (transcription regulator activity)	208
GO:0060089 (molecular transducer activity)	119

In bioinformatics, an attention requiring issue is avoiding repetitive information. Hence, in the preparation of this data, sequences that could increase the success of classifier better than it is actually. PDB's path has been followed to achieve this goal. PDB creates sets using BLASTClust tool which serves to collect proteins that have pairwise similarity. In data preparation, PDB's procedure was exactly applied and the data is saved from repetitions via keeping only one sample from the set of proteins that have similarity over 40%. Thus, in the data obtained, a randomly selected pair of sequence

[18] More than one function.

has 40% sequence identity at most. Via manipulating this constraint, different data sets are obtained. New data sets' sequence identity constraint varies as 30%, 40% 50%, 70%, 90%, 95%. Most of them are not used. But the data of 95%, for instance, is used to produce another data from %40. This procedure is described in following sections. Basically, in this pure state, only the data of 40% is used in experiments.

Note that, the number of protein in this data set is 785. Each sample included in the data set satisfies the constraint of being included in at least one class. There are samples which have multiple labels.

Table 8.2: Protein labeling in 5-class GO data set. Only first 10 rows are given. 0 indicates that the protein of that row does not belong to the class of that column. Note some proteins have multiple 1 which means that it belongs to multiple class at once.

Protein Index	PDBID	Label 1	Label 2	Label 3	Label 4	Label 5
1	1A02:F	0	0	0	1	0
2	1A02:N	0	0	0	1	0
3	1A04:A	0	0	0	1	1
4	1A0A:A	0	0	0	1	0
5	1A0S:P	0	1	0	0	0
6	1A22:B	0	0	0	0	1
7	1A2O:A	0	0	0	1	1
8	1A3A:A	0	1	0	0	0
9	1A54:A	0	1	0	0	0
10	1A6C:A	1	0	0	0	0

After the proteins have been selected for the data set, AAS and SS of these proteins are downloaded using web service provided by PDB. Finally, secondary structure prediction using PSIPRED is done, in order to evaluate the importance of using actual secondary structure. In the end, the data which contains 785 proteins' amino acid sequences, actual and predicted secondary structure, and which is also repetition-free, is ready to be used.

In experiments, to provide precise and accurate evaluations, the methods are tested with 10-fold cross validation. Recall that, k-fold cross validation is a partitioning procedure to define train and test sets, k times. At each partition, 1/k of the whole data set is defined as test set and the remaining samples form train set. Each time, the test set is formed by samples which have not already been selected as test set in former partitions. In addition, the distribution of class labels over test set is kept same as the distribution over whole data set. For instance, if the ratio of positive samples to negative samples is *a*/*b*, in each fold partitioning, for each *a* positive samples included in test set, *b* negative samples are also included into test set.

Remember that, 5-class GO data is soft labeled. It means that, a sample can have at least one class label, and probably more than one label (Please check Table 8.2, row 3 and 7). So we handled the classification problem as one-against-all classification problem. It actually means that we generated a classifier model of which the method is one of those defined in section 6, for each label at once and considered that label as positive class and combination of all remaining classes as negative class. Since there are 5 classes in this data set, one-against-all classification method results as producing 5 models given a classification method. In addition, considering the fact that a new k-fold cross validation is expected to be executed for each model, 10 fold-cross validation for 5 classes in that data results as 50 partitioning schemes[19]. Under these circumstances, for a fold partitioning, the distribution of instances in train and test set is kept with respect to subject class and other classes. This means that if the classifier model is constructed with respect to label *l,* the admitted distribution is such as: instances having that label *l* and instances which don't have this label. So the distribution of instances inside negative classes is ignored **(Aygün et al., 2008)**.

[19] Partitioning scheme means here reserving some instances for test set and considering the remaining instances as train set. Another scheme in this case would be selecting a test set totally disjoint from the former test set (or sets).

8.1.2 27-class GO data

Since review critics have indicated that a set of 5 classes is deficient to represent a function prediction problem, we decided to extend the class set to a higher degree. For the extension, two constraints respected in the preparation of 5-class GO data are omitted. First is that, only the highest level GO classes in the GO tree had been selected. Hence, all the classes of that data were at the same level. In this data, classes from various levels are selected. This way, we could have thousand of classes became

Table 8.3: Selected GO labels for 27-class data set with 40% sequence identity. The first column is the ID's of the classes in GO tree. The second column is the corresponding names and the third column is the number of sequences in the corresponding GO class.

GO ID	Name	# of Prot.
GO:0009405	pathogenesis	103
GO:0009055	electron carrier activity	105
GO:0006810	transport	107
GO:0016787	hydrolase activity	117
GO:0005506	iron ion binding	118
GO:0000166	nucleotide binding	132
GO:0003676	nucleic acid binding	137
GO:0003700	transcription factor activity	137
GO:0006508	proteolysis	148
GO:0006412	translation	150
GO:0003723	RNA binding	155
GO:0008270	zinc ion binding	170
GO:0005975	carbohydrate metabolic process	173
GO:0005179	hormone activity	177
GO:0016020	membrane	202
GO:0005515	protein binding	210
GO:0005634	nucleus	214
GO:0006355	regulation of transcription, DNA-dependent	221
GO:0005737	cytoplasm	232
GO:0005622	intracellular	278
GO:0005524	ATP binding	288
GO:0006118	electron transport	297
GO:0016491	oxidoreductase activity	300
GO:0003677	DNA binding	329

candidate. Within this class set, we selected 27 classes that have greatest number elements. This way we have increased the number of class. Table 8.3 and Figure 8.1 presents the selected GO classes and their population.

Recall that, in 5-class GO data, each protein is included in at least one class. To provide this feature to the data, we have eliminated all proteins which are included by no class. In 27-class GO data, we did not eliminate sequences which were not included in any class, considering that negative only samples are also a gain for the problem. In the end the number of classes is extended from 5 to 27 and the number of sequences in the data is extended from 785 to 4498 (**Filiz et al, 2008**).

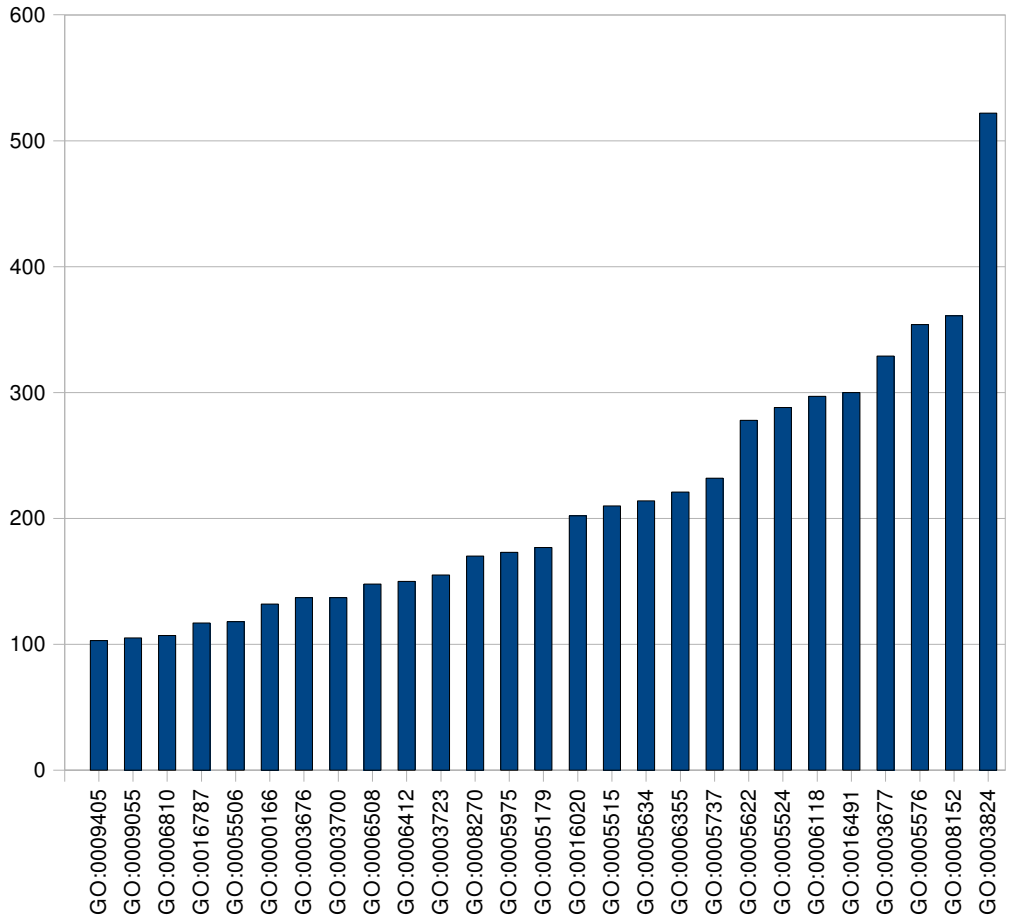

Figure 8.1: Bar graph depicting number of sequences per selected GO classes

The 27-class GO data described above has 40% sequence identity constraint, which means that, as explained in the previous section, a randomly selected pair of sequence has 40% sequence similarity at most. A variant of this data set, having 70% sequence identity constraint is obtained also. The unique difference between this data and the previous data is this constraint. Note that, when the sequence similarity threshold is elevated, the number of remaining proteins following the clustering is increased also.

Table 8.4: Selected GO labels for 27-class data set with 70% sequence identity. The first column is the ID's of the classes in GO tree. The second column is the corresponding names and the third column is the number of sequences in the corresponding GO class.

GO ID	Name	# of Prot.
GO:0006810	transport	142
GO:0016787	hydrolase activity	151
GO:0005509	calcium ion binding	154
GO:0003676	nucleic acid binding	162
GO:0020037	heme binding	167
GO:0000166	nucleotide binding	172
GO:0005488	binding	175
GO:0003723	RNA binding	186
GO:0003700	transcription factor activity	196
GO:0005506	iron ion binding	197
GO:0006412	translation	198
GO:0008270	zinc ion binding	243
GO:0016020	membrane	258
GO:0005975	carbohydrate metabolic process	260
GO:0005634	nucleus	289
GO:0006508	proteolysis	289
GO:0005576	extracellular region	296
GO:0005515	protein binding	298
GO:0006355	regulation of transcription, DNA-dependent	307
GO:0005737	cytoplasm	321
GO:0009055	electron carrier activity	333
GO:0005622	intracellular	349
GO:0005524	ATP binding	414
GO:0016491	oxidoreductase activity	420
GO:0003677	DNA binding	424
GO:0008152	metabolic process	495
GO:0003824	catalytic activity	784

Hence, the number of protein is increased from 4498 to 6202. Again, 27 GO classes are selected but this time, differently from the last 27-class GO data, since the cardinality of each class is changed due to the clustering. In the second case, the first 27 classes which have most element among all classes. Table 8.3 and Figure 8.1 presents selected classes' cardinalities in the preparation of the 27-class 40% similarity GO data, whilst Table 8.4 and Figure 8.2 gives class cardinalities of the selected classes in the preparation of 27-class 70% similarity GO data.

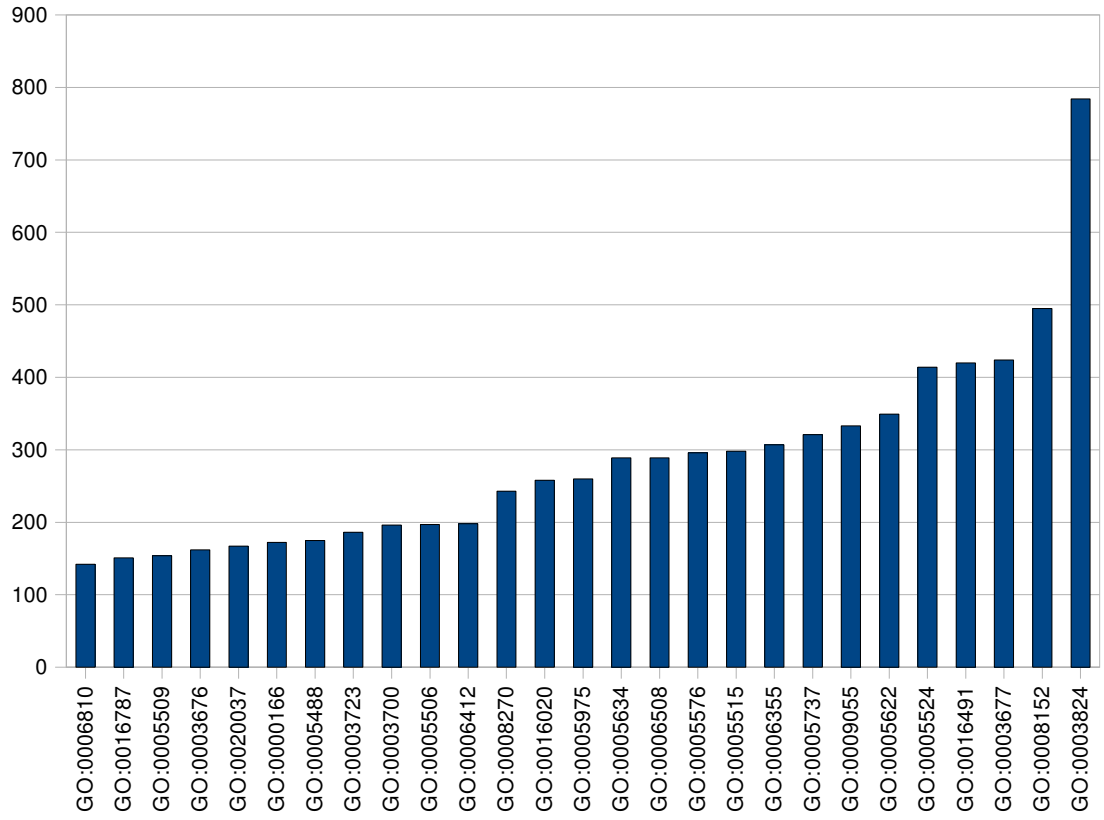

Figure 8.2: Bar graph depicting number of sequences per selected GO classes with 70% sequence identity

8.1.3 NR data set

Results of early experiments managed using profile methods described above on 5-class GO data were not satisfying. As it's mentioned in latter sections, accuracies in those tests are far below of our expectations. Assist. Prof. Dr. Hakan Erdoğan, noted that these failing results can be due to the relatively low population of sequence in data set. For some other methods, 785 sequences can be enough to extract knowledges for protein function prediction, but in profiling methods, this number might not afford. With Erdoğan's proposition, we decided to use NR data which is in fact a much greater data then 5-class GO data. It contains 6,494,103 sequence, takes about 3.5 GB on ext3 storage device, and around 10 min to scan only from most beginning to end. The data can be downloaded from NCBI's web site via NR file's address **(NCBIb)**.

In this study, this data is used for the unique purpose of increasing the neighborhood population joining in profile extraction. Note that there are 2 ways to increase this population. First way is to extend the neighborhood boundary, in order to include more samples with keeping sequence set. Recall that, the neighborhood boundary is defined by *e-value* threshold which can be assigned as a parameter in PSI-BLAST. *E-value* is the expectation value which is in fact, the number of alignments which has a higher than a predefined score, obtained by chance **(NCBIc)**. Therefore, lower is e-value defined, roughly, higher is similarity. As a consequence, extending the neighborhood includes sequences having higher *e-value*s which leads to extraction of profiles with sequences further from each other. This diminishes a profile's representing accuracy.

The second way is to import external sequences to the current data set. This way, we don't have to manipulate *e-value* threshold. This results as increasing the population in the neighborhood without modifying boundaries, hence any sequence within a neighborhood has a higher similarity than a threshold defined in earlier experiments. This way, we don't need to give up minimum similarity as a trade off for population

increase of a neighborhood. Unfortunately, there is still a trade off which is the lack of GO annotation for NR data. We don't have class information for any sequence from NR data set. We have only sequence information. This can be used to improve profile extraction. Note that a profile-HMM is built upon entities of protein evolution like insertion, deletion, matchings etc. Hence to extract profile-HMM's in good accuracy, a large number of sequence is required for multiple sequence alignment.

As a consequence, using NR data set, we have imported a large number of sequences which are similar to sequences subject to profile-HMM production. Those sequences from NR data set are chosen via local alignment search method, PSI-BLAST. Note that PSI-BLAST is an improved local alignment search technique which works much faster than known equivalent tools due to its heuristic algorithms **(Altschuland and Koonin, 1998)**. Once for a given sequence from one of our data set which are previously described, we can find similar sequences from NR data set. Following detection of similar sequences from NR, we can make first multiple sequence alignment and then profile-HMM extraction using multiple sequence alignments.

8.1.4 5-class Enriched Data

As indicated in the previous section which was about NR data set, the new data generation method has a major problem. There's no GO class information for sequences retrieved from NR data set which is not in fact a concrete data enrichment. Assoc. Prof. Dr. Zehra Çataltepe remarked that if we could find a way to increase the number of sequence in GO data sets via importing sequences having GO class information, that might be a gain of information in nearest neighbor or best-match algorithm. In this point of view, Zehra Çataltepe proposed to enrich 5-class GO data, by means of importing sequences from other larger GO data, since for each GO data we have, every sequence has GO class information. Recall that in section 8.1.1 (5-class GO data), data sets with various sequence identity constraint are described. We have 30%, 50%, 70%, 90% and 95% sequence identity data with same GO classes beside 40% data, i.e. sequences of

40% data are distributed over GO classes listed in table 8.1 and sequence's of other sequence identity level data set of 5 class data are distributed over classes listed in table 8.1 as well. As a consequence, when a new sequence is imported to 40% data, it is included in one or more classes present in data set, there's no need to create a new class.

Including new sequences violates the constraint of maximum sequence identity. Note that a sequence absent at a data set d_1 of p% sequence identity, but present in a data set d_2 of higher sequence identity is obviously eliminated due to the fact that it has a sequence identity higher than p% with at least one sequence from the d_1, and this sequence's sequence identity with all other sequences in data set d_2 is lower than data set's threshold. As a consequence, when sequences from a data set d_2 with higher sequence identities are imported to a data set d_1, this adds sequence identities higher than the constraint of the set d_1.

The question here is, which sequence identity that exceeds, are allowed and which are not? The reason of keeping a relatively low sequence identity while generating the data set for the classification problem is providing the problem meaningful and worth to work on. What makes the problem challenging and difficult is the low sequence identity between a test sample and training sample. Indeed, the level of sequence similarity between training samples does not have a key role on problem's hardness. With data sets having higher sequence identities, the problem of function prediction is not considered meaningful and challenging thus worth to deal with. Thus, while importing sequences from data sets of higher sequence identity, keeping the problem challenging and difficult at the same level is essential. Hence, a new condition is defined: The sequence identity between any pair of sample, one from test set and the other from train set must be kept while enrichment. Depending to this condition, a new sequence can be added either to train set or to test set. But at each case, foreign sequence's sequence identity with any sequence in other set must meet the new condition. For instance, if a new sequence *s* is imported to test set, no sequence from train set is allowed to have sequence identity with

s higher than the current condition. And if a new sequence *r* is added to train set, no sequence from test set can have sequence identity higher than the current condition, as well (Please see Figure 8.3).

The aim here is to improve the classification model. Recall that, train set is used to build the model and test set is used to evaluate it. So it's better, the new sequences are imported to train set. There's no need to enrich the test set. In addition, putting new sequences with high similarity with current ones, can produce biased tests due to test case repetition. Keeping attention on these two reasons, the good choice is to add new sequence to train set.

We have selected 5-class GO data with %95 sequence identity which is the most crowded among all 5-class GO data sets we have. There is no reason to use other data sets since all other sets have less sequences.

Given a train/test set couple, the procedure of enrichment can be outlined such as: For

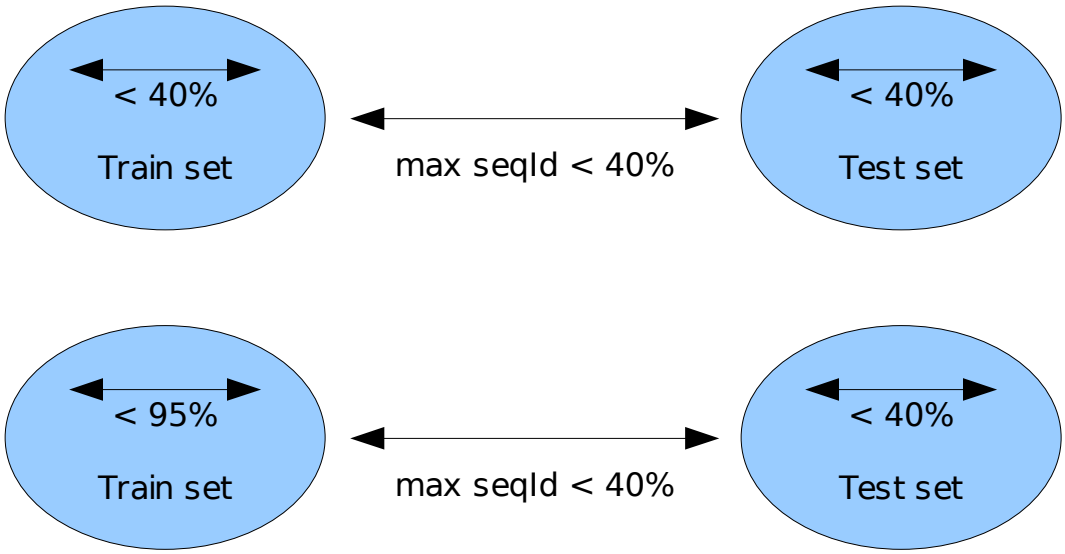

Figure 8.3: Venn diagram showing sequence identity variation with respect to sets, for 5-class GO data. The upper case is before enrichment, the lower case is after enrichment.

each sequence *s* in enrichment source set which is actually 5-class GO data set with 95% sequence identity, compute sequence identity between *s* and each sequence in test set. Eliminate *s* if a computed sequence identity exceeds the threshold, otherwise put *s* into train set.

Remember that the test and train set partition has been done as a part of 10-fold cross validation. In addition, 10-fold cross validation is applied 5 times, each one for a label in 5-class data set. This makes 50 train/test sets couples. As a consequence, an enrichment process requires 50 applications of the procedure described in the previous paragraph. Hence, at the end of the enrichment process of 5-class GO data, we obtained 50 enriched train sets for 50 test sets. Each enriched train set contains about 1350 sequences whereas an original train set contains 707 sequences.

8.1.5 27-class Enriched Data

For 27-class GO data set, the major problem is same as told for 5-class GO data set. There is not enough sequence to train profile-HMM's. Hence, an enrichment procedure similar to one applied to 5-class GO data set can be applied. However this time a different approach is held. Remember that in 5-class enrichment case, 5-class GO data with 40% sequence identity is enriched with imported sequences from 5-class GO data with 95% sequence identity. This time, to obtain an enriched 40% data, not an importing process is applied but a reduction process applied to source data set. 27-class GO data set with 70% sequence similarity is selected as source set. In this original state, this data is not convenient for a function prediction problem since the sequence similarity between data set proteins is high. Eliminating some proteins from this data set can provide wanted hardness for a function prediction problem. Recall that what makes a function prediction problem is maximum sequence identity between test and train set is 40%. By this reason, eliminating instances from 27-class GO data set with 70% sequence identity such a way that maximum sequence identity only between train and test sets and inside test set holds at 40% can provide a data set which makes function

prediction problem at aimed hardness level. According to this condition, maximum sequence identity inside train set is kept at 70% (Please see Figure 8.4). This keeps the population inside train set higher than a train set of original 40% 27-class GO data set. Likewise, each train set is prepared with respect to its corresponding test set.

The algorithm for the reduction is proposed by Dr. Zehra Çataltepe and Özgür Macit:

- For each train/test set couple
    - Do until no instance remains in test set:
        - Select an instance from test set randomly using function *f*.
        - From same test set and its corresponding train set, dismiss all instances which have greater sequence identity than 40% with currently selected test instance.
        - Put the currently selected test sample to reserved test samples set.

f function has a key role here. Randomization is important since we don't want any bias in dismiss. But a uniform random selection can keep instances of a class and dismiss mostly instances from the other classes with a small probability. Hence an *f* function is defined to make a fair elimination. Hence the probability of selection of an instance x_i from a class C_i is defined as:

$$P(x_i|C_i)=\frac{\exp(-n_i)}{\sum_{j=1}^{m}\exp(-n_j)} \qquad \textbf{(8.1)}$$

Here, m is the number of classes in data set plus one. Recall that there are instances which have no labels. Those instances are considered as forming a separate class. In our case, the data set has 27 class. So m is assigned 28. n_i is the number of already selected instances from class C_i . Hence, the more instances from class C_i are

selected, the more probability of next selected instance being from C_i decreases. The probability of selecting instances from other classes is relatively increased. *f* function, in one hand provides randomness, and in the other hand, keeps the equivalence.

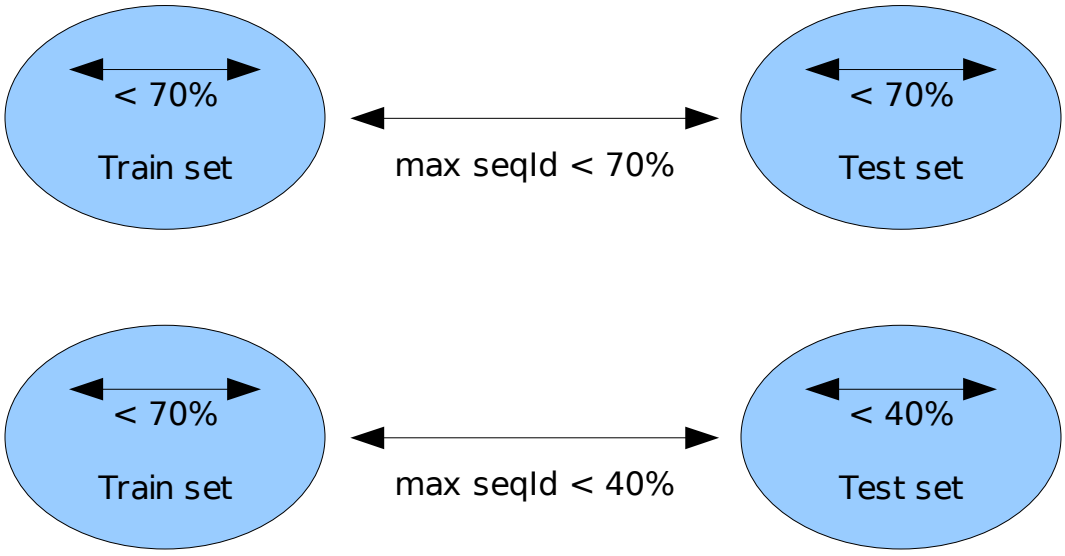

Figure 8.4: Venn diagram showing sequence identity variation with respect to sets for 27-class GO data. The upper case is before enrichment, the lower case is after enrichment.

8.2 Environments

Experiments are executed in various digital computer environments. The computers and used can be listed as:

1. Intel Pentium 4 3.20 GHz Hyperthreading with 1GB RAM, Debian 4.0 Etch. Python 2.5 Interpreter is used for experiments.

2. Intel Pentium 4 3.20 GHz Hyperthreading with 1GB RAM, WindowsXP SP2. Matlab 7.0.1 Academic Edition is used for experiments.

3. 2 Intel Xeon, 3.20 GHz Dual Core processors with 2 GB RAM, Ubuntu 8.04. Python 2.5 Interpreter is used for experiments.

4. 8 Intel Xeon 3.20 GHz Dual Core processors with 2 GB RAM, Fedora 7. Python 2.4 Interpreter, Matlab 7.0.1 and Matlab 7.6 (R20008a) are used for experiments.

9. RESULTS AND EVALUATION

9.1. Sequence-Profile Comparison with HMMER

In section 6.2, two different sequence-profile comparison methods are described: $N \times N$ similarity matrix-based classification (section 6.2.2.1) and Class-based generated data set classification (section 6.2.2.2). Since the results of both methods are quite similar only first methods results are given.

Note that we obtained 2 different $N \times N$ similarity matrices which are used in k-NN classifier, separately. These matrices are obtained using two different similarity scores which are produced by HMMER: *search score* and *search e-value* **(Eddy,1998)**. The results are given in two figures and in one table, below.

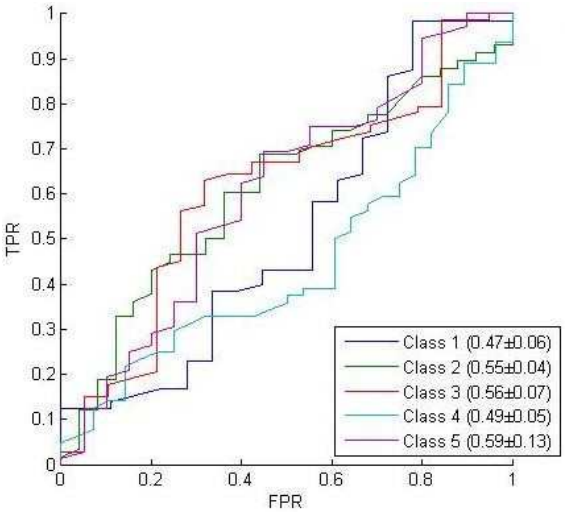

Figure 9.1: ROC and AUC obtained by search score of HMMER output.

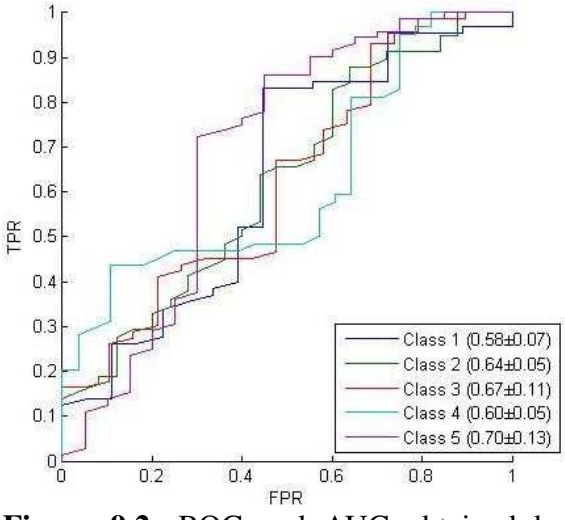

Figure 9.2: ROC and AUC obtained by e-value score of HMMER output

Table 9.1: AUC values obtained by search score and e-value score

	search scores	evalue scores
AUC	0.4715	0.5832
	0.5539	0.6398
	0.5623	0.6696
	0.4852	0.6023
	0.5945	0.6970
std dev	0.0612	0.0683
	0.0412	0.0534
	0.0733	0.1124
	0.0532	0.0540
	0.1300	0.1324

Figure 9.1 presents the ROC and AUC results obtained via generating similarity matrix using *search score* output of HMMER sequence-profile comparison. As it's clearly seen that, ROC curves are fairly close to $f(x)=x$ line, which is theoretically the random classifier's case. When the AUC results are checked, this interpretation is almost verified since all results are close to 50% which is AUC of $f(x)=x$ line. At label 1 and label 4, mean AUC's are below 50% which means that the classifier is even worse than random classifier. Figure 9.2 shows ROC curves and mean AUC results for classification which uses similarity matrix generated with HMMER's *e-values score* output. These results are slightly better. The least value is above 55% and the top reaches 70%. But anyway, the results are disappointing. The mean AUC values of 10-folds at each label are given with their standard deviation over again 10-fold at each label at Table 9.1.

9.2 Profile-Profile Comparison with HHsearch

According to procedure described in section 6.3, we obtained 11 $N \times N$ similarity matrices for 5-class GO data. 1 matrix for amino acid sequence only, 5 matrices for predicted secondary structure with contribution weight α at values

$\{0, 0.25, 0.50, 0.75, 1\}$ and 5 matrices for actual secondary structure with α at same five values. Remember that, each similarity matrix is used in k-nearest-neighbor classifier. For each of the 5 classes, one-against-all classification is applied. The results are evaluated with both ROC and AUC measures. The results are as follow:

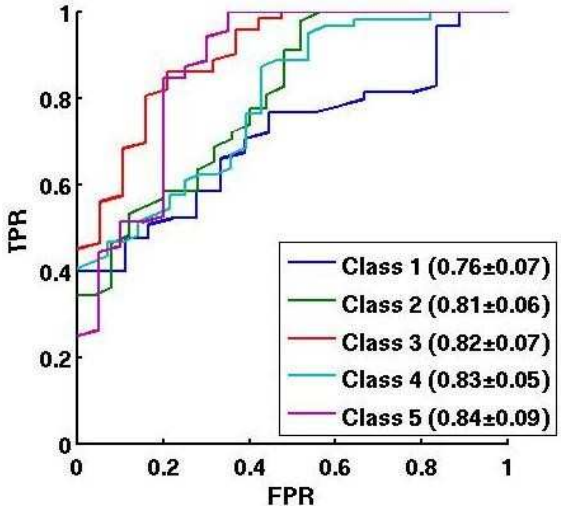

Figure 9.3: HHsearch results with no SS contribution

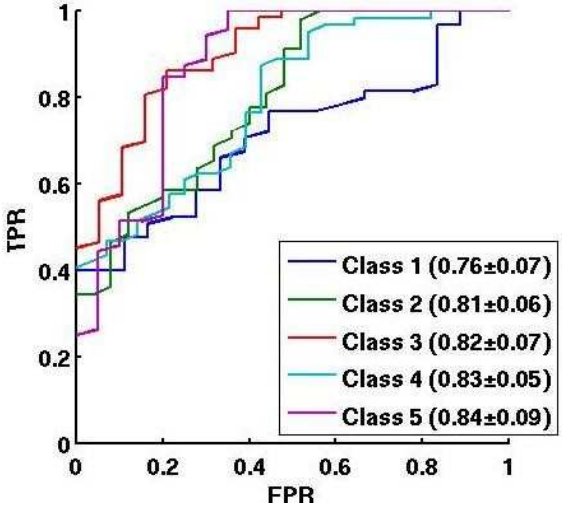

Figure 9.4: HHsearch results with predicted SS contribution weight $\alpha = 0$

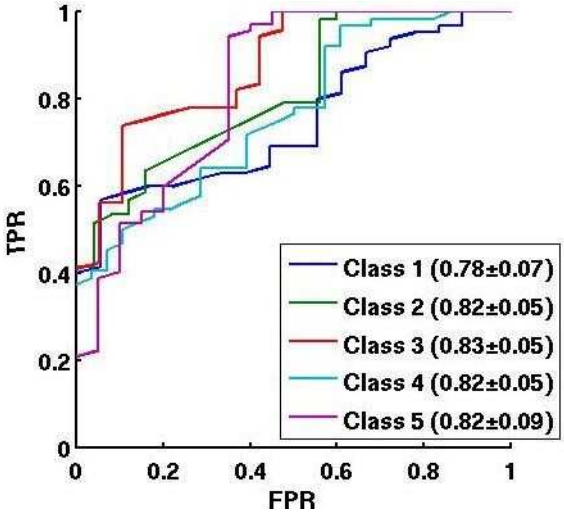

Figure 9.5: HHsearch results with predicted SS contribution weight $\alpha = 0.25$

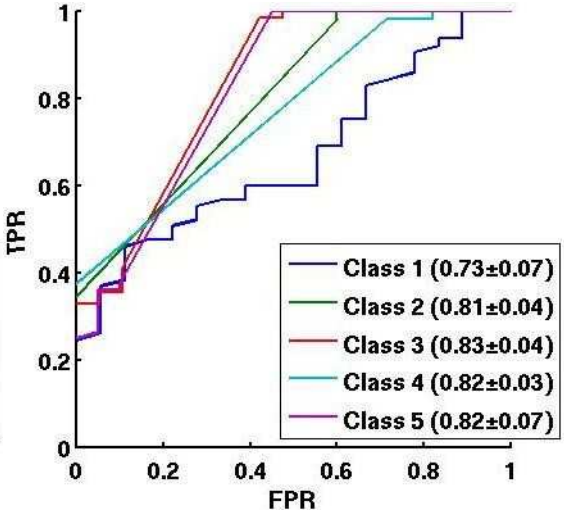

Figure 9.6: HHsearch results with predicted SS contribution weight $\alpha = 0.50$

Figures from 9.3 to 9.13 exhibit the ROC curve instances at 1 fold and average AUC values per labels for HHsearch method using amino acid sequence and, predicted or actual secondary structure with various α values except 9.1 which shows the case of amino acid sequence only. Note that α is secondary structure contribution weight. For instance, in following evaluation expressions, α = 0.50 means exactly same as SS contribution at 50%.

According to Figure 9.3, mean AUC values vary from 81% to 84 except label 1 which is 76%. For predicted Secondary Structure results, as expected, Figure 9.4 which is the case of SS contribution at 0% has the same curves and values with 9.3. It can be claimed that, the case of *no SS contribution* and *SS contribution at 0%* are same cases, which is valid theoretically. In practice, HHsearch has run in two different ways. Executing these cases separately, indeed tests HHsearch's consistency. Figure 9.5 is the case of α = 0.25 where labels from 2 to 5 are almost same as previous cases, but label 1 has a slight lift. Figure 9.6 which is the case of α = 0.50 shows similar results for labels 2-5 but for label 1, it has a sudden decrease to 73%. Figure 9.7 which is the case of α = 0.75, the results are similar to previous case. The final case of predicted SS tests which is presented in Figure 9.8 has best results ever in these HHsearch tests. Label 1 jumps to

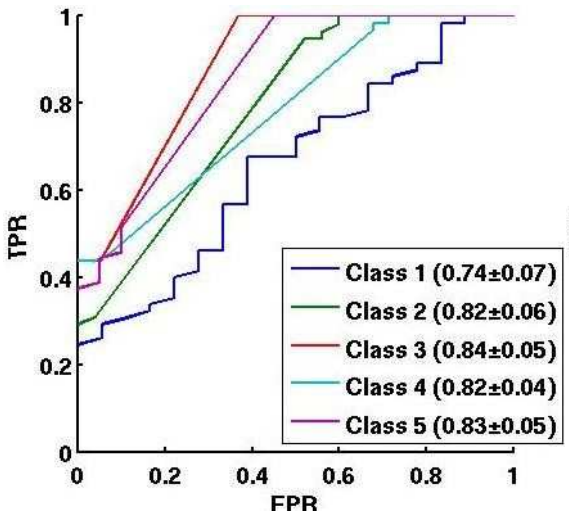

Figure 9.7: HHsearch results with predicted SS contribution weight α = 0.75

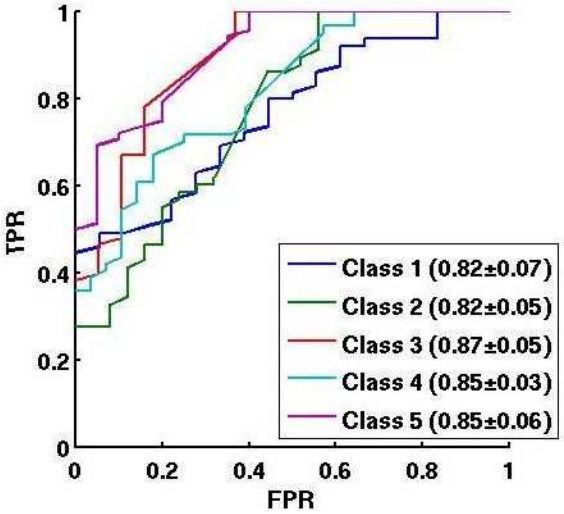

Figure 9.8: HHsearch results with predicted SS contribution weight α = 1.00

82%, it was under the limit of 80%, so far. Label 2 stays at the same level but the rest of labels increases 2-3%.

Results of actual secondary structure cases are given below. Figure 9.9 shows the ROC curves and mean AUC results for each label for SS contribution at level 0%. It's quite similar to predicted SS case at the same level. Label 1 is the worst. The other labels are in range of 82%-84%. There's a problem here. The results are not expected to be similar to predicted SS with 0% contribution but expected to be exactly the same, theoretically there is no SS contribution when $\alpha = 0$. This is actually a clue which cause doubt about HHsearch's consistency. Figure 9.10 presents the case of SS actual SS contribution at level of 25%. The values are very similar to the previous case except label 1. There's a significant increase in this label. It reaches at 80% at AUC.

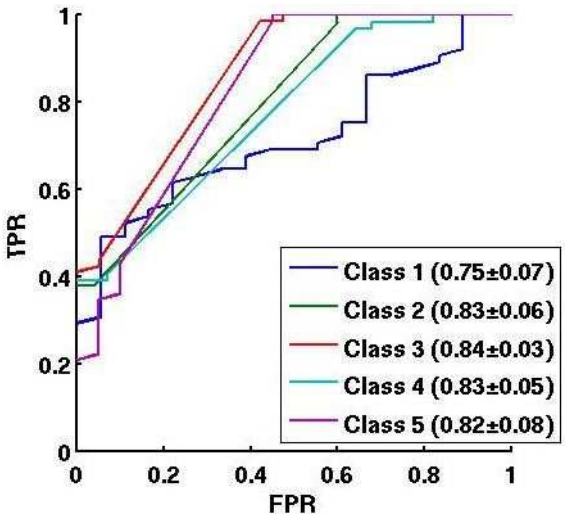

Figure 9.9: HHsearch results with actual SS contribution weight $\alpha = 0$

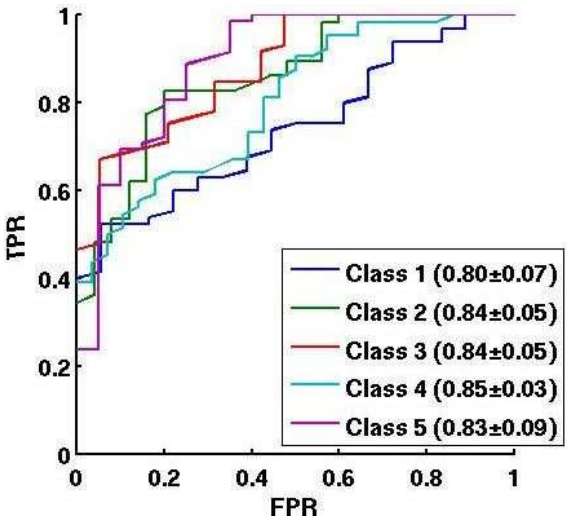

Figure 9.10: HHsearch results with actual SS contribution weight $\alpha = 0.25$

Figure 9.11 exhibits the results of the case of SS contribution at 50%. There's a surprising fall in some labels such as label 1. Label 1 has been under the boundary of 80% in some cases but never reached 70% before. Likewise, ROC curve of label 2 has never been under 80%, but in this test it has fallen below 80%. What is more surprising is that, those happen with variation of only 0.25 at α. Again label 4 shows a decrease of

5%. Figure 9.12 shows the results of the case of SS contribution at 75%. Label 1 lifts to 76% with 6% with respect to the previous case. This time, label 3 performs a significant increase with 4% with respect to previous case and reaches the top among the labels of the same case. Finally, Figure 9.13 displays the case of 100%. The results are exactly the same as the previous case.

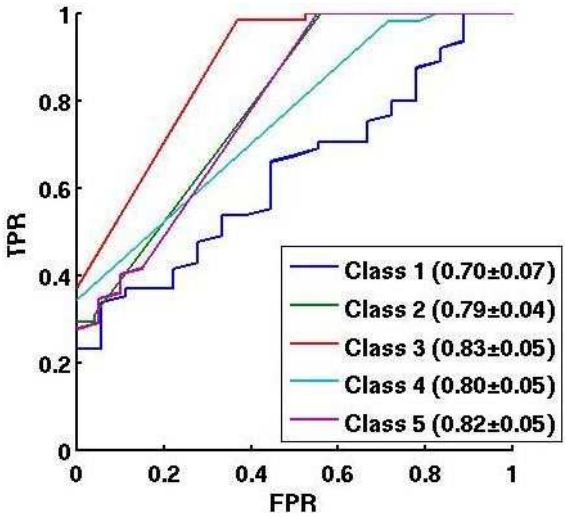

Figure 9.11: HHsearch results with actual SS contribution weight $\alpha = 0.50$

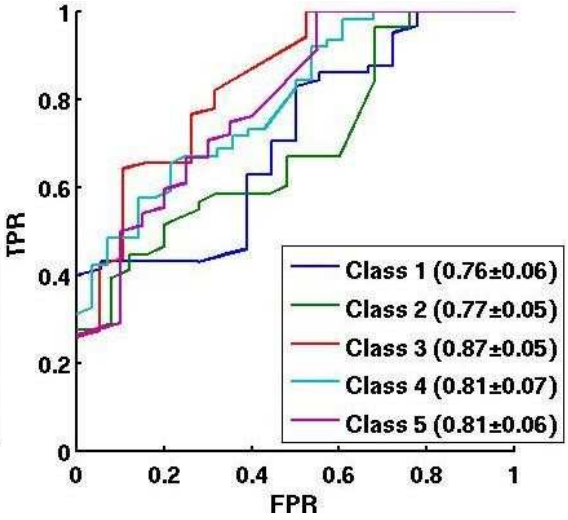

Figure 9.12: HHsearch results with actual SS contribution weight $\alpha = 0.75$

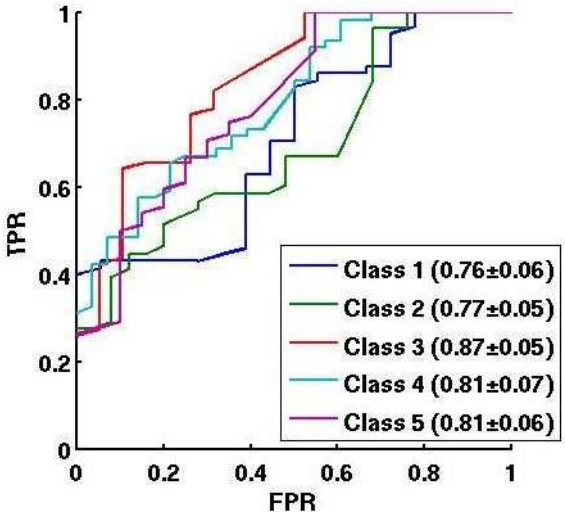

Figure 9.13: HHsearch results with actual SS contribution weight $\alpha = 1.0$

Table 9.2 shows mean AUC values all together, for the test cases of SS contribution with various weights except the case of amino acid sequence only.

Table 9.2: Mean AUC Actual and Predicted Secondary Structure Contribution at various weights in HHsearch method

		α=0.00	α=0.25	α=0.50	α=0.75	α =1.00
Actual SS.	Class 1	80.0%	75.0%	70.0%	76.0%	80.0%
	Class 2	84.0%	83.0%	79.0%	77.0%	80.0%
	Class 3	84.0%	84.0%	83.0%	87.0%	83.0%
	Class 4	85.0%	83.0%	80.0%	81.0%	84.0%
	Class 5	83.0%	82.0%	82.0%	81.0%	82.0%
Predicted SS	Class 1	76.0%	78.0%	78.0%	74.0%	71.3%
	Class 2	81.0%	82.0%	81.0%	82.0%	78.9%
	Class 3	82.0%	83.0%	83.0%	83.0%	81.1%
	Class 4	83.0%	82.0%	82.0%	82.0%	79.6%
	Class 5	84.0%	82.0%	82.0%	83.0%	82.2%

9.3 Profile-Profile Comparison with PRC

PRC, the profile-profile comparison tool used in this study, produces three scores as outputs of comparison computation: *simple score*, *coemission score*, *reverse score*. All three are used to generate similarity matrix separately. Then, we obtained three distinct similarity matrices and three classification models. The results are given again with ROC curves and mean AUC values at Figures 9.14, 9.15, 9.16 and Table 9.3. Figure 9.14 displays the results obtained with similarity score generated using *simple score* of PRC output. The results are unfortunately not brilliant. At the worst case which is label 4, the classifier results with 50% AUC which means that it is as good as a random classifier. At the best case which is class 3, the AUC value is 62% which is not a satisfying result.

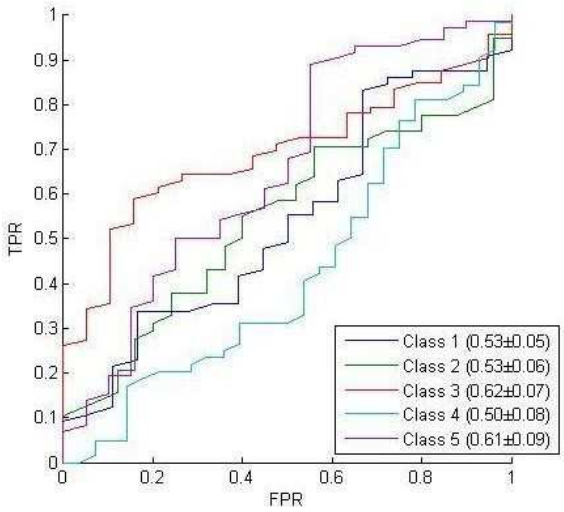

Figure 9.14: ROC curves and mean AUC values obtained using simple score of PRC in KNN classifier

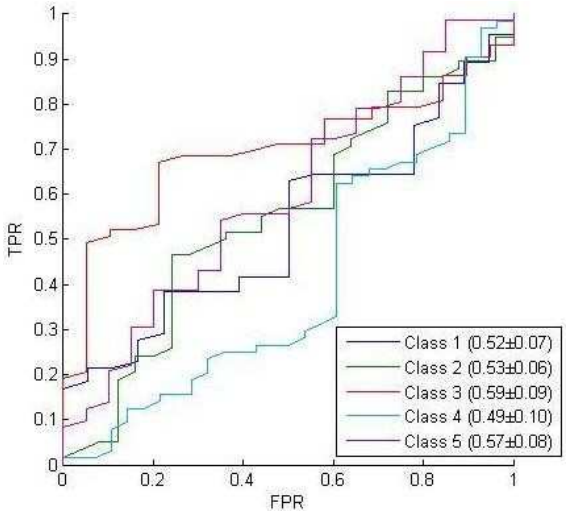

Figure 9.15: ROC curves and mean AUC values obtained using coemission score of PRC in KNN classifier

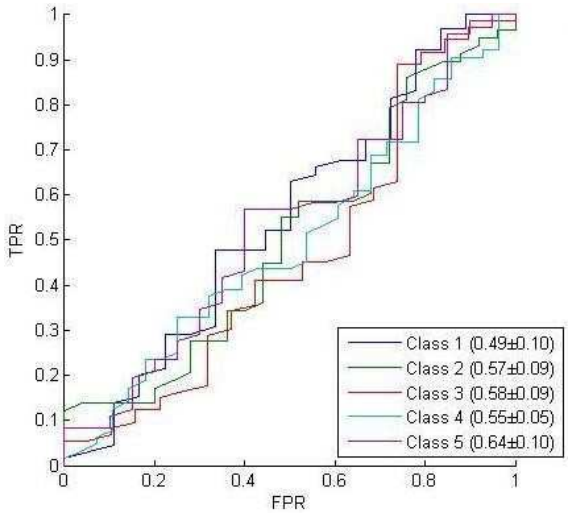

Figure 9.16: ROC curves and mean AUC values obtained using reverse score of PRC in KNN classifier

Figure 9.15 presents results obtained using *coemission score* output of PRC's. These results are even worse than the previous case. The AUC at the case of label 4, now, falls below 50% which means that the classifier in this case is worse than the random classifier. And note that in this test, no AUC exceeds 60% limit. Figure 9.16 shows the

results obtained using PRC's *reverse score* as similarity measure. Except one case which is label 1, AUC values seem more or less increased comparing with 2 previous tests. The classifier at class 5 almost reaches 65% and in other cases it exceeds the limit of 55%. Even though, these results are weak. Results for PRC's class-based method are ommitted since there's no significant difference with those given here.

Table 9.3: AUC values obtained by search score and e-value score

	simple score	coemission score	reverse score
AUC	0.53	0.52	0.49
	0.53	0.53	0.57
	0.62	0.59	0.58
	0.5	0.49	0.55
	0.61	0.57	0.64
std dev	0.05	0.07	0.1
	0.06	0.06	0.09
	0.07	0.09	0.09
	0.08	0.1	0.05
	0.09	0.08	0.1

9.4 Improved Profile Methods

The results given under subsections listed below are evaluated using “accuracy”. They are all given in tables. ROC and AUC are two more popular measures since accuracy can sometimes give unreliable results. But in this case they are given in order to have a general idea about the classifiers performance.

9.4.1 Sequence-Profile Comparison with NR Data

The results given below belongs to one of improved profile methods which is described in section 6.5.1, sequence-profile comparison in best-match method in NR data. Note that in this method, profile-HMM's are generated using HMMER and sequence-profile comparisons are computed using again HMMER. There are two diverse scores put out by HMMER's comparison: search score and e-value score. We have used e-value score

as the similarity measure since we have better results in tests described in section 9.1. In addition, in the output, the similarity results are sorted with respect to e-value. The parameter n determines the number of neighbors which are considered in Best-Match's voting step. Finally, the computation time of this test is 3-4 days.

Table 9.4: Accuracy of Best-Match with n = 1, using NR data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.85	0.83	0.86	0.81	0.85	0.81	0.82	0.9	0.9	0.82	0.84	0.01
2	0.85	0.83	0.85	0.9	0.9	0.9	0.78	0.87	0.87	0.89	0.86	0.01
3	0.94	0.84	0.86	0.88	0.86	0.92	0.92	0.86	0.94	0.88	0.89	0.01
4	0.88	0.84	0.79	0.9	0.83	0.84	0.87	0.81	0.86	0.9	0.85	0.01
5	0.88	0.95	0.97	0.87	0.86	0.86	0.92	0.95	0.94	0.89	0.91	0.01

Results of this method's tests are given with Tables 9.4- 9.8. Row identities are given according to label identities and column identities are given as fold identities; i.e. a $cell_{ij}$ gives the classification accuracy for label i and fold j. The final column is reserved for mean of results along a row. In table 9.4 which gives the results for n=1, the accuracies varie between 78% and 97%. At average, accuracies are form 84% to 91% which are satisfying. Even in one case, the accuracies reach to 97% which means that only a single or two test instances through 78 are false classified. The rest of the classification is true. Table 9.5 gives the results for n=3 which means that 3 training samples closest to the current test sample, are taken into consideration. The results decreas slightly with respect to the previous test case. But still the mean accuracies are above 80%, and we have an accuracy at 90%.

Table 9.5: Accuracy of Best-Match with n = 3, using NR data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.79	0.82	0.86	0.78	0.81	0.85	0.81	0.83	0.85	0.86	0.83	0.01
2	0.83	0.81	0.82	0.86	0.85	0.87	0.82	0.82	0.82	0.86	0.84	0.01
3	0.9	0.86	0.82	0.88	0.91	0.86	0.9	0.84	0.91	0.82	0.87	0.01
4	0.83	0.81	0.78	0.86	0.81	0.79	0.82	0.77	0.83	0.83	0.81	0.01
5	0.9	0.94	0.95	0.82	0.88	0.9	0.95	0.92	0.92	0.86	0.9	0.01

Table 9.6 presents the results for n = 7. These results are not significantly different from n = 3 case. But, apparently, there is a general decrease. Table 9.7 shows the results for n = 11. This case, although presenting non significantly, has a relative decrease comparing with previous cases. But in average accuracies, we have still 90%.

Table 9.6: Accuracy of Best-Match with n = 7, using NR data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.81	0.82	0.82	0.83	0.81	0.85	0.79	0.85	0.83	0.82	0.82	0.01
2	0.83	0.78	0.79	0.83	0.82	0.83	0.82	0.86	0.83	0.81	0.82	0.01
3	0.87	0.84	0.84	0.87	0.91	0.86	0.86	0.83	0.9	0.82	0.86	0.01
4	0.83	0.84	0.75	0.83	0.82	0.86	0.81	0.82	0.83	0.78	0.82	0.01
5	0.91	0.92	0.92	0.86	0.91	0.87	0.94	0.91	0.88	0.87	0.9	0.01

Table 9.8 stands for the case n = all, which means that each detected similarity between the current test instance and all train instances are taken into consideration. So far, there was no result below 80% at average. But in this test, it's remarkable that, in average accuracies, three labels fell below 80% and two of them almost reach to 70%. In addition non of mean accuracies is at 90%. This means without hesitation that, as n increases the accuracies decreas. Probably, this is caused by noise let as more instances far from test instance are let vote. Even, the classification has weighted voting, which actually means that if a train instance is far from test instance, the noise that it can join would be weak. But note that the distance of train instances included as n is increased, may not increase linearly with respect to n. Increasing n may take the contribution of instances from different classes which are not far enough.

Table 9.7: Accuracy of Best-Match with n = 11, using NR data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.85	0.78	0.79	0.85	0.76	0.86	0.74	0.83	0.79	0.8	0.81	0.01
2	0.82	0.77	0.81	0.86	0.82	0.83	0.79	0.83	0.83	0.8	0.82	0.01
3	0.86	0.86	0.83	0.84	0.9	0.86	0.84	0.84	0.9	0.82	0.85	0.01
4	0.79	0.83	0.78	0.83	0.83	0.82	0.81	0.79	0.84	0.77	0.81	0.01
5	0.91	0.92	0.91	0.86	0.9	0.86	0.92	0.91	0.88	0.85	0.89	0.01

Table 9.8: Accuracy of Best-Match with n = all, using NR data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0
2	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.7	0.73	0
3	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.79	0.84	0.01
4	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.7	0.74	0
5	0.86	0.86	0.86	0.86	0.86	0.86	0.86	0.86	0.86	0.78	0.85	0.01

9.4.2 Sequence-Profile Comparison with Enriched 5-class Data

The results given in this section are obtained with the method described in section 6.5.2, “Sequence-Profile Comparison in Best-Match Approach using 5-class Enrich Data” in the section titled as “Improved Profile Methods”. Note that in this method, the data used is different from the previous method. However, this update causes some modifications in the order of steps. In the previous methods, profile-HMM's are generated over train set and train/test partition is made at the very beginning. Although, in NR data case, the profile-HMM of a sequence is used both cases of train and test, directly, profile-HMM's of test sequences in this method must be recomputed even though they might be train instances in other folds. This multiplied the computation cost of the test. The results are given as accuracies, again in order to have general idea. 5 different cases are computed. In Each case, the parameter n of Best-Match is modified. This experiment took 3-4 days of computation with 10 CPU's in parallel run.

Table 9.9: Accuracy of Best-Match with n = 1, using enrich 5-class data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.51	0.67	0.67	0.72	0.68	0.74	0.63	0.66	0.7	0.63	0.66	0.02
2	0.6	0.68	0.62	0.67	0.61	0.73	0.73	0.66	0.64	0.63	0.66	0.01
3	0.68	0.69	0.64	0.73	0.76	0.71	0.67	0.68	0.72	0.58	0.69	0.02
4	0.59	0.57	0.6	0.59	0.63	0.61	0.64	0.56	0.65	0.58	0.6	0.01
5	0.68	0.66	0.54	0.65	0.71	0.57	0.67	0.59	0.58	0.63	0.63	0.02

The results of this method are given in Tables 9.9-9.13. Table 9.9 gives the results for n = 1. The accuracies varies from 51% to 94%. 2 of the mean accuracies are below 65%. And the top mean accuracy is 69%. The results are low with respect to previous method's case of n = 1.

Table 9.10: Accuracy of Best-Match with n = 3, using enrich 5-class data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.67	0.61	0.7	0.72	0.69	0.74	0.74	0.69	0.73	0.67	0.7	0.01
2	0.6	0.74	0.67	0.67	0.66	0.68	0.73	0.65	0.69	0.63	0.67	0.01
3	0.69	0.7	0.7	0.81	0.79	0.74	0.67	0.74	0.75	0.59	0.72	0.02
4	0.65	0.6	0.67	0.61	0.66	0.63	0.67	0.57	0.67	0.63	0.64	0.01
5	0.65	0.69	0.51	0.68	0.64	0.6	0.71	0.63	0.62	0.65	0.64	0.02

Table 9.10 presents the results for n = 3. The accuracies are rather lifted. Almost no accuracy except one is below 60% and we have a 79% at top. The mean accuracies are better. Table 9.11 which shows results for the case n = 7 has slightly better results. Accuracies of only 4 cases among 50 stays below 65%, top accuracy which was 79% in previous test is 84%, however, mean accuracies are either kept or improved 1-4% with respect to previous case. Table 9.12 exhibits results for the case of n = 11. The mean accuracies increased by 1% at each label except label 4 and 5. Actually, it is not a significant progress with respect to the case of n = 7. And finally Table 9.13 presents accuracies for the case of n = All, i.e. all training instances are included in voting given a test instance. In all labels, accuracies increase. Most significant increase happens at lable 3 as 9% of difference.

Table 9.11: Accuracy of Best-Match with n = 7, using enrich 5-class data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.68	0.64	0.66	0.71	0.72	0.73	0.72	0.71	0.73	0.68	0.7	0.01
2	0.68	0.74	0.7	0.69	0.69	0.67	0.71	0.7	0.68	0.66	0.69	0.01
3	0.73	0.72	0.7	0.84	0.79	0.79	0.67	0.78	0.76	0.68	0.74	0.02
4	0.73	0.68	0.7	0.69	0.7	0.65	0.71	0.59	0.68	0.64	0.68	0.01
5	0.65	0.69	0.55	0.69	0.71	0.64	0.74	0.65	0.68	0.74	0.67	0.02

Table 9.12: Accuracy of Best-Match with n = 11, using enrich 5-class data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.71	0.63	0.67	0.72	0.72	0.73	0.74	0.71	0.74	0.69	0.71	0.01
2	0.68	0.76	0.74	0.69	0.71	0.67	0.73	0.7	0.68	0.66	0.7	0.01
3	0.72	0.72	0.71	0.82	0.81	0.79	0.69	0.81	0.76	0.71	0.75	0.02
4	0.73	0.71	0.7	0.69	0.69	0.67	0.69	0.61	0.68	0.68	0.68	0.01
5	0.67	0.69	0.56	0.66	0.71	0.64	0.77	0.66	0.62	0.74	0.67	0.02

Table 9.13: Accuracy of Best-Match with n = All, using enrich 5-class data in sequence-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.75	0.76	0.77	0.77	0.76	0.77	0.76	0.74	0.77	0.74	0.76	0
2	0.7	0.77	0.74	0.71	0.73	0.71	0.74	0.73	0.71	0.7	0.72	0.01
3	0.83	0.82	0.84	0.85	0.86	0.86	0.79	0.9	0.85	0.79	0.84	0.01
4	0.73	0.71	0.73	0.73	0.72	0.72	0.72	0.67	0.7	0.67	0.71	0.01
5	0.67	0.73	0.66	0.7	0.79	0.66	0.73	0.73	0.68	0.73	0.71	0.01

9.4.3 Profile-Profile Comparison with NR Data

The method used in this experiment can be outlined as, generating profile-HMM's for each instance in 5-class GO data over NR space, then classifying of test instances using Best-Match classification method where the similarity measure used to compute similarities between test and train instances is profile-profile comparison using PRC tool. Remind that this method is described in details, in section 6.5.3 – Profile-Profile in Best-Match Method using NR Data. As stated previously, PRC tool puts out three different measure scores: *simple score*, *coemission score*, *reverse score*. Each of them can be held as a similarity metric. Remember that these scores are compared in experiment of section 9.3 – Profile-Profile Comparison with PRC. As clearly seen in Table 9.3, best accuracies are obtained with *reverse score* metric. Although all three outputs are stored at each comparison during this test, reverse score is selected and used as similarity measure in Best-Match. This experiment's time cost is approximately 10 days with 6 CPU's in parallel computation. And the results are given in Tables 9.14-9.18.

Table 9.14 shows results for case n = 1. Remember that, n is a parameter of Best-Match which is used to define the number of train instance allowed in weighted voting. In this test case, only the closest train instance is referred for the current test instance. The accuracies for each fold of each label varies from 86% to 96%. At average, two labels are at 89% and the others are above 90%.

Table 9.14: Accuracy of Best-Match with n = 1, using NR data in profile-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.88	0.9	0.91	0.9	0.87	0.85	0.92	0.9	0.91	0.86	0.89	0.01
2	0.94	0.86	0.94	0.93	0.85	0.95	0.91	0.91	0.89	0.93	0.91	0.01
3	0.92	0.89	0.96	0.95	0.92	0.94	0.96	0.9	0.93	0.9	0.93	0.01
4	0.86	0.94	0.87	0.95	0.88	0.89	0.93	0.88	0.84	0.89	0.89	0.01
5	0.9	0.95	0.95	0.88	0.91	0.92	0.93	0.93	0.93	0.95	0.92	0.01

Table 9.15 presents results for the case of n = 3. The accuracy interval within accuracies are distributed is dilated to 83%-97%, hence the standard deviation seems as increased. Nevertheless, all mean accuracies except label 2 improves also with respect to the previous case. As the final remark, 4th fold of label 4 has 100% as accuracy, i.e. each test instance is correctly classified.

Table 9.15: Accuracy of Best-Match with n = 3, using NR data in profile-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.9	0.92	0.94	0.9	0.86	0.88	0.94	0.91	0.92	0.84	0.9	0.01
2	0.96	0.9	0.92	0.93	0.92	0.91	0.95	0.91	0.88	0.93	0.92	0.01
3	0.95	0.88	0.93	0.92	0.91	0.96	0.96	0.9	0.95	0.91	0.93	0.01
4	0.84	0.95	0.89	1	0.88	0.88	0.92	0.88	0.83	0.9	0.9	0.02
5	0.94	0.95	0.96	0.88	0.93	0.97	0.97	0.95	0.97	0.96	0.95	0.01

Table 9.16 presents the results for the case of n = 7. No significant difference can be distinguished at individual accuracies with respect to previous case. Same fact is valid for mean accuracies, also. Still a 100% accuracy is available, at the 7th fold of label 5.

Table 9.16: Accuracy of Best-Match with n = 7, using NR data in profile-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.92	0.9	0.95	0.91	0.86	0.86	0.95	0.91	0.91	0.88	0.9	0.01
2	0.95	0.88	0.92	0.92	0.95	0.94	0.95	0.95	0.89	0.95	0.93	0.01
3	0.96	0.87	0.93	0.93	0.92	0.95	0.96	0.9	0.91	0.92	0.92	0.01
4	0.84	0.94	0.89	0.99	0.91	0.92	0.95	0.88	0.87	0.92	0.91	0.01
5	0.94	0.93	0.95	0.9	0.93	0.96	1	0.93	0.96	0.95	0.94	0.01

Table 9.17 exhibits results for the case of n = 11. The mean accuracies are lifted about 1% or kept constant with respect to the previous case. A very interesting fact in these results is that there are 2 100% accuracies which are exactly the same test set cases of the 2 previous cases (4^{th} fold of label 4 and 7^{th} fold of label 5).

Table 9.17: Accuracy of Best-Match with n = 11, using NR data in profile-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.92	0.9	0.95	0.92	0.85	0.86	0.96	0.88	0.91	0.89	0.9	0.01
2	0.96	0.9	0.92	0.93	0.94	0.95	0.94	0.92	0.93	0.95	0.93	0.01
3	0.96	0.87	0.93	0.93	0.93	0.95	0.96	0.91	0.91	0.93	0.93	0.01
4	0.86	0.96	0.89	1	0.95	0.93	0.96	0.86	0.88	0.9	0.92	0.02
5	0.95	0.95	0.96	0.9	0.95	0.96	1	0.96	0.96	0.95	0.95	0.01

Table 9.18 shows the results for the case of n as all instances in train set. The significant change with respect to the previous case is that there are 2 mean accuracies below 90% and 2 other accuracies are set to 90%. This means that the general accuracy of the classifier is decreased. In addition, no 100% accuracy exists among individual accuracies.

Table 9.18: Accuracy of Best-Match with n = All, using NR data in profile-profile comparison

Label/Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.88	0.87	0.86	0.86	0.82	0.85	0.9	0.87	0.86	0.84	0.86	0.01
2	0.92	0.84	0.94	0.92	0.87	0.86	0.86	0.92	0.85	0.89	0.89	0.01
3	0.91	0.88	0.89	0.91	0.88	0.94	0.91	0.86	0.92	0.9	0.9	0.01
4	0.9	0.91	0.89	0.96	0.94	0.89	0.92	0.86	0.9	0.85	0.9	0.01
5	0.96	0.95	0.96	0.9	0.93	0.97	0.97	0.95	0.95	0.92	0.95	0.01

9.4.4 Profile-Profile Comparison with Enriched 5-class Data

The final experiment is profile-profile comparison with enriched 5-class data using Best-Match classifier. Remember that, the enriched 5-class data is already partitioned as train and test sets for each fold at each label. For the experiment of sequence-profile comparison with Best-Match classifier which is described in section 9.4.2, the profile-HMM's are generated for train set sequences at each fold of each label. Obviously, there are some instances in the data set, probably a significant portion, which are in the test set of a fold of a label and in the train set of another fold. As a consequence, profile-HMM's are built for these test instances. Nevertheless, they cannot be used in the case where they are test instances since, in this data set, profile-HMM's are generated with respect to the train set of the corresponding fold since profile-HMM's are built with respect to another train set within these subject test instances are included. Hence, additional computation is added for profile-HMM generation of test instances. Note that profile-profile comparison takes much longer than sequence-profile comparison. Following some initial computation of the experiment, the estimation of duration for this experiment is guessed as 40 days, even with 10 CPU's in parallel computation. This experiment cannot be finished until the writing of the thesis, it is still running. Only results for first label are presented. Those results are obtained for the cases of n in {1,3,7,11 and all neighbors}. Table 9.19 presents the results for all n, for each of 10 folds of label 1. Few accuracies are below 60%, the rest are above up to 78%. mean accuracies are from 61% to 76%.

Table 9.19: Accuracy of Best-Match with n = All, using NR data in profile-profile comparison

n\Fold	1	2	3	4	5	6	7	8	9	10	Avg	ErrBar
1	0.62	0.68	0.68	0.56	0.55	0.64	0.6	0.59	0.6	0.6	0.61	0.01
3	0.64	0.71	0.72	0.56	0.58	0.64	0.6	0.6	0.63	0.66	0.63	0.02
7	0.65	0.73	0.68	0.58	0.62	0.64	0.59	0.68	0.62	0.69	0.65	0.02
11	0.68	0.76	0.67	0.63	0.63	0.65	0.62	0.68	0.64	0.71	0.67	0.01
All	0.74	0.78	0.77	0.73	0.73	0.77	0.76	0.78	0.77	0.76	0.76	0.01

10. CONCLUSION

In this study, the protein function prediction problem is attempted to be solved using profile Hidden Markov Models with Machine Learning techniques. Profile-HMM comparison is considered as the key method to measure protein similarity. Two different types of protein information are used in profile-HMM generation: amino acid sequence and secondary structure sequence. Various computation steps such as data retrieval, local alignment, multiple sequence alignment, profile-HMM generation, sequence-profile comparison, profile-profile comparison, classifier modeling and statistical evaluation are applied separately. First, the previous work is observed and accurate profile-HMM tools which are used in other bioinformatics problems such as fold recognition or remote homology detection are selected to be used.

The whole implementation of this study can be considered in two section. First experiment set of which the methods are discussed in sections 6.2-6.4, and second experiment set of which the experiments are discussed in section 6.5. In the first section, the methods are built on the hypothesis that profile comparison which is proven to be an accurate similarity measure for fold recognition and remote homology detection is also an accurate similarity measure for function prediction. The experiments are executed and the results are evaluated. The second experiment set is built on solution propositions for problems observed on the first experiment set. The experiment conditions are met and the results are evaluated again.

In the first set of experiments, HMMER for sequence-profile comparison, HHsearch for profile-profile comparison and PRC for profile-profile comparison are used. The classification results obtained using HMMER are much worse than expected. Profile-

HMM techniques are expected to result better than sequence-sequence comparison methods, since profile-HMM includes homology and residue conservation into similarity measures. Nevertheless the results obtained with HMMER are approximately at 60-70% of AUC, where sequence-sequence comparison techniques result in 90%. Similarly, profile-profile comparison with PRC gives approximately 50-60% of AUC at average. We expected sequence-profile comparison to outperform sequence-sequence comparison methods, and profile-profile comparison to outperform sequence-profile comparison methods. None of them happened in the first set of experiments. We tried HHsearch as another profile-profile comparison method. In this method, we checked the contribution of secondary structure in profile-profile comparison. The results obtained are 82-87% of AUC at best, with 25% secondary structure contribution weight for predicted secondary structure and 80-85% of AUC at best with 100% secondary structure contribution weight for actual secondary structure. These results are fairly better than former results. However, the sequence-sequence comparison methods performs better than other ones, which is not the results that we have predicted.

The problems encountered in the first experiment set can be caused by two distinct conditions. The first possible cause is data. The number of training instances in the data set is not enough to train profile-HMM's. The solution propositions tested are using NR data and some other data to enrich the training set. The second possible cause is guessed to be the classification algorithm. Although pairwise similarity matrix results are good when other similarity metrics are used in function prediction, it might fail in profile-HMM comparison methods. Hence a new simple Best-Match algorithm is used.

A number of experiments are performed taking into account these ideas. In the first enhanced method which is sequence-profile comparison with HMMER and NR data, NR data is used in addition to the training sequences when producing multiple sequence alignments. The mean accuracies are between 84-91%. They get worse as the parameter *n* in the Best-Match is increased. This is probably due to noise which is increases as the

number of participating train profile-HMM's increases.

The second enhanced method is again sequence-profile comparison with HMMER but on enriched data instead of NR data. The enrichment is performed just as in the NR data. However, sequences of the function which are far from the test set but close enough to the train set are used in enrichment. The mean accuracies are between 68-75%, and the best case is achieved with Best-Match parameter n = all. This time, as n is increased the results get better. The possible cause is that although the data set used is enriched, i.e. the population of train instances in data set is higher than the 5-class GO data, it is still much sparser than NR data. So, again the number of train instances might be insufficient to train profile-HMM's. Thus the more we include train instances in weighted voting of the Best-Match algorithm, the better accuracy the classifier has.

The third enhanced method is the profile-profile comparison using PRC with NR data. The mean accuracies are in the interval of 90-95% for n equal to 3. This is the best result ever in this study. Close accuracies are obtained for other n values except n = all where the accuracies go down to 86%. The possible cause is that, again, the number of instances included in profile-HMM training is enough and as the all train instance are included in weighted voting, the noise can be increased significantly.

The fourth enhanced method is profile-profile comparison using PRC with enhanced data. It is not right to infer a general conclusion according to early results already obtained. But we can have some idea. The accuracy results are in the interval of 61-76%. They increase as n increases. It seems that the same situation with previous enhanced method with enriched data is valid for this case. It might be that the number of train instances is not enough to train profile-HMM's.

In general, we can deduce that the size of data is the major actor in the success of profile-HMM's. Note that the number sequence in NR data is around 6,000,000 where the number of sequence in enriched 5-class GO data is around 1400. The classifier can

also have a major role. When we compare the results of enriched data methods with the first set of experiments, we can see that the accuracies increase. Both data enrichment and using a different classifier algorithm can have a role on this increase.

As a future task, 27-class GO **(Filiz et al., 2008)** data can be used on these enhanced methods. Since there are around 5000 instances in train sets of enriched 27-class GO data, the profile-HMM's can be trained significantly better, hence we can obtain better results. Note that the 27-class GO data makes the problem more tough than it is with the 5-class GO data, as there are 5 times more classes in data set. The probability of misclassifying in this data set is much higher than the previous one.

As the second future task, the Best-Match classifier algorithm can be replaced by a classifier known and proven to be accurate such as SVM. SVM classifier's computation cost is much higher K-NN and Best-Match. Nevertheless, since there's no significant cost in K-NN and Best-Match, the computation cost of SVM should not be considered as a significant handicap, at least, at this moment.

The computation cost is a concrete problem when PRC is used. We have talked about computation duration in the order of days, even 40 days. This is not admissible for a classification method. As the third future task, the method can be used in much better performing computers of ITU. Although we have tested the methods using PRC with 10 CPU's working in parallel, ITU has supercomputers providing round 200 CPU's which can work in parallel. Using these computers, the computation time can be reduced to 2-3 days.

As the fourth future task, HHsearch can be used with NR data and enriched data, and also with the Best-Match methods. Remember that HHsearch outperformed HMMER and PRC in first experiment set. Thus we can expect to obtain better results with HHsearch if we use it as a part of second experiment set. In addition, HHsearch can be used with 27- class GO data. It can give good results with this new data.

As the final future task, secondary structure can be used in HMM methods. There are some studies about secondary structure contribution in HMM. However, those studies are about fold recognition problem **(Hargbo and Eloffson, 1999)**, **(Karchin et al., 2003)**. Note that we have used secondary structure information in HHsearch method. It was a small part of the study. Secondary structure contribution deserves much deeper work.

REFERENCES

Alpaydın, E., 2004. Introduction to Machine Learning, p. 158. *MIT Press,* London.

Ashburner, M., 1998. On the representation of gene function in genetic databases, *Proceedings of the Intelligent Systems for Molecular Biology*, vol. 6, 1998.

Ashburner, M., Ball, C., Blake, J., Botstein, D., Butler., H., Cherry, H., Davis, A., Dolinski, K., Dwight, S. and Eppig, J., 2000. Gene Ontology: tool for the unification of biology, *Nature Genetics*, vol. **25**, pp. 25–29, 2000.

Altschul, S.F., Gish, W., Miller, W., Myers, E W. and Lipman, D. J., 1990. Basic Local Alignment Search Tool. *J. Mol. Bio.,* (1990) 215, 403-410.

Altschul, S.F. and Koonin, E. V., 1998. Iterated Profile Searches with PSI-BLAST – A Tool for Discovery in Protein Databases, *ScienceDirect*, Trends in Biochemical Sciences, Volume **23**, Issue 11, pp 444-447.

Aygün, E., Kömürlü, C., Çataltepe, Z. and Aydın, Z., 2008. Protein function prediction with amino acid sequence and secondary structure alignment score. *Hibit 2008*.

Bateman, A. and Finn,R. D., 2007. SCOOP: a simple mehod for identification of novel protein superfamily relationships, Bioinformatics 2007 23(7):809-814.

Berman, H.M., Westbrook J., Feng Z., Gilliland G., Bhat T. N., Weissig H., Shindyalov I.N. and Bourne P. E., 2000. The Protein Data Bank. *Nucleic Acids Research*, 28 pp. 235-242.

Branden, C. and Tooze, J., 1999. Introduction to Protein Structure, 2nd ed. *Garland Publishing,* New York, NY.

Brenner, S.E., Chotia, C. and Hubbard T.J.P., 1998. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci.,* USA. 1998 May 26; 95(11): 6073–6078.

Bystroff, C., Thorsson, V. and Baker, D., 2000. HMMSTR: a hidden markov model for local sequence structure correlations in proteins. *Nucleic Acids Research*, *Journal of Molecular Biology*:173– 90, 2000.

Camon E., Magrane M., Barrell D., Lee V., Dimmer E., Maslen J., Binns D., Harte N., Lopez R. and Apweiler R., 2004. The Gene Ontology Annotation (GOA) Database: sharing knowledge in Uniprot with Gene Ontology. *Nucleic Acids Research* 32(1): D262-D266.

Chan, S.C., Wong, A.K.C., Chiu and D.K.Y., 1992. A Survey of Multiple Sequence Comparison Methods. *Bulletin of Mathematical Biology,* Vol. 54, No. 4, pp. 563-598.

Cheng, J. and Baldi, P., 2006. A machine learning information retrieval approach to protein fold recognition. *Bioinformatics*, 22(12):1456–1463, 2006.

Chenna, R., Sugawara, H., Koike, T., Lopez, R., Gibson, T.J., Higgins, D.G. and Thompson, J. D., 2003. Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Res* **31**: 3497–500.

Dewey, C.N. and Patcher, L., 2006. Evolution at the nucleotide level: the problem of multiple whole-genome alignment. *Human Molecular Genetics,* 15 (Review issue 1), pp. R51-R56.

Durbin, R., Eddy, S., Krogh, A., Mitchison, G., 1998. Biological sequence analysis: Probabilistic models of proteins and nucleic acids, pp. 2-3. *Cambridge University Press,* Cambridge.

Durbin, R., Eddy, S., Krogh, A., Mitchison, G., 1998. Biological sequence analysis: Probabilistic models of proteins and nucleic acids, p. 101. *Cambridge University Press,* Cambridge.

Eddy, S.R., 1996. Hidden Markov models. *Current Opinion in Structural Biology*, Volume 6, Issue3, pp. 361-365.

Eddy, S.R., 1998. HMMER: Profile hidden Markov models for biological sequence analysis.

Eddy, S.R., 2003. HMMER User's Guide: Biological sequence analysis using profile hidden Markov Models. Saint Louis, Missouri, USA.

Filiz, A., Aygün, E., Çataltepe, Z. and Keskin, Ö., 2008. Importance of secondary structure elements for prediction of GO annotations, *Hibit2008.*

Fitch, W. M., 2000. Homology a personal view on some of the problems. *Trends in Genetics, v*olume 16, issue 5, pp. 227-231.

Gribskov, M., McLachlan, A. D., Eisenberg, D., 1987. Profile Analysis: Detection of distantly related proteins. *Proc. Natl. Acad. Sci. USA*, Vol. 84, pp. 4355-4358, July 1987.

Hamilton, H., 2007. Knowledge Discovery in Databases, Course Notes, http://www2.cs.uregina.ca/~hamilton/courses/831/index.html

Hargbo, J. and Elofsson, A., 1999. Hidden Markov Models That Use Predicted Secondary Structures For Fold Recognition. *PROTEINS: Structure, Function, and Genetics,* 36:68-76.

Higgins, D. G., Sharp P. M., 1988. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene* **73** (1): 237–44.

Hirosawa, M., Totoki, Y., Hoshida, M., Ishikawa, M., 1995. Comprehensive study on iterative algorithms of multiple sequence alignment. *Comput Appl Biosci* **11**: 13–8.

Hughey, R., Karplus, K. and Krogh, A., 2003. SAM sequence alignment and modelling software system.

Huynen, M.A., Snel, B., Mering, C.V. and Bork, P., 2003. Function Prediction and protein networks. *Current Opinion in Cell Biology*, 15: 191-198.

Joachims, T., 2003. Transductive Learning via Spectral Graph Partitioning, *Proceedings of the International Conference on Machine Learning (ICML)*.

Joachims, T., 2003. Learning to Align Sequences: A Maximum-Margin Approach, *Technical Report*, August.

Jones, D.T., 1999. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292, 195–202.

Kabsch, W. and Sander, C., 1983. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, Vol. 22, No. 12, pp. 2577-637, 1983.

Karchin, R., Cline, M., Mandel-Gutfreaund, Y. and Karplus, K., 2003. Hidden Markov Models That Use Predicted Local Structure for Fold Recognition: Alphabets and Backbone Geometry. *PROTIENS: Structure, Function, and Genetics*, 51:504-514.

Karplus K, Barrett C. and Hughey R., 1998.Hidden Markov models for detecting remote protein homologies. *Bioinformatics,* **14** (10): 846–856.

Kersey P.J., Duarte J., Williams A., Karavidopoulou Y., Birney E. and Apweiler R., 2004. The International Protein Index: An integrated database for proteomics experiments. *Proteomics* 4(7): 1985-1988.

Landau, M., Mayrose, I., Rosenberg, Y., Glaser, F., Martz, E., Pupko, T. and Tal-Ben, N., 2005. ConSurf 2005: the projection of evolutionary conservation scores of residues on protein structures. *Nucleic Acid Research,* Vol. 33, Web Server issue, W299-W302.

Lassmann, T. and Sonnhammer, E.L.L., 2005. Kalign – an accurate and fast multiple sequence alignment algorithm, *BMC Bioinformatics,* 6:298, 2005.

Lindahl, E. and Eloffson, A., 2000. Identification of Related Proteins on Family, Superfamily and Fold Level. *J. Mol. Bio.*, 295, pp. 613-625, 2000.

Lipman, D.J., Altschul, S.F. and Kececioglu, J.D., 1989. A tool for multiple sequence

alignment. *Proc. Natl. Acad. Sci.* USA, 86: 4412-5.

Lüthy, R., Xenarios, I. and Bucher, P., 1994. Improving the sensitivity of the sequence profile method. *Protein Science,* (1994) 3:139-146.

Madera, M., 2002, The profile comparer, *Ph.D. Thesis*.

Nair, S.A., 2007. Computational Biology & Bioinformatics: A Gentle Overview, *Communications of the computer society of India.*

NCBI, National Center of Biotechnology Information, Fasta Format, http://www.ncbi.nlm.nih.gov/blast/fasta.shtml

NCBI, National Center of Biotechnology Information, NR Data Source, ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nr.gz

NCBI, National Center of Biotechnology Information, Glossary, http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/glossary2.html

Ohsen, N.V., Sommer, I., Zimmer, R. and Lengauer., T., 2004. Arby: automatic protein structure prediction using profile–profile alignment and confidence measures. *Bioinformatics*, 20(14):2228–2235.

Petsko, G.A. and Dagmar, R., 2004. Protein structure and Function, p. 7. New Science Press Ltd., London.

Petsko, G. A., Dagmar, R., 2004. Protein structure and Function, pp. 4-5. New Science Press Ltd., London.

Pfam Protein Families Database, The, Oxford Journals, 2002.

RCSB Protein Data Bank, http://www.rcsb.org/.

Sadreyev, R.I. and Grishin, N. V., 2003. Compass: a tool for comparison of multiple protein alignments with assessment of statistical significance. *J. Mol. Biol.*, 326:317–336, 2003.

Söding, J., 2005. Protein homology detection by hmm-hmm comparison. *Bioinformatics*, 21:951–960, 2005.

Thompson, J.D., Higgins, D. G. and Gibson, T.J., 1994. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res* **22**: 4673–80.

Tramontano, A., 2007. Protein architecture, *in Introduction to Bioinformatics*, pp. 20-26, CRC Press Taylor & Francis Group, London.

Tramontano, A., 2007. Protein architecture, *in Introduction to Bioinformatics*, pp. 79-83, CRC Press Taylor & Francis Group, London.

The Universal Protein Resource, http://www.expasy.uniprot.org/.

Yona, G., Levitt, M., 2002. Within the twilight zone: a sensitive profile–profile comparison tool based on information theory. *J. Mol. Biol.*, 315:1257–1275.

Wallqvist, A., Fukunishi, Y., Murphy, L.R., Fadel, A. and Levy, R.M., 2000. Iterative sequence/secondary structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases. *Bioinformatics*, Vol. 16 no. 11 2000 Pages 988-1002.

Wang, L. and Jiang, T., 1994. On the complexity of multiple sequence alignment. *J. Comput. Bio.*, 1: 337-48.

Watson, J.D., Laskowski, R.A. and Thornton, J.M., 2005. Predicting protein function from sequence and structural data. *Current Opinion in Structural Biology*, **15**:275-284.

Wu, C.H., Apweiler R., Bairoch A., Natale D.A., Barker, W.C., Boeckmann B., Ferro S., Gasteiger E., Huang H., Lopez R., Magrane M., Martin M. J., Mazumder R., O'Donovan C., Redaschi N. and Suzek B., 2006. The Universal Protein Resource (UniProt): an expanding universe of protein information. *Nucleic Acids Res.* 34: D187-191.

AUTOBIOGRAPHY

Caner Kömürlü was born on 26th December, 1982, in Istanbul. As he graduated from Lycée de Galatasaray in June 2002, he was accepted to Computer Engineering Department of Galatasaray University. The following year, he was transferred to Computer Engineering Department of Istanbul Technical University. A year later, he started a double major program with Mathematics Engineering. With no loss of year, he finished his undergraduate education in June 2006 at his main major and was in Computer Sciences Program of Informatics Institute, at the end of that summer. The following year he was graduated from double major program. During his MSc. education, he studied with Assoc. Prof. Dr. Zehra Çataltepe in Bioinformatics Project and published a poster and a conference paper. He was financed by TÜBİTAK, with BİDEB-2228 MSc. scholarship.

Publications:

E. Aygun, C. Komurlu, Z. Aydin, Z. Cataltepe, "Protein Function Prediction with Amino Acid Sequence and Secondary Structure Alignment Scores", HIBIT 2008, Istanbul, Turkey.

Z. Cataltepe, E. Aygun, A. Filiz, O. Keskin, C. Komurlu and Y. Altunbasak, "Dimensionality Reduction for Protein Function Prediction", poster at Automated Function Prediction – Biosapiens Joint Special Interest Group Meeting, Vienna, Austria July 2007.