ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE

PROTEIN FOLD CLASSIFICATION AND MOTIF RETRIEVAL METHODS BY USING THE PRIMARY AND SECONDARY STRUCTURES

Ph.D. THESIS

Özlem POLAT

Department of Electronics and Communication Engineering

Electronics Engineering Programme

SEPTEMBER 2015

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE

PROTEIN FOLD CLASSIFICATION AND MOTIF RETRIEVAL METHODS BY USING THE PRIMARY AND SECONDARY STRUCTURES

Ph.D. THESIS

Özlem POLAT
(504082206)

Department of Electronics and Communication Engineering

Electronics Engineering Programme

Thesis Advisor: Prof. Dr. Zümray DOKUR ÖLMEZ

SEPTEMBER 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

PRİMER VE SEKONDER YAPILAR KULLANILARAK PROTEİNLERİN FOLD DÜZEYİNDE SINIFLANDIRILMASI VE MOTİF ÇIKARIMI

DOKTORA TEZİ

Özlem POLAT
(504082206)

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Zümray DOKUR ÖLMEZ

EYLÜL 2015

Özlem POLAT, a Ph.D. student of ITU Graduate School of ScienceEngineering and Technology 504082206 successfully defended the thesis entitled "PROTEIN FOLD CLASSIFICATION AND MOTIF RETRIEVAL METHODS BY USING THE PRIMARY AND SECONDARY STRUCTURES", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Zümray DOKUR ÖLMEZ** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Mehmet KORÜREK** Istanbul Technical University	.....
	Prof. Dr. Ahmet Hamdi KAYRAN Istanbul Technical University	.....
	Prof. Dr. Nizamettin AYDIN Yildiz Technical University	.....
	Asst. Prof. Dr. Gökhan BİLGİN Yildiz Technical University	.....

Date of Submission : **6 July 2015**
Date of Defense : **17 September 2015**

To my spouse and daughter,

FOREWORD

First and foremost, I would like to thank my Ph.D. advisor, Prof. Dr. Zümray Dokur, whose support and guidance made my thesis possible. She has been actively interested in my studies and has always been available to advise me. I am very grateful for her geniality, patience, motivation, enthusiasm and knowledge. I am also very grateful to Prof. Dr. Tamer Ölmez for his scientific advice, many insightful discussions and suggestions; and I have to thank the members of my Ph.D. committee, Prof. Dr. Mehmet Korürek and Prof. Dr. Nizamettin Aydın for their helpful advice in order to improve the thesis studies.

I want to thank Prof. Virginio Cantoni, the director of Pavia University Computer Vision and Multimedia Laboratory. He helped me come up with the thesis topic and guided me almost for a year.

I also thank to my dear friends, Mehmet Tükel, Mustafa Kösem and Berat Doğan, for their support and help when I needed.

Lastly, I would like to thank my family for all their love and encouragement. Especially I thank my supportive, encouraging and patient husband, Dr. Ali Polat, for his faithful support during this time.

September 2015

Özlem POLAT
(Research Assistant)

TABLE OF CONTENTS

Page

ABBREVIATIONS

3D	**:** Three Dimensional
DAG	**:** Directed Acyclic Graph
DSSP	**:** Dictionary of Secondary Structure of Proteins
EGI	**:** Extended Gaussian Image
GAL	**:** Grow and Learn
GHT	**:** Generalized Hough Transform
HS	**:** Hough Space
HT	**:** Hough Transform
NMR	**:** Nucleic Magnetic Resonance
OvO	**:** One-versus-Others
PDB	**:** Protein Data Bank
PGI	**:** Protein Gaussian Image
RNN	**:** Recursive Neural Network
RP	**:** Reference Point
RT	**:** Reference Table
SCOP	**:** Structural Classification of Proteins
SOM	**:** Self-Organizing Map
SOM-SD	**:** Self-Organizing Map for Structured Data
SS	**:** Secondary Structure
SSC	**:** Secondary Structure Couple Co-occurrences
SSS	**:** Single Secondary Structure
SST	**:** Secondart Structure Triplet Co-occurrences
SVM	**:** Support Vector Machine

LIST OF TABLES

Page

LIST OF FIGURES

Page

PROTEIN FOLD CLASSIFICATION AND MOTIF RETRIEVAL METHODS BY USING THE PRIMARY AND SECONDARY STRUCTURES

SUMMARY

Proteins are crucial molecules in biological phenomena because they form much of the functional and structural machinery in every cell in organisms and their function is determined by their spatial structures. Protein structures can be described at various levels in detail, ranging from atomic coordinates, through vector approximations, to secondary structure elements. Protein structure comparison is an important issue that helps biologists understand various aspects of protein function and evolution. It is commonly believed that the 3D fold has a major effect on the ability of a protein to bind other proteins or ligands. The similarity analysis of protein structure is therefore an important process in understanding the protein's role in the machinery of life. Comparison of protein structures is also essential for estimating the evolutionary distances between proteins and protein families. Protein fold classification is also an important problem in bioinformatics and a challenging task for machine-learning algorithms. According to convention a protein could be classified into one of four structural classes based on its secondary structure components; all-α, all-β, α/β, $\alpha+\beta$. Structural Classification of Proteins (SCOP) provides a detailed and comprehensive description of the structural and evolutionary relationships among all proteins whose structures are known. According to SCOP four structural classes are divided into folds. Protein fold classification problem is to determine that the query protein belongs to which fold. In this thesis we deal with two problems related to proteins; protein fold classification and structural block comparison (motif retrieval).

Proteins are formed by two basic regular 3D structural patterns called *secondary structures*; helices and strands. A structural *motif* is a compact 3D protein structure referring to a small specific combination, which appears in a variety of molecules. In this thesis, primarily protein fold classification problem is employed. For the classification of protein folds, neural network based three methods are used; Grow and Learn (GAL) network, Self-Organizing Maps (SOM) and Self-Organizing Maps for Structured Data (SOM-SD). For GAL and SOM primary protein structures are used, on the other hand for SOM-SD secondary protein structures are used.

Firstly GAL method is used to classify the protein folds. Here, six attributes which are physicochemical features of amino acids (amino acid composition, predicted secondary structure, hydrophobicity, normalized van der Waals volume, polarity and polarizability) are used as features. A number of proteins are selected from Protein Data Bank (PDB). Then, 27-class protein fold classification problem is tried to be solved with this method. To increase the success rate one-versus-others (OvO) prediction method is used. Secondly SOM is used to classify the protein folds. Features and proteins in the previous method are used also in here. As in the previous

method, OvO method is applied for performance evaluation. Thirdly SOM-SD method is used for protein fold classification. While using SOM-SD, Protein Gaussian Image (PGI) representation of proteins is used as feature. PGI is a representation in the Gaussian sphere in which each secondary structure is mapped with a unit vector from the origin of the sphere having the orientation of the secondary structures. The chain sequence of secondary structures is recorded as a list which is mapped on the sphere surface. To test this method the dataset including three folds with 45 proteins (15 proteins in each fold) from PDB is used.

To determine the effectiveness of the attributes some tests were made using GAL. Firstly, only C (amino acid composition) attribute was used to be contained in the feature vectors. Then S (predicted secondary structure) attribute was appended to C, so C+S was used to be the elements of the feature vectors, progressively in the last set all six attributes were used and tested by using GAL. The test results showed that the most important attribute is the amino acid composition. This attribute has a good performance even tested alone.

Besides in here, for reducing dimension of the feature vector without changing success rate divergence analysis was applied. This analysis calculates divergence values of the features and put them in order according to their importance. After this analysis the most significant 30, 40, 50 and 60 features were determined and they were tested with GAL. The results related to protein fold classification problem showed that proteins are classified according to their folds with a good precision and the results are comparable to the existing methods in the literature.

In this thesis after protein fold classification problem, motif retrieval problem is handled. Here, a particular motif is retrieved from a particular protein using structural block comparison. To do this, three methods based on Generalized Hough Transform (GHT) are used. The first method uses single secondary structure, the second one uses secondary structure couple co-occurrences and the third one uses secondary structure triplets. For all three methods the barycenter (geometric mean) of the motif is assigned as Reference Point (RP) and in order to determine this point a mapping rule is figured out. Then, voting process is applied and the point having maximum number of votes is assigned as the candidate RP. For the test, a few proteins selected from PDB are used and the test results showed that the RP is determined with a good precision and the motif is retrieved from the protein with expected number of votes.

PRİMER VE SEKONDER YAPILAR KULLANILARAK PROTEİNLERİN FOLD DÜZEYİNDE SINIFLANDIRILMASI VE MOTİF ÇIKARIMI

ÖZET

Proteinler canlı hayatındaki en temel biyolojik birimlerdir ve canlı vücudundaki bütün biyolojik işlevler proteinler tarafından gerçekleştirilir. Moleküler biyoloji ve genetik alanında yapılan çalışmalar sonucunda çoğu hastalığın protein yapısındaki kusur, hasar ve değişiklikten kaynaklandığı ortaya çıkarılmıştır. Proteinlerin fonksiyonları onların yapıları tarafından belirlenmektedir. Bu nedenle proteinlerin yapı analizi, protein yapılarının karşılaştırılması, benzer yapıların belirlenmesi, motif çıkarımı ve proteinlerin sınıflandırılması moleküler biyoloji açısından önemlidir.

Proteinler, amino asitlerin belirli türde, belirli sayıda ve belirli diziliş sırasında karakteristik düz zincirde birbirlerine kovalent bağlanmasıyla oluşmuş polipeptidlerdir. Her proteinin kendisine has özelliklerinin olmasını sağlayan özel amino asit dizilimleri vardır. Protein yapısı, primer, sekonder, tersiyer ve kuaterner olmak üzere dört ayrı şekilde incelenebilir. Tezde proteinin fold düzeyinde sınıflandırılması ve motif çıkarımı olmak üzere iki konu ele alınmıştır. Bu konulara ilişkin çalışmalar yapılırken proteinin primer ve sekonder yapılarına ilişkin öznitelikler kullanılmıştır.

Polipeptidin düzenli katlanmalar yapması sekonder yapıyı oluşturmaktadır. Yaygın olarak iki tip sekonder yapı vardır: α-heliks ve β-tabaka. Proteinler sekonder yapı bileşenlerine göre dört ana gruba ayrılmaktadır; all-α, all-β, α/β, $\alpha+\beta$. SCOP'a göre bu dört ana sınıf kendi içinde foldlara ayrılırlar. Foldlar sekonder yapıların belirli bir düzene göre katlanmalarından meydana gelen üç boyutlu şekillerdir. Proteinlerin fonksiyonları onların yapıları tarafından belirlendiğinden, aynı yapıya sahip proteinlerin belirlenmesi yani proteinlerin fold seviyesinde sınıflandırılması moleküler biyolojinin önemli çalışmalarından biridir. Proteinlerin fold seviyesinde sınıflandırılması problemi sorgulanan proteinin hangi folda ait olduğunu belirlemektir. Tezde öncelikle proteinlerin fold seviyesinde sınıflandırılması problemi ele alınmakta ve sınıflandırma için yapay sinir ağlarının alt modellerinden olan GAL, SOM ve SOM-SD kullanılmaktadır. GAL ve SOM için primer yapılara ilişkin öznitelikler kullanılırken, SOM-SD için sekonder yapılara ilişkin öznitelikler kullanılmıştır.

Proteinlerin fold düzeyinde sınıflandırılması amacı ile kullanılan ilk yöntem büyü ve öğren (GAL) ağıdır. Büyü ve öğren ağında sınıf sınırları en yakın mesafe ölçüsüne göre belirlenmektedir. Giriş vektörü ile ağdaki tüm vektörler arasındaki mesafe hesaplanır. Giriş vektörünün sınıfı, bu vektöre en yakın mesafedeki ağ düğümünün sınıfı olarak belirlenir. Ağın düğüm sayısı eğitim sırasında otomatik olarak belirlenir. Ağın eğitimi öğrenme ve unutma algoritması olarak iki algoritma tarafından gerçekleştirilir. Öğrenme algoritmasında ağa düğüm eklenirken unutma algoritmasında ağın performansını düşürmeyecek olan gereksiz düğümler ağdan çıkarılırlar. Kalan düğümler ile de yeni girişler test edilir. Bu yöntem ile yapılan testlerde eğitim kümesi için 311 test kümesi için 383 olmak üzere protein veri

bankasından (PDB) alınan 694 protein kullanılmıştır. Kullanılan öznitelik vektörü 125 boyutludur ve amino asitlere ilişkin fizyokimyasal özellikleri belirtmektedir. Altı adet öznitelik kullanılmıştır: Amino asit kompozisyonu (20D), tahmin edilen sekonder yapı (21D), hidrofobisite (21D), normalize van der Waals hacmi (21D), polarite (21D) ve polarizebilite (21D). Bu öznitelikler ve GAL ağı kullanılarak 27-sınıflı proteinin fold seviyesinde sınıflandırılması problemi ele alınmıştır.

İkinci olarak proteinlerin sınıflandırılması amacı ile SOM ağı (Kohonen ağı) kullanılmıştır. SOM danışmansız bir yapay sinir ağıdır, yarışmacı öğrenme algoritmasını kullanır. Bu yöntemde ağın nöronları aktif edilmek için aralarında yarışırlar ve sonuçta yalnızca bir nöron yarışı kazanır. Buradaki temel hedef herhangi bir boyuttaki giriş sinyal desenini iki boyutlu bir haritaya adaptif bir şekilde dönüştürmektir. Sonrasında ise ağa giriş olarak verilen sorgu proteinlerinin sınıfını belirlemektir. Bu yöntemde yapılan testlerde önce üç sınıflı problem sonrasında bir önceki yöntemde olduğu gibi 27-sınıflı problem ele alınmış ve aynı veri kümesi kullanılmıştır.

Proteinin fold düzeyinde sınıflandırılması amacıyla kullanılan üçüncü yöntem SOM-SD'dir. SOM'dan kullandığı veri yapısı nedeniyle farklıdır. SOM-SD girişinde veri olarak grafları kullanan bir yapay sinir ağı modelidir. Bu yöntemde veri yapısı olarak proteinlerin PGI gösterimi kullanılmıştır. PGI, EGI'nın (Extended Gaussian Image) proteinler üzerine uygulanmış halidir. Protein içindeki sekonder yapılar Gauss küresi üzerine, başlangıç noktaları küre merkezine bitiş noktaları küre yüzeyine gelecek şekilde yerleştirilirler. Burada küre yüzeyindeki sekonder yapıların bulunduğu noktalar o sekonder yapının oryantasyon bilgisini içermektedir. Sekonder yapıların zincir sırası küre yüzeyine haritalanan bir liste olarak kaydedilir. PGI gösterimi küre yüzeyindeki noktaların sekonder yapıların sırasına uygun olarak birleştirilmiş halidir. Burada proteinlerin içerisindeki sekonder yapı sayıları farklı olduğundan dolayı ağa giriş verilerinin uzunluğu da farklı olacaktır. SOM-SD ile bu problem ortadan kaldırılmakta ve her bir giriş verisinin uzunluğu eşitlenmektedir. Bu metot PDB'den seçilen üç folda ilişkin 45 protein (her foldda 15 protein) üzerinde test edilmiş ve üç sınıflı, proteinlerin fold düzeyinde sınıflandırılması problemi ele alınmıştır.

Tezde ayrıca yukarıda bahsedilen altı öznitelikten hangisinin sınıflandırmada daha baskın olduğunu belirlemek amacıyla testler yapılmıştır. Testler yapılırken daha iyi başarım vermesinden dolayı GAL ağı kullanılmıştır. Öncelikle amino asit kompozisyonu tek başına GAL ağı ile test edilmiş ve sınıflandırma başarımı hesaplanmıştır. Sonrasında amino asit kompozisyonuna ek olarak tahmin edilen sekonder yapı da öznitelik vektörüne eklenmiş ve sınıflandırma başarımı test edilmiştir. Öznitelikler bu şekilde sırasıyla birbirine eklenmiş ve son aşamada altı özniteliğin hepsi kullanılarak sınıflayıcı test edilimiştir. Test sonuçlarına göre proteinlerin sınıflandırılmasında amino asit kompozisyonunun diğerlerine göre daha etkili olduğu ve tek başına bile proteinleri, literatürdekilerle karşılaştırılabilecek düzeyde sınıflandırdığı ortaya çıkarılmıştır.

Tez kapsamında, kullanılan öznitelik vektörünün boyutunu sınıflayıcının performansını değiştirmeden azaltmak amacıyla diverjans analizi kullanılmıştır. Diverjans analizi iki veya daha fazla sınıfın söz konusu olduğu problemlerde kullanılan tüm özniteliklerin arasından istenen sayıda, performansı azaltmayan en iyi özniteliklerin seçilmesi amacıyla uygulanır. Diverjans hesaplamada sınıf içi saçılım ve sınıflar arası saçılım, sınıfları ayırma kriteri olarak kullanılmaktadır. Tezde kullanılan veriye

diverjans analizi uygulanmış ve GAL ağı ile test edilmiştir. Test sonucu, proteinlerin daha az sayıda öznitelik ile başarım değişmeden sınıflandırılabildiğini göstermiştir.

GAL ve SOM ağları kullanılırken sınıflayıcının performansını artırmak amacıyla OvO yöntemi kullanılmıştır. Bu yöntem çok sınıflı problemlerde kullanılan K-sınıflı problemi iki sınıflı probleme indirgeyen bir yöntemdir. Bir sınıf 1. sınıftaki proteinleri içerirken diğer sınıf 1. sınıf dışındaki K-1 sınıfta olan proteinleri içerir. Aynı şekilde bir sınıf 2. sınıftaki proteinleri içerirken diğer sınıf 2. sınıf dışındaki K-1 sınıfta olan proteinlerin tümünü içerir. Bu şekilde K tane 2 sınıflı sınıflandırıcı oluşturulur ve sorgulanan protein K adet sınıflayıcıda ayrı ayrı test edilir.

Proteinin fold düzeyinde sınıflandırılmasına ilişkin yapılan testlerde sınıf başarımı hesaplanırken literatürde çoğunlukla kullanılan duyarlılık hesabı kullanılmıştır. Buna göre yapılan testler 27 sınıflı problem için GAL ve SOM'un literatürde kullanılan yöntemlerle karşılaştırılabileceği sonucunu ortaya çıkarmış ve proteinler yüksek bir başarımla sınıflandırılmıştır.

Proteinler heliks ve tabaka olmak üzere sekonder yapıların belirli bir sırada dizilimlerinden meydana gelmektedir. Bir yapısal motif ise proteinin belirli küçük bir parçası olup daha az sayıda sekonder yapıdan meydana gelmekte ve aynı motifi içeren farklı proteinler benzer işlevler yapabilmektedirler. Tezde proteinlerin fold düzeyinde sınıflandırılmasından farklı olarak, yapısal blokların karşılaştırılması ile motif çıkarımı konusu da ele alınmıştır. Bunun için Genelleştirilmiş Hough dönüşümü tabanlı üç yöntem önerilmektedir. Genelleştirilmiş Hough dönüşümü, genellikle obje tanımada kullanılmakta ve parametre uzayında oylama işlemine dayanmaktadır. Tez kapsamında kullanılan bu yöntemde amaç, motif çıkarımı için motifin referans noktasının belirlenmesidr. Bu yöntemde motife ilişkin bazı özellikler referans tablosuna kaydedilmekte ve oy verilecek nokta veya koordinatlar referans tablosu elemanları vasıtasıyla hesaplanan özel bir haritalama kuralı uygulanarak belirlenmektedir. Genelleştirilmiş Hough dönüşümü bazlı metotlardan birincisi sekonder yapı teklilerini, ikincisi sekonder yapı ikililerini ve üçüncüsü ise sekonder yapı üçlülerini kullanmaktadırlar. Her üç metot için de motifin geometrik ortası referans noktası olarak belirlenmektedir. Burada amaç motif içindeki sekonder yapılar ile protein içindeki sekonder yapıları karşılaştırıp, oluşturulan haritalama kuralına göre oylama prosedürünü uygulamak ve en fazla oy alan noktayı aday referans noktası olarak belirlemektir. Tekli yöntemde motif içerisindeki her bir sekonder yapı için referans noktasının lokasyonu tanımlanırken, ikili ve üçlü yöntemlerde sırasıyla ikili ve üçlü sekonder yapılar için referans noktasının lokasyonu tanımlanmaktadır. Açı ve mesafe değerleri ile oluşturulan bu tanımlama haritalama kuralını oluşturmaktadır. Tekli yöntemde haritalama kuralı protein içerisindeki her bir sekonder yapıya uygulanırken ikili ve üçlü yöntemlerde sırasıyla protein ve motif ikililerinin ve üçlülerinin eşleşmesi durumunda uygulanmaktadır. Yöntemleri test etmek amacıyla PDB'den seçilen 1FNB proteini içinden rasgele dört ve beş sekonder yapıdan oluşan iki motif seçilmiş ve her üç yöntem de test edilmiştir. Diğer bir testte PDB'den altı adet protein seçilmiş ve bunlardan üçü dört sekonder yapıdan oluşan, kalan üçü ise beş sekonder yapıdan oluşan motiflerin çıkarımı amacıyla test edilmiştir. Bu kısımda son olarak PDB'den 20 adet protein seçilmiş, bu proteinler içindeki üç, dört ve beş sekonder yapıdan oluşan olası bütün motifler test edilmiştir. Yapılan tüm bu testler, referans noktasının yüksek bir doğrulukla belirlendiğini ve motifin proteinden beklenen sayıda oy alarak çıkarıldığını göstermiştir.

1. INTRODUCTION

1.1 Purpose of Thesis

In this thesis we focused on protein fold classification and motif retrieval by structural block comparison. The first purpose of the thesis is to classify proteins according to their folds. For fold classification neural network based three methods; Grow and Learn (GAL) networks, Kohonen's Self-Organizing Maps (SOM) and Self-Organizing Maps for Structured Data (SOM-SD) were employed. For GAL and SOM, features related to primary structure, and for SOM-SD, features related to secondary structure (SS) were used; and better results were obtained compared to existing methods.

For protein motif retrieval task, three new algorithms based on Generalized Hough Transform (GHT) were used. Here, a few attributes related to SS were used as features; and experimental results showed the effectiveness of these approaches in motif retrieval.

1.2 Literature Review

Proteins are fundamental biological macromolecules which organize essential parts of living organisms to control all of their vital functionalities. Protein functions depend on protein chemical reactions with their surrounding and other proteins. Also protein functions are determined by its shape and three-dimensional (3D) structure [1]. The information related to all known proteins is deposited in a data bank called Protein Data Bank (PDB) [2]. There are currently 109,661 (at 22/06/2015) 3D protein structures experimentally determined in PDB and this number increases consistently with an increment of about 700 new molecules for month. However there are a lot of similar structures (not identical) in this protein set. So protein structure comparison came into question in computational biology and is an important issue that helps biologists understand various aspects of protein function and evolution. It is commonly believed that the 3D fold has a major effect on the ability of a protein to bind other proteins or

ligands. The similarity analysis of protein structure is therefore an important process in understanding the protein's role in the machinery of life. Comparison of protein structures is also essential for estimating evolutionary distance between proteins and protein families [3].

In proteins, a structural motif is a 3D structural element which appears in a variety of molecules and usually consists of just a few elements. Several motifs packed together to form compact, local, semi-independent units are called domains. The size of individual structural domains varies from about 25 up to 500 amino acids, but majority, 90%, has less than 200 residues with an average of approximately 100 residues. The term family as it is used in taxonomy should not be confused with protein family which is a group of evolutionarily related proteins, that is, proteins in a protein family descend from a common ancestor and typically have similar 3D structures, functions and significant sequence. Note that it is also often used the term super-*, where * can stand for motif, or domain, or family, or fold, or class [4]. There are several methods for defining protein SSs; Dictionary of Secondary Structure of Proteins (DSSP) and Structural Identification (STRIDE) [5,6] are the most commonly used. The DSSP defines eight types of SSs, nevertheless, the majority of secondary prediction methods simplify further to the three dominant components; helix, strand and coil. The structural analysis for protein recognition and comparison is conducted mainly on the basis of the two most frequent components; the helices and the strands [7]. Structural Classification of Proteins (SCOP) [8] provides a detailed and comprehensive description of the structural and evolutionary relationships among all proteins whose structures are known [9]. According to convention a protein could be classified into one of four structural classes based on its SS components; all-α, all-β, α/β and $\alpha+\beta$; and according to the SCOP four structural classes are divided into folds, folds are divided into superfamilies, and superfamilies are divided into families. Folds represent the 3D shape of proteins and because of that the protein structure define the protein function. Hence, classifying the proteins according to their folds is an important issue for structural biology.

In the literature there are several types of works about proteins. The basic works are about protein structure comparison [4, 10–16] prediction of protein SSs [17–24], prediction of protein structural classes [25–34] and classification of protein

folds [1, 9, 35–74]. In this thesis we focused on protein structure comparison and fold classification. Related to protein structure comparison, these studies have been done: Can et al. [10] presents a new method for conducting protein structure similarity searches and applies differential geometry knowledge on protein 3D structure for extracting signatures such as curvature, torsion and SS type. Camoglu et al. [11], in order to find similarities in a protein structure database, builds an indexing structure based on SS elements triplets by using R-tree. Chionh et al. [12] propose the SCALE algorithm to compare protein 3D structures based on angle-distance matrices that utilizes angles and distances between SS elements. Chi et al. [13] design a fast system for protein structure retrieval by using image based distance matrices and a multidimensional index. Zotenko et al. [14] propose an approach to speed up protein structure comparison by mapping a protein structure to a high-dimensional vector and approximating structural similarity by distance between the corresponding vectors. Krissinel et al. [15] describe the Secondary Structure Matching (SSM) algorithm of protein structure comparison in 3D, which includes an original procedure of matching graphs built on the protein's SS elements, followed by an iterative 3D alignment of protein backbone C_α atoms. Cantoni et al. [4,16] made a study for retrieving structural motifs by using Generalized Hough Transform (GHT) and range tree. They also retrieved the Greek Key motif, which is formed by four SSs, from the protein files by using the GHT. In a part of the thesis, we propose three new methods based on GHT for protein motif retrieval.

Protein fold classification is the prediction of protein's tertiary structure (fold) given the protein's sequence without relying on sequence similarity. In the past three decades, many efforts have been made to classify the proteins according to their folds. A variety of classification methods have been applied to this task such as neural networks [9,35–38,43,47,60], Bayesian classifiers [46], K-nearest neighbor [44,50,66,67], support vector machine (SVM) [40,45,52] and ensemble classifiers [1,39,48,49,51,54,58,68,73,74].

One of the early studies about fold classification was done by Reczko and Bohr [35]. They used a special type of feed-forward neural networks called Cascade-Correlators. In their study, the training algorithm optimizes the weights and the number of hidden units in a feed-forward network by adding units during the training process. The

process of adding new hidden units that maximize the correlation between their activity and the error remaining at the output layer is repeated until the mapping has the desired accuracy. Their predictive performance turned out to be rather successful with a score of around 82% for predicting fold classes (with a total of 42 classes).

In 1999, Dubchak et al. [9] developed a neural network based computational method for the assignment of a protein sequence to a folding class in the SCOP. In that study three layer feed-forward neural networks were used with the neural network weights adjusted by conjugate gradient minimization. This method used global descriptors of a primary protein sequence in terms of the physical, chemical and structural properties of the constituent amino acids and performed well for protein fold prediction.

In 2001, Edler et al. [36] made a study to show the role and results of statistical methods in the prediction of protein fold classes. They used feed-forward neural networks and standard statistical classification procedures to classify proteins. They applied logistic regression, additive models, and projection pursuit regression from the family of methods based on posterior probabilities; linear, quadratic, and a flexible discriminant analysis from the class of methods based on class conditional probabilities, and the nearest neighbor classification rule to a dataset of 268 proteins (including 42 folds). In the same year, Ding and Dubchak [37] used SVMs and neural networks learning methods as base classifiers to recognize protein folds. They used the database, including 27 protein folds, in their earlier study [9]. These folds include 311 proteins for training set and 383 proteins for test set, totally 694 proteins. To increase the success rate they applied one-versus-others (OvO), unique one-versus-others (uOvO) and all-versus-all (AvA) prediction methods. They classified the protein folds with a 56% success rate using SVM with AvA prediction method.

Motivated by Ding and Dubchak, Bologna and Appel used an ensemble of four-layer Discretized Interpretable Multi Layer Perceptron (DIMLP) [38] trained with the dataset produced by Ding and Dubchak [37]. In addition, they added the length of the amino acid sequence as an effective feature to each feature group. Different from Ding and Dubchak's study, in [37], each classifier learned all the folds simultaneously. They classified the proteins with a 61.1% success rate.

In 2003, Bindewald et al. presented a hybrid of different approaches as protein fold recognition method called MANIFOLD (MANheIm FOLD recognition) [39]. They used sequence similarity, SS and functional information to improve fold recognition. They tested this method on the dataset provided by Ding and Dubchak [37] and they obtained a prediction accuracy of 74.9%. Markowetz et al. [40] compared the performance of SVM to neural networks and to standard statistical classification methods as Discriminant Analysis and Nearest Neighbor classification. They used the dataset, in [36] including 268 proteins (42 folds) and they obtained the best error rate 23.2%. Tan et al. [41] applied ensemble learning algorithm to the prediction of fold classes. Huang et al. [42] proposed a hierarchical learning architecture (HLA) method that classified proteins into four major classes; all-α, all-β, α+β, α/β. Then in the next level they used another set of classifiers (MLP, GRNN, RBFN, SVM) to further classify the proteins into 27 folds. The best accuracy of HLA method based on RBFN is up to 65.5% on the Ding and Dubchak's dataset.

In 2004, Igel et al. [43] applied standard feed-forward neural networks to assign primary sequences of proteins to one out of 42 fold classes. They used early-stopping for implicit regularization to improve the generalization properties of the neural networks. For comparison of their neural network approach with the results in the literature, they used the same data and basically the same performance measures as in [36, 40] and they obtained the best error rate 22.4%. To classify the proteins according to their folds, Okun used a modified nearest neighbor algorithm called the K-Local Hyperplane Distance Nearest Neighbor (HKNN) [44]. He used the dataset in [37] and classified the proteins with a success rate 57.4%. Shi et al. [45] proposed a multi-objective feature analysis algorithm. The objective of this algorithm is simultaneously selecting the effective features, improving the accuracy and providing bias information of train and test data. To achieve this objective, authors used an extended wrapper method for feature selection and processed SVM for classification task. They used the same dataset used in [37] and achieved 61.6% classification performance.

In 2005, Chinnasamy et al. [46] presented a framework using Tree-Augmented Networks (TAN) based on the theory of learning Bayesian networks but with less restrictive assumptions than the naive Bayesian networks. In order to enhance the

TAN's performance, they did pre-processing of data with feature discretization and post-processing by using Mean Probability Voting (MPV) scheme. They used the datasets in [36] and [37], and they obtained classification performances of 58.9% and 74.4% respectively. Huang et al. [47] used neural network based hierarchical learning architecture to deal with fold classification problem. In this study, the network can not only select features in an online manner during learning, but it also does some feature extraction. They combined the feature selection with their hierarchical learning architecture and applied it multi-class protein fold classification and they had a result in a test accuracy of 56.4%.

In 2006, Nanni made two studies related to protein fold classification [48, 49]. In the first study he proposed a new ensemble of K-local hyperplanes based on random subspace and feature selection. In the second, he developed an ensemble classifier by combining Fisher's linear classifier and HKNN [44]. He tested these methods on the dataset including 27 folds and he obtained 61.1% and 60.3% classification performance respectively. Shen and Chou developed an ensemble classifier for fold pattern recognition [50]. In this study, the operation engine for the constituent individual classifiers was OET-KNN (optimized evidence-theoretic K-nearest neighbors) rule. Their outcomes were combined through a weighted voting to give a final determination for classifying a query protein. The method was tested on the dataset in [37] and the proteins were classified with 62.1% success rate.

In 2007, Chen and Kurgan proposed PFRES method for automated protein fold classification [51]. This method combines evolutionary information by using the PSI-BLAST profile-based composition vector and information extracted from the SS predicted with PSI-PRED. In this study, they used a voting-based ensemble classifier and obtained 68.4% prediction accuracy. Shamim et al. [52] developed a SVM-based classifier that uses secondary structural state and solvent accessibility state frequencies of amino acids and amino acid pairs as feature vector. They performed their method on the dataset in [37] and they obtained 70.5% performance accuracy by using OvO prediction method. In 2008, Abual-Rub and Abdullah [53] reviewed different algorithms related to protein fold classification and they concluded that the evolutionary algorithms can be used for this task.

In 2008, motivated by [50], Guo and Gao proposed a novel hierarchical ensemble classifier named GAOEC (Genetic-Algorithm Optimized Ensemble Classifier) [54] and they overcame the shortcomings in [50]. To construct the classifier, firstly, a novel optimized classifier named GAET-KNN (Genetic-Algorithm Evidence-Theoretic K Nearest Neighbor) was proposed as a component classifier. Secondly, six component classifiers in the first layer were used to get a potential class index for every query protein. Thirdly, according to the results of the first layer, every component classifier in the second layer generated a 27-dimensional vector whose elements represented the confidence degrees of 27 folds. Finally, genetic algorithm was used for generating weights for the outputs of the second layer to get the final classification result. Guo and Gao tested this classifier on the dataset in [37] and they had a 64.7% prediction accuracy. A different study was done by Krishnaraj and Reddy in 2008 [55]. They used two variants of Boosting algorithms (AdaBoost and LogitBoost) for the problem of fold recognition. Prediction accuracy was measured on the dataset including the most famous 27 SCOP folds, and AdaBoost and LogitBoost achieved 57.7% and 60.13% fold recognition accuracy, respectively. Damoulas and Girolami [56] proposed a multi-class multi-kernel learning method to recognize protein folds. It applied a single multi-class kernel machine on all of the characteristic spaces simultaneously and then combined their results. The method achieved the best accuracy of 70% on the benchmark dataset proposed by Ding and Dubchak [37].

In 2009, Shen and Chou [57] proposed a new approach which is featured by combining the functional domain information and the sequential evolution information through a fusion ensemble classifier, and they called it FPF-FunDEsqE. Tests were performed for identifying proteins among their 27 fold patterns and the classifier achieved 70.5% prediction accuracy. Hashemi et al. [1] applied two classification methods; MLP and RBF networks. Also they used Bayesian and Majority Voting classifier ensemble methods to improve the prediction results of the base classifiers. In their study the MLP network had only one hidden layer with tangent sigmoid as activation function. For recognizing the exact class of a protein they used the label of the maximum output unit in the network as the protein class label. They used Correct Classification Rate (CCR) as the evaluation measure which is the number of correct classified instances over the total number of instances. In this study final CCR became around 59%. Chen et al. [58]

developed a novel approach based on genetic algorithms and a SVM to determine the best feature selector. The SVM applied the best feature selector to the feature vectors in the test dataset to classify the protein folds. This method achieved an accuracy of 71.28% for protein fold classification problem. To predict protein folds, Ghanty and Pal [59] proposed several new features and used some existing features including frequencies of adjacent residues, frequencies of residues separated by one residue, and triplets of amino acid compositions; and they used MLP network, RBF network and SVM as machine learning tools. To improve the recognition accuracies further, they used fusion of different classifiers and their system achieved 68.6% test accuracy for the fold recognition with 27 folds. Jazebi et al. [60] employed a fusion method (using weighted voted approach and OWA operators) for fold pattern recognition. They used the Probabilistic Neural Network (PNN) as base classifier in the fusion method. Tests were performed on the dataset in [37] and 52.3% classification performance was obtained.

In 2010, Dehzangi et al. [61] used Random Forest, which is a recently introduced method based on bagging algorithm that trains a group of base classifiers by randomly selecting sets of features then, combining results obtained from base classifiers by majority voting. In this study, to investigate the effectiveness of the number of base learners on the performance of the Random Forest, twelve different number of base classifiers (between 30 and 600) were applied for the classifier and they achieved 62.7% prediction accuracy. They also used Rotation Forest [62], a straightforward extension of bagging algorithms which aims to promote diversity within the ensemble through feature extraction by using Principal Component Analysis (PCA). Valavanis et al. [63] used five different classification techniques, namely MLP, PNN, nearest neighbor classifier, multi-class SVM and Classification Trees (CTs) for protein fold recognition problem on the dataset including 27 folds, and polynomial SVM achieved 42.8% classification performance. Wang and Gao [64] developed a two-layer learning architecture, named TLLA, for multi-class protein fold classification problem. In the first layer, OET-KNN was used as the component classifier to find the most probable K-folds of the query protein. In the second layer, SVM was used to build the multi-class classifier just on the K-folds. The dataset in [37] was used to evaluate the performance of the classifier and the method achieved 63% success rate.

Motivated by [55, 61, 62] in 2011, Dehzangi and Karamizadeh used a fusion of heterogeneous Meta classifiers, namely LogitBoost, Random Forest and Rotation Forest [65]. They used Ding's feature set and achieved 65.3% protein fold prediction accuracy. Kavousi et al. extracted ten different features from protein sequences and used ten OET-KNN classifiers as the classification engine [66] and in 2012 they dealt with protein fold classification based on the concepts of hyperfold [67]. Tests were performed on the dataset in [37] and they achieved 67.2% and 73.1% classification performances in 2011 and 2012, respectively. Yang et al. [68] proposed a novel margin-based ensemble classifier, called MarFold, for multi-class protein fold recognition task where multiple heterogeneous feature space were available. They built this method on three component classifiers, namely, adaptive local hyperplane (ALH), SVM and ALHK (a variant of ALH). To evaluate the proposed method they used the dataset established by Ding and Dubchak [37] and they obtained 71.7% overall prediction accuracy by MarFold.

In 2012 Suvarnavani et al. [69] applied boosting algorithm (SMOTE) to rebalance the imbalanced dataset to boost the performance and then they used a decision tree classifier to classify folds from the features of contact map. They obtained over 70% accuracy for the feature set generated by Triangle Sub division method. Another study was made by Chmielnicki and Stapor [70]. They suggested a hybrid discriminative/generative approach. Accordingly, they combined the well-known SVM classifier with regularized discriminant analysis (RDA). In this method, SVM classified the proteins using the results of RDA. The dataset including 27 folds was used and 77.9% classification performance was achieved.

In 2013, Bae et al. [71] made a study to deal with fold classification problem. In this study, prediction of structural fold classes of proteins with torsion angle based SS profile library and multi-class linear discriminant analysis, was performed. In one of the recent studies, Suryanto et al. [72] proposed a new approach for protein fold classification by introducing the concept of large margin nearest neighbor for combining multiple measures of distance between protein structures. They combined the Euclidean distance matrices of 12 features extracted from the amino acid sequence of the protein, the root mean square deviation (RMSD) obtained from the geometrical alignment using Combinatorial Extension, and the canonical angles between the

subspaces generated from the synthesized multi-view protein structure images. To demonstrate the effectiveness of the proposed method they used the dataset produced by Ding and Dubchak and they achieved 92.4% prediction accuracy. Another up-to-date study was made by Lin et al. [73] in 2013. They utilized a K-means clustering algorithm to choose a series of different base classifiers and a circulating, combined static selective strategy. Tests were performed on the dataset in [37] and 74.2% accuracy rate was obtained.

In 2015, Aram et al. [74] used a Two-Layer Classification Framework (TLCF) and a fusion of MLP, RBFN and Rotation Forest. In the first layer they classified the proteins according to their structural classes, and in the second layer according to their folds. The fusion method was performed on the dataset in [37] and 65.7% prediction accuracy was obtained.

In this thesis we dealt with two problems: Protein fold classification and motif retrieval by structural block comparison. For the first problem, we used GAL networks, SOM and SOM-SD approaches to solve protein fold classification problem. While GAL and SOM use primary protein structures, SOM-SD uses secondary protein structures. Neural network models are widely used in literature for protein fold classification problem, but to the best of our knowledge, GAL networks, SOM and SOM-SD have not been used for this task. For the second problem, motif retrieval, we proposed three new methods based on GHT. These three methods use single SS, SS couple co-occurrences and SS triplet co-occurrences as primitive aggregates, and they use the attributes related to secondary protein structures for motif retrieval.

1.3 Thesis Organization

The aim of this thesis is to retrieve a motif from the protein with structural block comparison by using GHT based methods and to classify the proteins according to their folds using GAL, SOM and SOM-SD networks.

Before going into details of the algorithms for motif retrieval and fold classification, bioinformatics and basic concepts about the protein structure and function are introduced in Chapter 2.

Chapter 3 discusses feature extraction from primary and secondary protein structures and feature reduction with divergence analysis.

The methods used for fold classification and motif retrieval are explained in Chapter 4. For fold classification GAL, SOM and SOM-SD networks; for motif retrieval GHT-based approaches are told in Section 4.1, 4.2, 4.3 and 4.4, respectively.

In Chapter 5 experiments and results are explained and Chapter 6 concludes the thesis.

2. PROTEIN STRUCTURES

Protein fold classification, structural block comparison and motif retrieval examined in the thesis scope are studies related to proteomics (protein recognition, protein classification) and are included by bioinformatics study areas. Being a subclass of Computational Biology, bioinformatics is a scientific discipline dedicated to solve biological problems at the molecular level with computer methods. It is an attempt to describe biological phenomena in terms of numerical and statistical methods. Historically biology has used less mathematical approaches than other scientific disciplines such as physics and chemistry. Bioinformatics then, tried to address this gap by providing the typical results of biochemistry and molecular biology a kit of analytical and numerical tools, involving computer science, applied mathematics, statistics, chemistry and artificial intelligence concepts [75].

Proteomics is a subclass of Bioinformatics and deal with large scale study of proteins especially protein structures and functions. The basic study areas of proteomics are like that:

- predicting protein structure from their sequence
- structural classifications of proteins
- protein fold classification
- protein motif retrieval
- predicting protein-ligand interactions
- aligning proteins' sequences
- defining criteria of similarity between proteins
- protein structure comparison

Within this context, there are a number of approaches for each one of the mentioned study areas. For the scope of this thesis, we will concentrate our attention on protein motif retrieval, protein structure comparison and protein fold classification.

2.1 Proteins

Proteins can be found in all living systems ranging from bacteria and viruses through the unicellular and simple eukaryotes to vertebrates and higher mammals such as humans. Proteins are present in larger quantities than any other biomolecule and make up over 50 percent of the dry weight of cells. Proteins are also very important amongst other macromolecules because they are the basis for any reaction that occurs in living systems [76].

2.1.1 Biological functions of proteins

Proteins provide a wide range of different biological features that range from DNA replication to molecules transformation. The possible functions of proteins are large enough and today we still learn new ones as long as knowledge on protein increases. Below is a list of several types of proteins and their functions inside a body [75]:

Antibodies are specialized proteins commonly found in blood or other bodily fluids of vertebrates. They are used by the immune system to identify and neutralize antigens (foreign invaders like virus or bacteria). One way antibodies destroy antigens is by immobilizing them so that they can be destroyed by white blood cells.

Contractile proteins participate in contractile processes. They are not only involved in muscle contraction and movement, but they also participate in localized events in the cytoplasm or general cell aggregation phenomena.

Enzymes are proteins that facilitate and speed up biochemical reactions. They are often referred as catalysts. Examples include the enzymes lactase and pepsin. Lactase breaks down the sugar lactose found in milk and is essential for digestive hydrolysis of lactose milk. Pepsin works in the stomach and is fundamental for the process of digestion to break down proteins in food.

Hormonal proteins are messenger chemically released by a cell in one part of the body that sends out messages that affects cells in other part of the organism. This

function helps to coordinate certain bodily activities. An example is insulin, that helps to regulate glucose metabolism by controlling the blood-sugar concentration.

Structural proteins are fibrous and provide both external protection and internal connective support for the body. For example keratin forms protective covering of land vertebrates: skin, fur, hair, wool, nails, horns, beaks and feathers. Another example is elastin that provides support for connective tissues such as tendons, hides and ligaments.

Storage proteins store amino acids and metal ions used by organisms. Seeds, particularly of leguminous plants, contain high concentrations of storage proteins. Examples of amino acid storage proteins in animals include casein and ovalbumin.

Transport proteins are carriers of molecules from one place to another inside the body. The most known examples are haemoglobin, which carries oxygen from the lungs to the tissue, and myoglobin, which takes oxygen from the haemoglobin in the blood and carries it around until needed by the muscle cells.

A placement of each protein in a formal class is not correct because each protein can have more than one function at a time; because of that many proteins are not easy to classify. However, all proteins are characterized by the same basic structure made of 20 types of amino acids. What really differentiates them is the composition; not all have the same amount of each amino acid and some may even lack one or two members of the group of 20 entirely. It was realized early in the study of proteins that variation in size and complexity is common and the molecular weight and number of subunits (polypeptide chains) show tremendous diversity [76]. There is no correlation between the size of the protein and the number of polypeptide chains. For example haemoglobin has four subunits and a molecular mass of 64500 while insulin has two subunits but only a molecular mass of 5700 [75].

2.1.2 Protein structure

Proteins are formed by different combinations of amino acids and peptide bonds connecting two amino acids; and they are defined by different levels of protein structures (primary, secondary, tertiary and quaternary structures). In the next three

sub sections amino acids, peptide bonds and different levels of protein structures will be introduced briefly.

2.1.2.1 Amino acids

All proteins are made of a linear composition of amino acid residues. This assemblage is called *polypeptide chain*. Amino acids are the chemical units or building blocks of the body that make up the proteins. Twenty different amino acids are used to synthesize proteins. The shape and other properties of each protein are dictated by the precise sequence of amino acids in it.

Each amino acid consists of an alpha carbon atom to which the structures in below are connected (see Figure 2.1):

- a hydrogen atom;
- an amino group NH_3^+;
- a carboxyl group COO^-. This gives up a proton and is thus an acid;
- one of 20 different "R" groups (side chain). It is the structure of the R group that determinates univocally each amino acid and also its special properties.

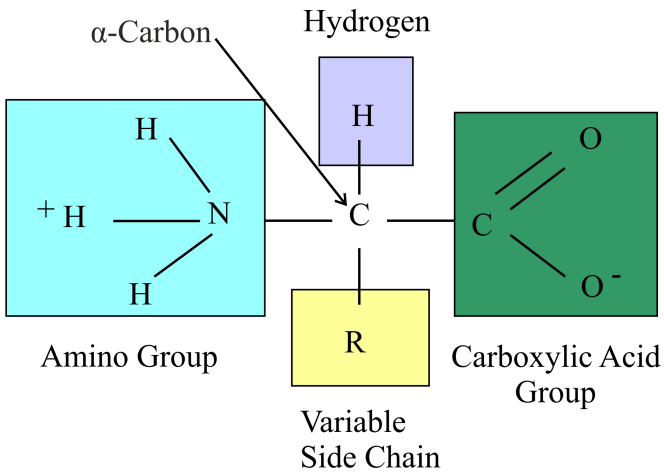


Figure 2.1 : The basic amino acid structure.

All amino acids contain carbon, hydrogen, nitrogen and oxygen with two of the 20 amino acids also containing sulfur (Table 2.1 summarizes each amino acid composition). At pH7 the amino and carboxyl groups are charged, but at different values of pH from 1 to 14 these groups exhibit different behaviors involving binding

Table 2.1 : Amino acid names, three and one-letter standard abbreviations and linear structures.

Name	Abbreviation	Linear Structure
Alanine	ala A	CH3-CH(NH2)-COOH
Arginine	arg R	HN=C(NH2)-NH-(CH2)3-CH(NH2)-COOH
Asparigine	asn N	H2N-CO-CH2-CH(NH2)-COOH
Aspartic Acid	asp D	HOOC-CH2-CH(NH2)-COOH
Cysteine	cys C	HS-CH2-CH(NH2)-COOH
Glutamic Acid	glu E	HOOC-(CH2)2-CH(NH2)-COOH
Glutamine	gln Q	H2N-CO-(CH2)2-CH(NH2)-COOH
Glycine	gly G	NH2-CH2-COOH
Histidine	his H	NH-CH=N-CH=C-CH2-CH(NH2)-COOH
Isoleucine	ile I	CH3-CH2-CH(CH3)-CH(NH2)-COOH
Leucine	leu L	(CH3)2-CH-CH2-CH(NH2)-COOH
Lysine	lys K	H2N-(CH2)4-CH(NH2)-COOH
Methionine	met M	CH3-S-(CH2)2-CH(NH2)-COOH
Phenylalanine	phe F	Ph-CH2-CH(NH2)-COOH
Proline	pro P	NH-(CH2)3-CH-COOH
Serine	ser S	HO-CH2-CH(NH2)-COOH
Threonine	thr T	CH3-CH(OH)-CH(NH2)-COOH
Tryptophan	trp W	Ph-NH-CH=C-CH2-CH(NH2)-COOH
Tyrosine	tyr Y	HO-Ph-CH2-CH(NH2)-COOH
Valine	val V	(CH3)-CH-CH(NH2)-COOH

and dissociation of a proton. This behavior distinguishes the groups as weak acids or weak bases. The acid-base behavior in particular is very important since it affects the potential properties of the protein and their reactivity. As shown in Figures 2.1 and 2.2 all the components of amino acids (amino, carboxyl, hydrogen and R group) are arranged tetrahedrally around the central alpha carbon.

Each amino acid has a characteristic side chain, or R group that imparts chemical individuality to the molecule. These R side chains contain different structural features, such as aromatic rings, -OH groups, -NH3+ groups, COO- groups, sulfur containing

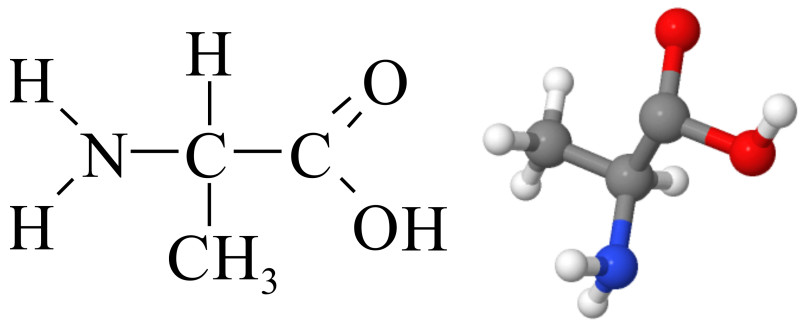


Figure 2.2 : Alanine composition and its 3D spatial arrangement generated with Jmol [78].

residues etc. This variety in side chains causes difference in the properties of the individual amino acids and the proteins containing different combinations of them.

2.1.2.2 Peptide bonds

A peptide bond is a covalent bond that is formed between two molecules when the carboxyl group of one molecule reacts with the amino group of another molecule, releasing a molecule of water [77]. This is how amino acids are joined together. The resulting CO-NH bond is called a peptide bond, and the resulting molecule is an amide (see Figure 2.3).

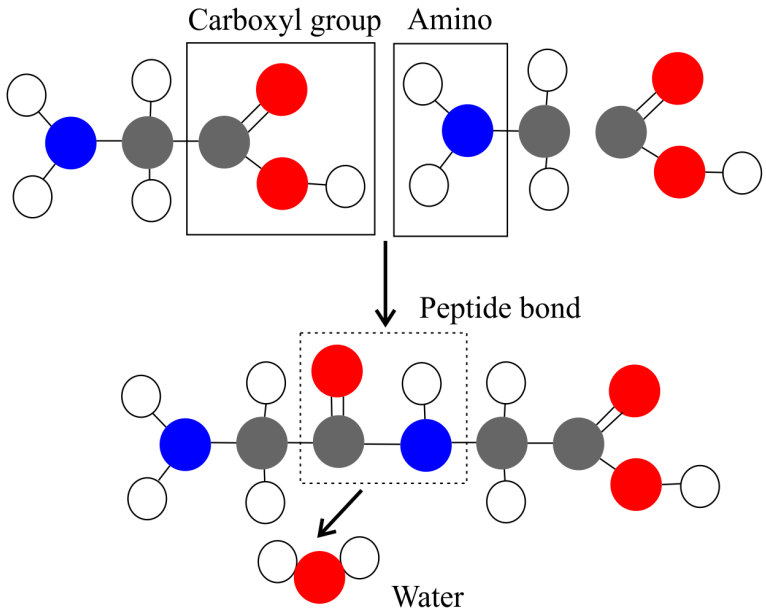


Figure 2.3 : Two amino acids join together by the carboxyl group of one and the amino group of the second, and a molecule water is removed in this process. Here, grey shows the carbon, blue nitrogen, red oxygen and white hydrogen.

The molecules must be orientated so that the carboxylic acid group of one can react with the amino group of the other.

Any number of amino acids can be joined together in chains of 50 amino acids called peptides, 50-100 amino acids called polypeptides, and over 100 amino acids called proteins. A number of hormones, antibiotics, antihumor agents and neurotransmitters are peptides (proteins).

A peptide bond can be broken down by hydrolysis (addition of water). The peptide bonds that are formed within proteins have a tendency to break spontaneously when subjected to the presence of water releasing about 10 kJ/mol of free energy.

Amino acid sequences are read from left to right (from amino to carboxyl terminal). In order to save space in long protein sequences, only a three letter code (or a single letter code) is used for each amino acid (see Table 2.1).

The protein sequence can be divided into main chain and side chain components. The main chain is the same for any protein and differs only in extensions. The main chain includes all the residues found in the polypeptide chain. This backbone represents repetition of peptide bonds made up of the N, C_α and C atoms. On the contrary the side chain represents different components in each protein [75].

2.1.2.3 Primary structure

The primary structure of a protein is the linear order of amino acid residues along the polypeptide chain [76]. It is the amino acid sequence between terminals expressed with three of single letter codes. Here is an example of primary structure:

NH3-Ala-Glu-Glu-Ser-Ser-Lys-Ala-Val-Lys-Tyr-Tyr-Thr-...

NH3-A-E-E-S-S-K-A-V-K-Y-Y-T-...

Each protein is defined by a unique sequence of residues. All other representations are based on this primary level of structure. It is important to realize that two proteins that contain the same amino acid composition can have very different primary structures. The following two peptides, for example, are formed by the same amino acids, but their primary structures are different, since the sequences are different:

H2N-Glu-Ala-Val-Ser-Leu-Ala-Lys-Cys-COOH

H2N-Ala-Glu-Val-Ser-Ala-Leu-Lys-Cys-COOH

The primary structure determines the three-dimensional structure of the protein, which in turns determines its biological function. Alteration in normal primary structure of proteins can produce catastrophic results [75].

2.1.2.4 Secondary structure

Primary structure leads to SS that takes into consideration the local confirmation of the polypeptide chain or the spatial relationship between amino acids' residues that are close together in the primary sequences. The three basic units of the SS are α-helix, β-strand and turns. It is possible to have other structures but they are considered variations of the three previous.

α-helix: The most common structural pattern which can be found in a protein is the right-handed α-helix. A regular α-helix has 3.6 residues per turn with an offset between residues of 0.15 nm. The pitch of the helix is therefore 0.54 nm (3.6×0.15) that is the translation distance between two corresponding atoms on the helix. The α-helix model has two important angles with regular values that are called ϕ and ψ, torsion or dihedral angles. The particular arrangement of the α-helix allows some of the backbone atoms to form hydrogen bonds between the backbone carboxyl oxygen (acceptor) of one residue and the amide hydrogen (donor) of a residue four ahead in the polypeptide chain. α-helices are represented with a wireframe, spacefilling, ball-and-stick and usually cartoon representation (see Figures 2.4, 2.5, 2.6 and 2.7).

β-strand: The β-strand is the second unit of the SS. Although it is displayed as an arrow in the SS its actual conformation is very similar to an extremely elongated helix (see Figure 2.4). Regular β-strand has only two residues per turn and a pitch of 0.7 nm. Since a single β-strand is not stable because of the limited number of local interactions, the majority of β-strands are arranged adjacent to other strands and form an extensive hydrogen bond network with their neighbors in which the N-H groups in the backbone of one strand establish hydrogen bonds with the C=O groups in the backbone of the adjacent strands.

Adjacent strands can align in parallel or anti-parallel arrangements. The arrow in the cartoon representation of proteins shows the direction towards the C terminal. It is rare to find less than five interacting parallel strands in a motif, suggesting that a smaller number of strands may be unstable.

Figure 2.4 : Cartoon representation of the 1FNB (PDB ID) protein. Helices are represented as pink and purple ribbons, strands are represented as yellow arrow. Image generated with Jmol [78].

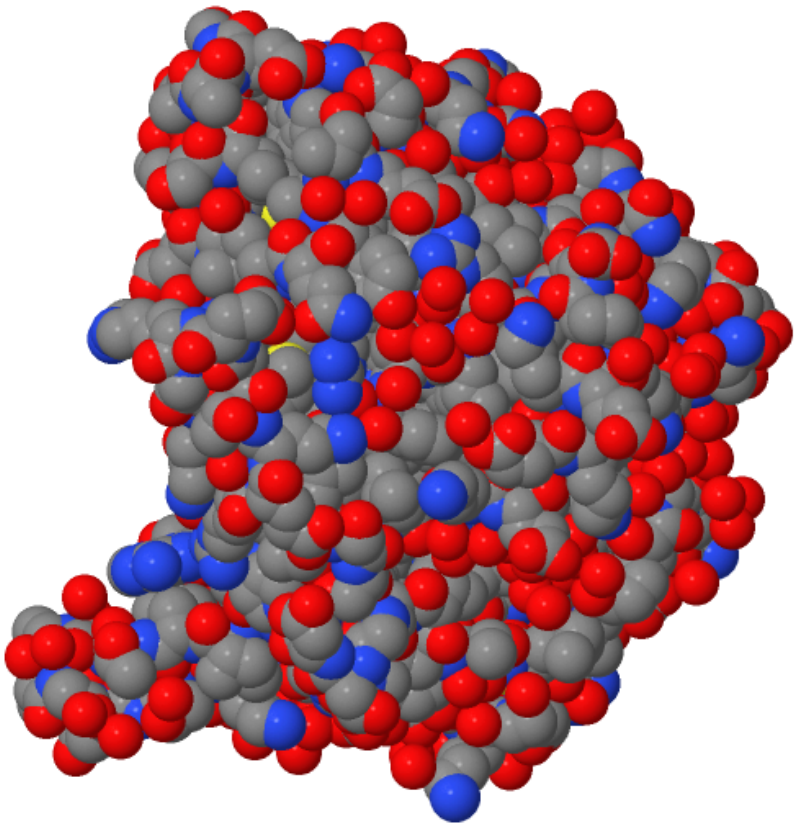


Figure 2.5 : Spacefilling representation of the 1FNB (PDB ID) protein. Image generated with Jmol [78].

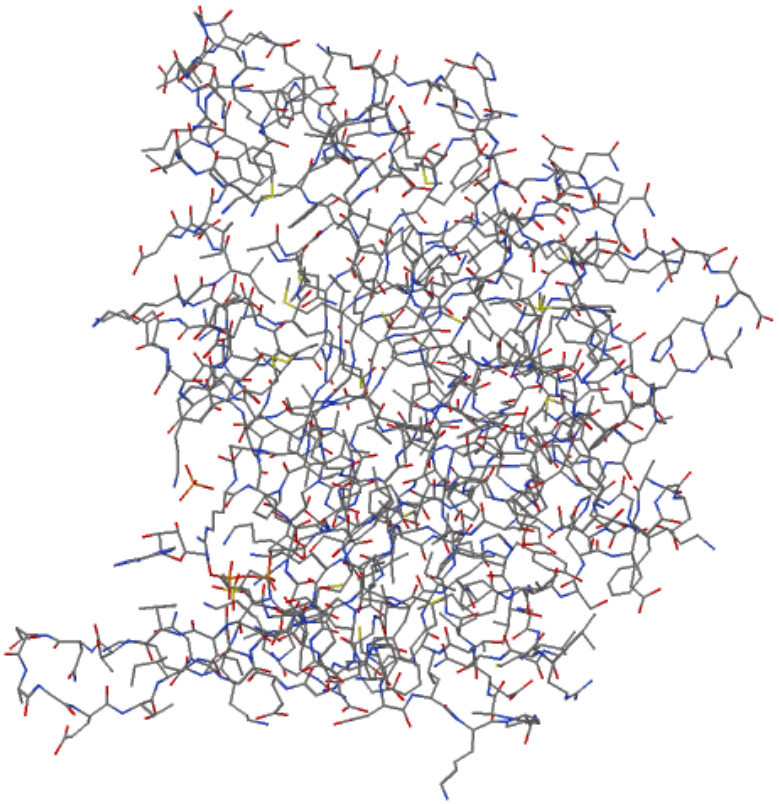


Figure 2.6 : Wireframe representation of the 1FNB (PDB ID) protein. Image generated with Jmol [78].

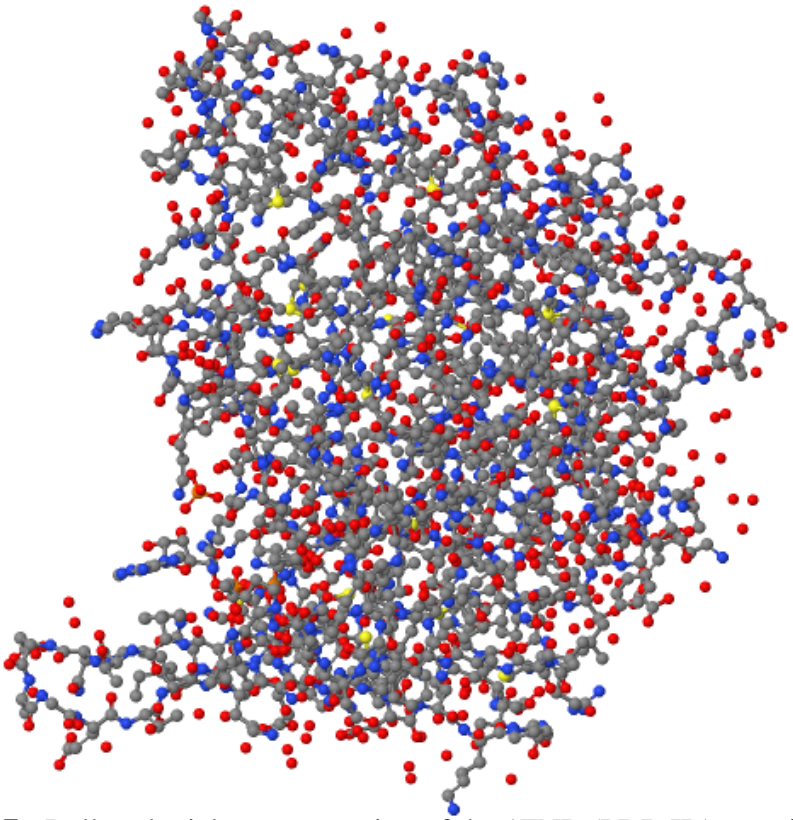


Figure 2.7 : Ball-and-stick representation of the 1FNB (PDB ID) protein. Image generated with Jmol [78].

A stable block of β-strands is usually called β-sheet and consists of some β-strands connected laterally by at least two or three backbone hydrogen bonds, forming a generally twisted, pleated sheet.

The most known and spectacular arrangement of β-strands is called β-barrel. The β-barrel is a large β-sheet that twists and coils to form a closed structure in which the first strand is hydrogen bonded to the last. β-strands in β-barrels are typically arranged in an anti-parallel fashion.

Turns: Turns have the universal role of enabling the polypeptide to change direction and in some cases to reverse back on itself. The reverse turns or bends arise from the geometric properties associated with these elements of protein structure.

Several definable turns and bends in protein structure have been recognized and classified either by the relationship between the ϕ, ψ angles of the residues in the turn or the hydrogen bonding of their amide N-H and carbonyl-oxygen atoms. The tightest turns involve only three residues with hydrogen bonding between the carbonyl of the first residue (N-terminal end) and the N-H of the third residue. These turns are referred to as γ turns. Turns involving four residues are more common with hydrogen bonding from the carbonyl of residue 1 to the N-H of residue 4. One class of these turns is called β-turns, typically found at turns of β-sheet structure [75].

2.1.2.5 Tertiary structure

The description of the complex and irregular folding of the peptide chain in three dimensions is called tertiary structure. It is essentially a picture of what the shape of the entire protein actually looks like. The functionality of the protein derive strictly from its conformation. If something changes the three dimensional conformation the activity of the protein will be lost. These complex structures are held together by the R-groups of each amino acid in the chain that generate a combination of several molecular interactions. In short, R-groups determine the structure of regions of peptide chains that do not form a regular SS [75].

2.1.2.6 Quaternary structure

The quaternary structure of a protein describes the interactions between different polypeptide chains that make up the protein [79]. Some proteins (such as hemoglobin)

have more than one peptide chain (these are called multimeric or oligomeric proteins). Oligomeric proteins can be composed of multiple identical polypeptide chains. Proteins with identical subunits are termed homo-oligomers. Proteins containing several distinct polypeptide chains are termed hetero-oligomers. The manner in which these chains fit together is the quaternary structure. Obviously, if a protein is made up of only one chain (monomeric), there is no quaternary structure for that protein. The forces that hold different chains together are the same that hold the tertiary structure together. Figure 2.8 shows the structure of hemoglobin, the oxygen carrying protein of the blood, that has four subunits. Each subunit is defined with a different color and contains two α and two β subunits arranged with a quaternary structure in the form, $\alpha_2\beta_2$. Hemoglobin is, therefore, a hetero-oligomeric protein [75].

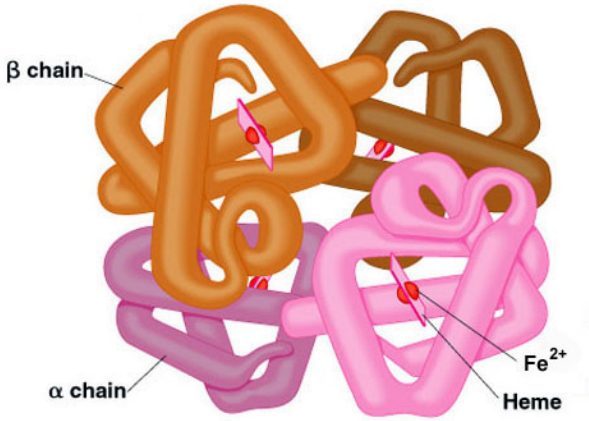


Figure 2.8 : Quaternary structure of hemoglobin [79].

3. FEATURE EXTRACTION FROM PROTEIN STRUCTURES

In this thesis, for protein fold classification and motif retrieval, different features extracted from primary and secondary protein structures were used. To classify proteins according to their folds GAL network, SOM and SOM-SD methods were used. GAL network and SOM were tested with features extracted from primary protein structures and representing physicochemical attributes of amino acids. SOM-SD was tested with a new data structure extracted from protein SS to classify the proteins. For motif retrieval, GHT-based three methods were used with the features extracted from protein SSs. These features will be explained in the following sections.

3.1 Feature Extraction for Primary Protein Structures

To deal with the fold classification problem, Ding and Dubchak extracted the following six attributes from protein sequences; amino acid composition, predicted SS, hydrophobicity, normalized van der Waals volume, polarity and polarizability [37]. Of the above six attributes, only amino acid composition contains 20 components, with each representing the occurrence frequency of one of the 20 native amino acids in a given protein. For the remaining five attributes, each contains 21 components [50]. These attributes are shown in Table 3.1.

Table 3.1 : The six attributes that form feature vector, their symbols and number of components. Feature vector's dimension is 125.

Symbol	Attribute	#Components
C	Amino acid composition	20
S	Predicted SS	21
H	Hydrophobicity	21
V	Normalized van der Waals volume	21
P	Polarity	21
Z	Polarizability	21

The composition vector is computed directly from amino acid sequence. Given that the 20 amino acids which are ordered alphabetically (A,C,D,E,F,G,

H,I,K,L,M,N,P,Q,R,S,T,V,W,Y) are represented as AA_1, AA_2,..., AA_{19} and AA_{20}; and the number of occurrences of AA_i in the entire sequence is denoted as n_i, the composition vector is defined as:

$$\frac{n_1}{L}, \frac{n_2}{L}, ..., \frac{n_{19}}{L}, \frac{n_{20}}{L} \tag{3.1}$$

where L is the length of the sequence [51]. The predicted SS is divided into three groups which are helix, strand and coil; and also for the other four attributes, those of hydrophobicity, normalized van der Waals volume, polarity and polarizability, the 20 amino acids are divided into three groups according to the magnitudes of their numerical values. These three groups are shown in Table 3.2. Three descriptors, composition (C), transition (T) and distribution (D) are calculated for a given attribute to describe the global percent composition of each of the three groups in a protein, the percent frequencies with which the attribute change its index along the entire length of the protein, and the distribution pattern of the attribute along the sequence, respectively. Five different amino acid attributes produce five parameter vectors each containing 3(C)+3(T)+5×3(D)=21 scalar components. The sixth parameter vector used was the vector of the percent composition of amino acids. Consequently, the feature vector includes $20 + 21 \times 5 = 125$ features (components) for a protein [9].

Table 3.2 : Five attributes (predicted SS, hydrophobicity, normalized van der Waals volume, polarity and polarizability) and the division into three groups. While the first attribute is related to SSs, the other four attributes are related to amino acids.

Property	Group 1	Group 2	Group 3
S	Helix	Strand	Coil
H	Polar R,K,E,D,Q,N	Neutral G,A,S,T,P,H,Y	Hydrophobic C,V,L,I,M,F,W
V	0-2.78 G,A,S,C,T,P,D	2.95-4.0 N,V,E,Q,I,L	4.43-8.08 M,H,K,F,R,Y,W
P	4.9-6.2 L,I,F,W,C,M,V,Y	8.0-9.2 P,A,T,G,S	10.4-13.0 H,Q,R,K,N,E,D
Z	0-0.108 G,A,S,D,T	0.128-0.186 C,P,N,V,E,Q,I,L	0.219-0.409 K,M,H,F,R,Y,W

3.2 Feature Extraction for Secondary Protein Structures

To extract the features for secondary protein structures a new file format having *.nss* extension was obtained for each individual protein. In these files each SS of protein

has the format shown in below:

type <ALPHA,BETA>

CharName = E beta beta-strand

CharName = B bridge short-beta-bridge

CharName = G 3_{10} 3_{10}-helix

CharName = H helix alpha-helix

CharName = I pi pi-helix

CharName = T turn turn

CharName = S bend bend

CharName = C coil random coil

number of residues

chain

sequence of residues

(BP1 BP2 sheet) for each residue (only for beta)

coordinates of the first α-Carbon (x,y,z)

number of the first residue

coordinates of the last α-Carbon (x,y,z)

number of the last residue

barycentre (x,y,z)

orientation (x,y,z)

Of the above attributes coordinates of first α-Carbon atom and coordinates of last α-Carbon atom related to each SS in protein were used as features for structural block comparison with GHT-based methods; and orientation related to each SS in protein was used as feature to classify proteins with SOM-SD.

3.2.1 Protein Gaussian image (PGI)

Extended Gaussian Images (EGI) are useful for representing the shapes of surfaces [80]. Surface normal vector for any object can be mapped onto a unit sphere, called the Gaussian sphere. Mapping is called the Gaussian image of the object. The mapping is that the surface normals for each point of the object are placed so that their tails lie at the center of the Gaussian sphere and heads lie on a point on the sphere appropriate to the particular surface orientation. It can be extended assigning

a weight to each point on the Gaussian sphere equal to the area of the surface having the given normal. In this situation, the mapping is called extended Gaussian image. EGI represents the histogram of the surface orientations (see Figure 3.1). The EGI introduced for applications of photometry by B.K.P. Horn [80] has been extended by K. Ikeuchi [81, 82] (the complex EGI). PGI is a representation in the Gaussian image

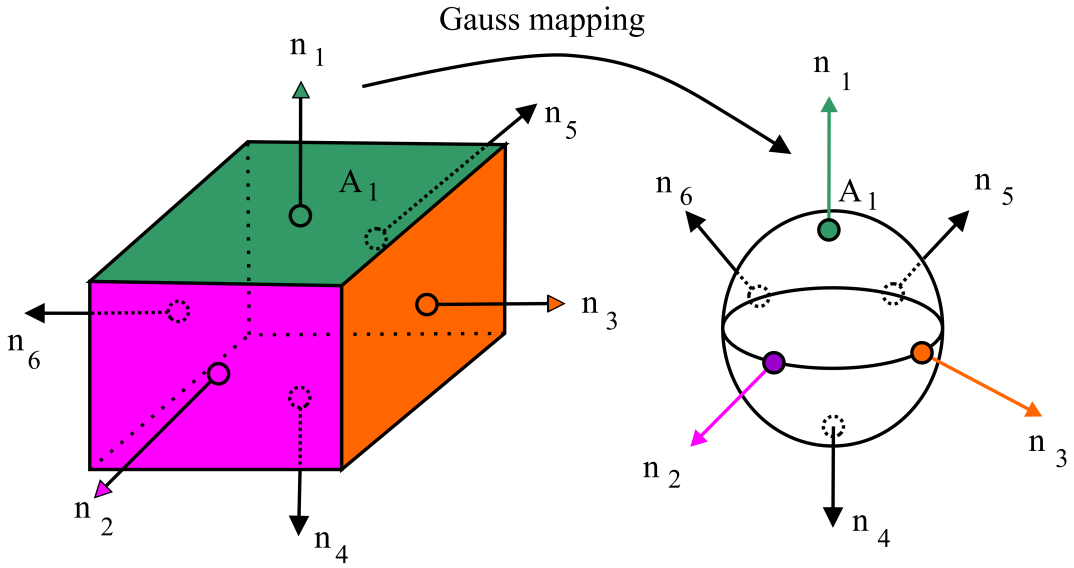


Figure 3.1 : The EGI of cube. The EGI of a 3D object or shape is an orientation histogram that records the distribution of surfeace area with respect to surface orientation [80].

in which each SS is mapped with a unit vector from the origin of the sphere having the orientation of the SS. Each point of the sphere surface contains the data orientation (length, location of starting and ending residue, etc.) of the existing protein SSs having the corresponding orientation. In Figures 3.2 and 3.3, two examples of 1FNB and 4GCR proteins are shown in PGI representation.

The chain sequence of SS is recorded as a list which is mapped on the sphere surface and a directed graph is obtained. Nodes of the graph include the orientation information related to SS. In Figure 3.4 green arrows represent the chain sequence (directed graph) related to Greek Key motif.

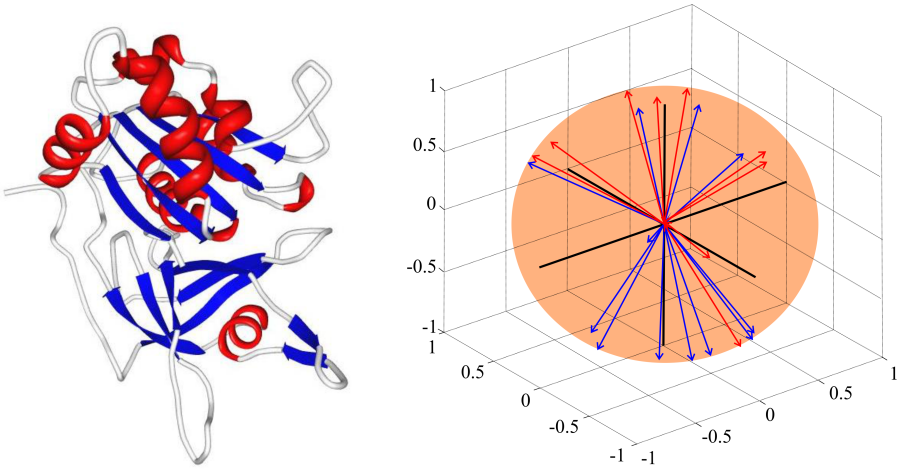


Figure 3.2 : Left picture generated by JMol represents the SSs of protein 1FNB. Right picture is PGI of the 1FNB. For both pictures blues are β-sheets and reds are α-helices.

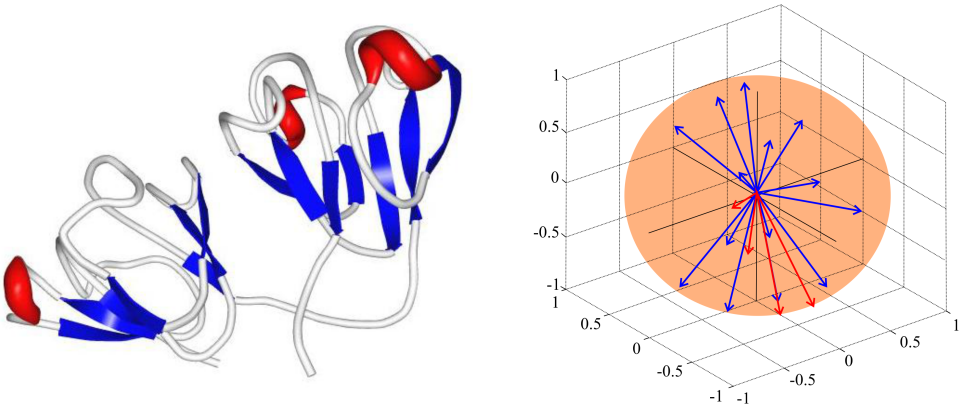


Figure 3.3 : Left picture generated by JMol represents the SSs of protein 4GCR. Right picture is PGI of the 4GCR. For both pictures blues are β-sheets and reds are α-helices.

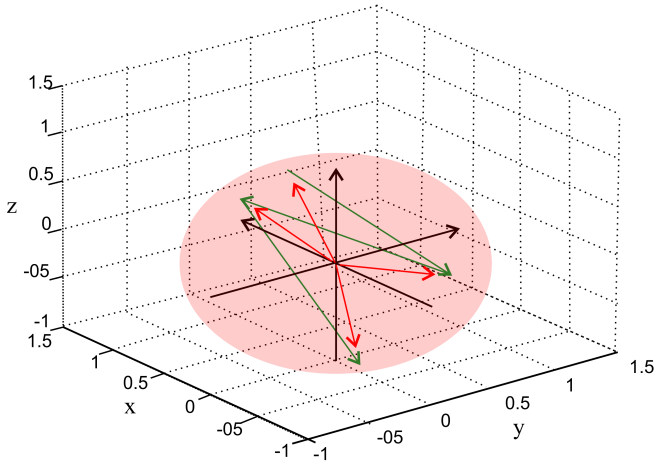


Figure 3.4 : PGI of Greek Key motif contained in protein 1FNB. Red arrows represent the Greek key motif, while the green lines show the sequence of SSs.

3.3 Feature Reduction

Feature reduction is to transform the data from high dimensional space to fewer dimensional space. The aim of this technique is to define the most significant features in order to save time without changing success rate. There are several methods related to feature reduction; in this thesis divergence analysis is employed for this task.

3.3.1 Divergence analysis

Divergence analysis is generally used to examine scatter of classes and distances between classes in feature space. In divergence analysis the best d features are selected among the given n features. We can say that it is used to determine the best features. In this thesis divergence analysis was used to reduce dimension of feature vector. To calculate divergence value within-class scatter matrix and between-class scatter matrix are used. These two matrices can be used as class separation criteria. Within-class scatter matrix W is the covariance matrix of the features (attributes) of the given class. Between-class scatter matrix B is the covariance of class means. Divergence value is proportional to between-class scatter matrix and inversely proportional to the within-class scatter matrix [83]. Divergence value is calculated using (3.2):

$$
\begin{aligned}
W_i^j &= \sum_t (\underline{\beta}_i^{tj} - \underline{\widehat{\mu}}_i^j) \cdot (\underline{\beta}_i^{tj} - \underline{\widehat{\mu}}_i^j)^T \\
\underline{\widehat{\mu}}_i &= \sum_{k=1}^{K} \underline{\widehat{\mu}}_i^k \qquad W_i = \sum_{k=1}^{K} W_i^k \\
B_i &= \sum_{k=1}^{K} (\underline{\widehat{\mu}}_i - \underline{\widehat{\mu}}_i^k) \cdot (\underline{\widehat{\mu}}_i - \underline{\widehat{\mu}}_i^k)^T \\
D_i &= tr((W_i)^{-1} B_i) \\
j &= 1,2,...,K; \quad i = 1,2,..,n
\end{aligned}
\tag{3.2}
$$

Here, $\underline{\beta}_i^{tj}$ represents the feature vector belonging to j-th class and having i dimension; $\underline{\widehat{\mu}}_i^j$ represents the mean of feature vectors belonging to j-th class and having i dimension; W_i^j represents within-class scatter matrix of j-th class; B_i represents between-class scatter matrix; K represents the number of classes; D_i represents the divergence value in i-dimension; and $tr(.)$ represents the application of trace process to the matrix obtained as a result of division.

Divergence value provides information about the scatter of the vectors formed by chosen features. Small divergence values and large divergence values represent vectors' scattering and vectors' aggregating, respectively in the feature space [84]. So, in this thesis, the features having large divergence values were searched for better classification performance and dimension reduction.

4. PROTEIN CLASSIFICATION METHODS

Protein fold classification and motif retrieval are important issues for proteomics. There are several methods to classify proteins according to their folds in the literature. In this thesis to classify proteins three neural network models were used: GAL, SOM and SOM-SD networks. This thesis has a novelty because to the best of our knowledge, these methods have not been used for protein fold classification task. Besides, for motif retrieval problem three methods based on GHT were developed. These fold classification and motif retrieval methods will be described in the following sections.

4.1 Grow and Learn (GAL) Network

In protein fold classification, it is desired that the decision-making processes lead to high performances with low computational loads, and are controlled with few parameters. These requirements are almost satisfied by the Grow and Learn (GAL) network. GAL is an incremental neural network for supervised learning, and determines the number of nodes during training if need arises. The network grows when it learns and shrinks when it forgets [85]. GAL represents the distribution of feature vectors according to the minimum distance measure. Computational loads of training and classification processes of GAL are rather low. Moreover, there is not any parameter to be determined before the training.

4.1.1 GAL network structure

The structure of GAL network is portrayed in Figure 4.1. The first layer is the layer of the input units. The second layer is that of the exemplars (prototypes) and the third is the class layer. The number of nodes in the exemplar layer is automatically determined during the training. When an input feature vector $\mathbf{X}$ is presented to the network, the distances between $\mathbf{X}$ and the weight vectors ($\mathbf{W}_j$) of exemplars, E_j nodes, are computed using a suitable metric, e.g., Euclidean distance. The *winner-takes-all* ensures that only one node will be activated, namely the node whose weight vector

is closest to the input vector is determined as the winner node. Network structure is

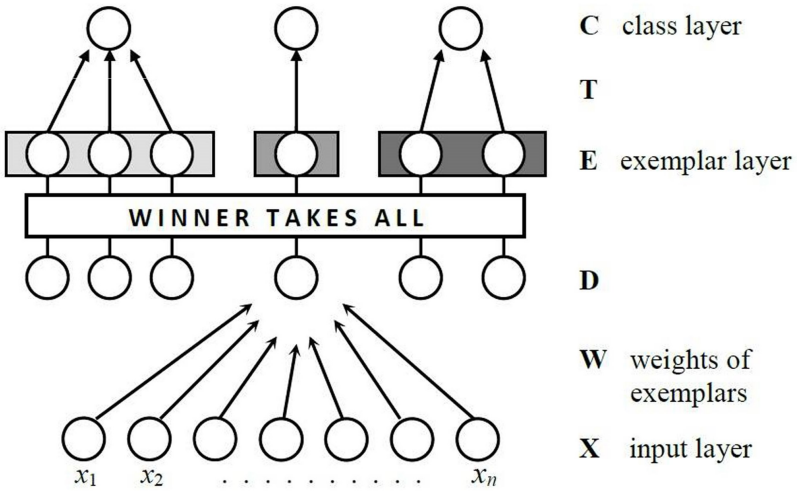


Figure 4.1 : GAL network structure. $\mathbf{X}$ is the input vector. $\mathbf{W}_j$ is the weight vector of the exemplar node E_j. D_j is the distance between $\mathbf{X}$ and $\mathbf{W}_j$. The nodes in the exemplar layer compete and only one of E_j is active. C is the layer of class units. n is feature space dimension [85].

described by the following equations [84]:

$$
\begin{aligned}
D_j &= \sum_{i=1}^{n} (x_i - w_{ji})^2 \\
E_e &= \begin{cases} 1, & D_e = \min_j (D_j) \\ 0, & \text{otherwise} \end{cases} \\
T_{ec} &= \begin{cases} 1, & \text{if } e \text{ is an exemplar of class } c \\ 0, & \text{otherwise} \end{cases} \\
C_c &= \sum_e E_e \cdot T_{ec}
\end{aligned}
\tag{4.1}
$$

where, x_i denotes the i-th element of input feature vector $\mathbf{X}$. $\mathbf{W}_j$ is the weight vector of the exemplar node E_j, and w_{ji} denotes the i-th element of $\mathbf{W}_j$. D_j is the distance between input vector $\mathbf{X}$ an $\mathbf{W}_j$. Only one of E_j is active, namely that whose weight vector is closest to the input vector which in turn activates the corresponding class node. C is the layer of class nodes. T_{ec} is the connection between exemplar e to class c. T_{ec} values are 1 or 0 depending on whether e is an exemplar of class c or not. These connections are initially set to zero, and become 1 during the training. The activations of class nodes are computed by a dot product, at this last stage Boolean OR operation is thus executed over the exemplars representing the same class. n is the dimension of feature space.

4.1.2 Partitioning of feature space by GAL network

As an example to depict how GAL partitions the feature space, a two-dimensional phantom feature space is formed, see Figure 4.2. In this phantom space, it is seen that there are two different classes each having three nodes (exemplars). As the nearest distance measure is used in the classification, a hyperplane passes through the two closest nodes which belong to different classes, and this hyperplane is equidistant to both nodes. A piecewise class boundary is thus created with several hyperplanes (the term "hyperplane" is used for the general form in an n-dimensional feature space; it is a "line" in two dimensions, and a "plane" in three dimensions). The same mechanism is valid in multi-dimensional feature spaces.

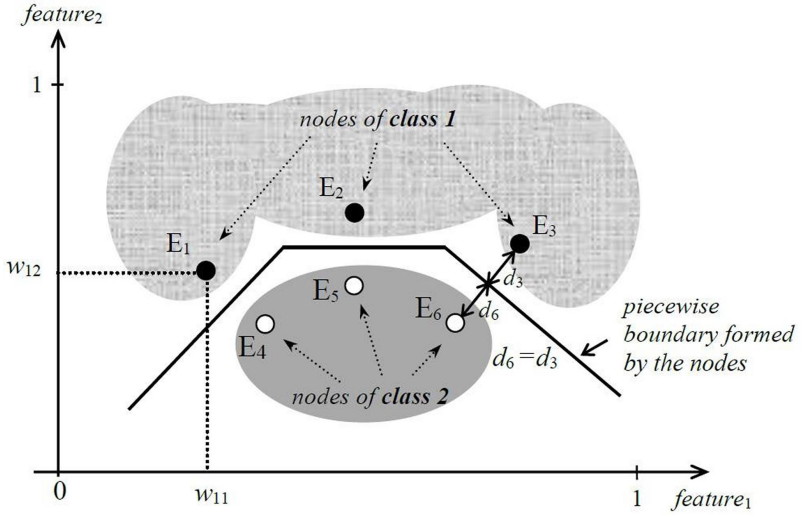


Figure 4.2 : GAL's partitioning of a two-dimensional phantom feature space. $E_{1-3}$and E_{4-6} are nodes (exemplars) of class 1 and class 2, respectively. A piecewise boundary is approximated with several hyperplanes, each one passing equidistantly through the closest two nodes of opposite classes. $feature_1$ and $feature_2$ values are assumed to be scaled within $[0-1]$ range [84].

4.1.3 Learning and forgetting in GAL

The most important feature of GAL is that the number of E_j nodes are automatically determined and gradually increased during learning according to the distribution of the feature vectors in the feature space. The choice of exemplar layer nodes, hence, the structure of GAL network depends on the order of the initially given input vectors. A node stored in previous iterations may become redundant when a new node closer to class boundary is generated. In order to keep the topology of the network simple those redundant nodes may be excluded from the network with the *forgetting* algorithm of

GAL. The aim of the forgetting algorithm is to detect and exclude those nodes that do not change the performance of the network when being eliminated. The steps during the learning are mentioned below [85].

- $Step1$: Initially choose a number of feature vectors randomly from the training set as many as the number of classes. Each vector should be chosen among the patterns of a particular class. Set each chosen vector as an exemplar layer node of GAL. Initialize the iteration number.
- $Step2$: Decrease the iteration number. If the iteration number is equal to zero terminate the learning algorithm, otherwise go to *Step 3*.
- $Step3$: Choose a vector randomly from the training set, and present it to the network as input.
- $Step4$: Calculate the distances between the input vector and the nodes and determine the closest node according to (4.1). If the classes of the closest node and input vector are the same, go to *Step 2*. Otherwise, go to *Step 5*.
- $Step5$: Include the input vector as a new node to the network. The input vector is assigned as the associated weight vector of the new node. Go to *Step 2*.

Forgetting algorithm can be run several times during the training depending on the iteration number. The steps during the forgetting are given below. Iteration number is initialized as the number of exemplar layer nodes.

- $Step1$: Temporarily remove a node from the network in some order and present this node as an input vector to the network.
- $Step2$: Calculate the distances between input vector and network nodes. If the classes of the input vector and the closest node are the same, go to *Step 4*.
- $Step3$: Include the input vector again in the network.
- $Step4$: Decrease the iteration number. If the iteration number is equal to zero terminate the forgetting algorithm, otherwise go to *Step 1*.

4.2 Kohonen's Self-Organizing Map (SOM)

Kohonen's SOM, which is developed by Tuevo Kohonen in 1982 [86], is a type of neural network which uses unsupervised learning method. The aim of the SOM learning algorithm [87] is to learn a feature map which, given a vector in the input space return a point in the output space. This is obtained in the SOM by associating each point in the output space to a different neuron (output node). Given an input vector, the SOM returns the coordinates within the output space, of the node with the closest weight vector. Thus, the set of output nodes induces a partition of the input space, where input vectors that are close to each other will activate neighbor output nodes. In the training of Kohonen network not only the weights of the winner output node but also weights of its neighbors within a pre-determined neighborhood are updated. During learning, as the iteration number increases the neighborhood size is decreased nonlinearly [84]. The structure of SOM network is depicted in Figure 4.3. The distance between the j-th node ($\boldsymbol{w}_j$) in the output layer and the input vector $\boldsymbol{x}$ is calculated as follows:

$$D_j = \sum_{i=1}^{n} (x_i - w_{ji}(k))^2 \tag{4.2}$$

where n is the feature vector dimension and k is the iteration number. Training of

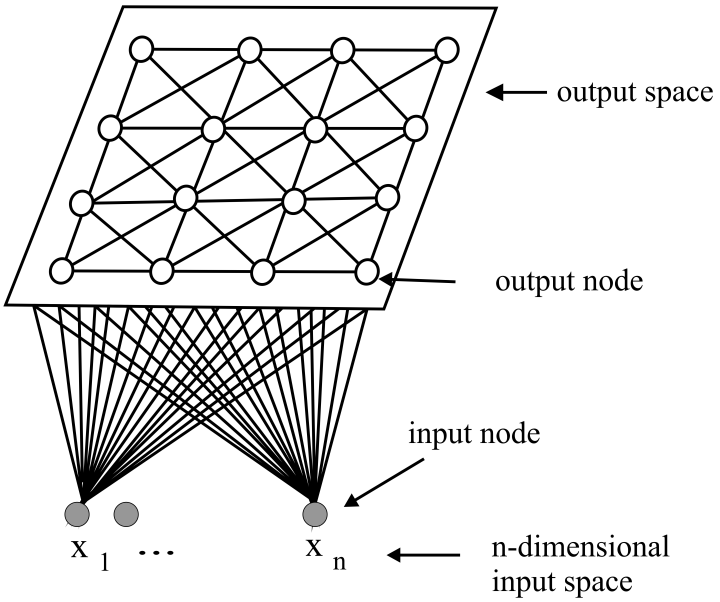


Figure 4.3 : Input and output spaces related to Kohonen's SOM network.

Kohonen's SOM network is as follows [88]:

Step 1. Before starting the learning; number of output nodes, number of iterations and neighborhood function are determined. Initial weights of the output nodes are set to

random values within [0-1] range.

Step 2. A vector randomly chosen from the training set is presented to the network as input.

Step 3. The distances between the input vector and network nodes are calculated using (4.1). Here, x_i and w_{ji} represent the i-th element of the input vector and i-th weight of the j-th output node, respectively $i = 1, 2, ..., n$.

Step 4. j-th output node having the minimum distance is determined.

Step 5. The weights of the j-th output node and its neighbors are updated using the expression below.

$$w_{ji}(k+1) = w_{ji}(k) + \eta(k) \cdot (x_i - w_{ji}(k)) \tag{4.3}$$

Here, $\eta(k)$ is learning rate and k is iteration number.

Step 6. Number of iterations is reduced. If the number of iterations is not equal to 0, Step 2 and other steps are repeated. If the number of iterations is 0, the learning algorithm is terminated.

After completing the training, class labels are assigned to the output nodes. To accomplish the labeling, each vector in the training set is fed to the trained network and the winner node at the output layer whose weight vector lies closest to the input vector is determined. The output nodes are associated with training data classes according to majority voting, i.e, the training data class that is assigned most frequently to an output node becomes its label.

The number of output nodes of SOM network should be determined before the training process. Inability to determine the optimum number of nodes decreases the classification performance. Using an excessive number of nodes causes elongation of the classification time. In GAL network, output nodes are located close to the boundaries. However, in SOM network, the training algorithm distributes (places) the output nodes homogeneously through the input feature space. SOM's training algorithm, with this strategy, causes the use of more output nodes in multi-dimensional feature space [84].

4.2.1 Partitioning of feature space by SOM network

SOM network's partitioning of feature space is similar to that of GAL network. There are two important differences between SOM and GAL networks: (i) The number of output nodes in GAL network is determined according to the needs during training, and (ii) weights of the output nodes in GAL network are determined by assigning the input vectors' elements directly to those output nodes [84].

4.3 Self Organizing Maps for Structured Data (SOM-SD)

In this part of the thesis proteins are classified according to their folds using SOM-SD. A new data representation PGI is derived from EGI and in order to validate the effectiveness of the PGI representation of the protein structure we employ an unsupervised framework for structured data (SOM-SD [89]) in a practical structural learning problem, where each protein is represented by a PGI. SOM-SD represents an extension of the SOM framework, where the input space is structured domain and the computational framework is similar to that defined for recursive neural networks [90, 91].

SOM-SD is a fully unsupervised model, namely an extension of traditional SOM, for the processing of labelled directed acyclic graphs (DAGs). The extension is obtained by using the unfolding procedure adopted in recurrent and recursive neural networks, with the replicated neurons in the unfolded network comprising of a full SOM. The essential idea of recursive neural networks is to model each node of an input DAG by a multilayer perceptron, and then to process the DAG from its sink nodes toward the source node, using the structure of the DAG to connect the neurons from one node to another [89]. In this section the same notation as in [89] was used.

SOM-SD framework can be defined by using a computational framework similar to that defined for recursive neural networks [91]. The class of functions which can be realized by an RNN can be characterized as the class of functional DAG transductions $\tau : \mathcal{I}^{\#} \rightarrow \mathbb{R}^k$, which can be represented in the following form $\tau = g \circ \widehat{\tau}$, where $\widehat{\tau} : \mathcal{I}^{\#} \rightarrow \mathbb{R}^n$ is the encoding function and $g : \mathbb{R}^n \rightarrow \mathbb{R}^k$ is the output function. $\widehat{\tau}$ is defined recursively as:

$$\widehat{\tau}(\mathcal{D}) = \begin{cases} \mathbf{0}(\text{the null vector in } \mathbb{R}^n), & \text{if } \mathcal{D} = \text{void graph} \\ \tau(\mathbf{y}_s, \widehat{\tau}(\mathcal{D}^{(1)}, ..., \mathcal{D}^{(c)})), & \text{otherwise} \end{cases} \tag{4.4}$$

where $\mathcal{D}$ is a DAG, $\mathbf{y}_s$ is the label of the supersource of $\mathcal{D}$ and τ is defined as:

$$\tau : \mathbb{R}^m \times \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{\text{c times}} \rightarrow \mathbb{R}^n \tag{4.5}$$

A typical form for τ is:

$$\tau(\mathbf{u}_\upsilon, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(c)}) = \mathbf{F}(\mathbf{W}\mathbf{u}_\upsilon + \sum_{j=1}^{c} \widehat{\mathbf{W}}_j x^j + \theta) \tag{4.6}$$

where $\mathbf{F}_i(\mathbf{v}) = sgd(\upsilon_i)$ (sigmoidal function), $\mathbf{u}_\upsilon \in \mathbb{R}^m$ is a label, $\theta \in \mathbb{R}^n$ is the bias vector, $\mathbf{W} \in \mathbb{R}^{m \times n}$ is the weight matrix associated with the label space, $\mathbf{x}^{(j)} \in \mathbb{R}^m$ are the vectorial codes obtained by the application of the encoding function $\widehat{\tau}$ to the subgraphs $\mathcal{D}^{(j)}$ (i.e., $\mathbf{x}^{(j)} = \widehat{\tau}(\mathcal{D}^{(j)})$), and $\widehat{\mathbf{W}}_j \in \mathbb{R}^{m \times m}$ is the weight matrix associated with the j-th subgraph space. The output function g is generally realized by a feed-forward neural network.

The key consideration to adapt this framework to the unsupervised SOM approach, is that the function τ maps information about a node and its children from a higher dimensional space (i.e., $m + c \cdot n$) to a lower space (i.e., n). The aim of the SOM learning algorithm is to learn a feature map

$$\mathcal{M} : \mathcal{I} \rightarrow \mathcal{A} \tag{4.7}$$

which, given a vector in input space $\mathcal{I}$ returns a point in the output space $\mathcal{A}$. This is obtained in the SOM by associating each point in $\mathcal{A}$ to a different neuron. Given an input vector υ, the SOM returns the coordinates within $\mathcal{A}$ of the neuron with the closest weight vector. Thus, the set of neurons induces a partition of the input space $\mathcal{I}$. In typical applications $\mathcal{I} \equiv \mathbb{R}^m$, where $m \gg 2$, and $\mathcal{A}$ is given by a two dimensional lattice of neurons. In this way, input vectors which are close to each other will activate neighbor neurons in the lattice. SOM-SD represents an extension of the SOM framework, where the τ function is implemented in an unsupervised learning framework, in order to generalize (4.7) to deal with the case $\mathcal{I} \equiv \mathcal{Y}^{\#(c)}$, i.e., the input space is a structured domain with labels in $\mathcal{Y}$. The function

$$\mathcal{M}^{\#} : \mathcal{Y}^{\#(c)} \rightarrow \mathcal{A} \tag{4.8}$$

is realized by defining (4.4) as shown in (4.9)

$$\mathcal{M}^{\#}(\mathcal{D}) = \begin{cases} nil_{\mathcal{A}}, & \text{if } \mathcal{D} = \text{void graph} \\ \mathcal{M}_{node}(\mathbf{y}_s, \mathcal{M}^{\#}(\mathcal{D}^{(1)}), ..., \mathcal{M}^{\#}(\mathcal{D}^{(c)})), & \text{otherwise} \end{cases} \tag{4.9}$$

where $s = source(\mathcal{D}), (\mathcal{D}^{(1)}, ..., \mathcal{D}^{(c)}$ are the subgraphs pointed by the outgoing edges leaving from $s, nil_{\mathcal{A}}$ is a special coordinate vector into the discrete output space $\mathcal{A}$, and

$$\mathcal{M}_{node} : \mathcal{Y} \times \underbrace{\mathcal{A} \times \ldots \times \mathcal{A}}_{\text{c times}} \rightarrow \mathcal{A} \tag{4.10}$$

is a SOM, defined on a generic node, which takes as input the label of the node and the "encoding" of the subgraphs $\mathcal{D}^{(1)}, ..., \mathcal{D}^{(c)}$ according to the $\mathcal{M}^{\#}$ map. By "unfolding" the recursive definition in (4.9), it turns out that $\mathcal{M}^{\#}(\mathcal{D})$ can be computed by starting to apply $\mathcal{M}_{node}$ to leaf nodes (i.e., nodes with null outdegree), and proceeding with the application of $\mathcal{M}_{node}$ bottom-up from the frontier nodes (sink nodes) to the supersource of the graph $\mathcal{D}$. In this process $\mathcal{M}_{node}$ returns the coordinates of the winning neuron, which, due to the data reduction capability of the SOM, still constitutes a reduced descriptor of the node. An example of structured data and how the computation described above proceeds is shown in Figures 4.4 and 4.5, respectively. In this case, $nil_{\mathcal{A}}$ is represented by the coordinates (-1, -1). A highlighted neuron in Figure 4.5 refers to the best matching neuron for the given input vector.

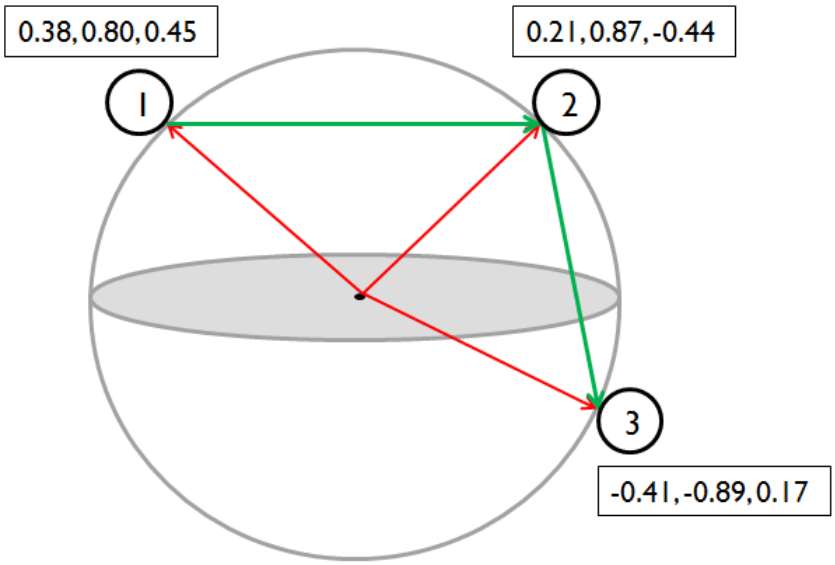


Figure 4.4 : Structured-data (DAG), obtained using the PGI representation, on a cube. Red lines show the SSs and the green lines represent the sequence of the SSs and forms a graph including three edges and three vertexes (nodes). Each node has a three-dimensional label.

At the application of $\mathcal{M}_{node}$ each label in $\mathcal{Y}$ encode in $\mathcal{U} \subset \mathbb{R}^m$. So, for each node (vertex) v in $vert(\mathcal{D})$ there is an m-dimensional vector $\mathbf{u}_v$ and the output space $\mathcal{A}$ is formed by a q-dimensional lattice of neurons. For example, if $q = 2$, and we have n_1 neurons on the horizontal axis and n_2 neurons on the vertical axis, then output

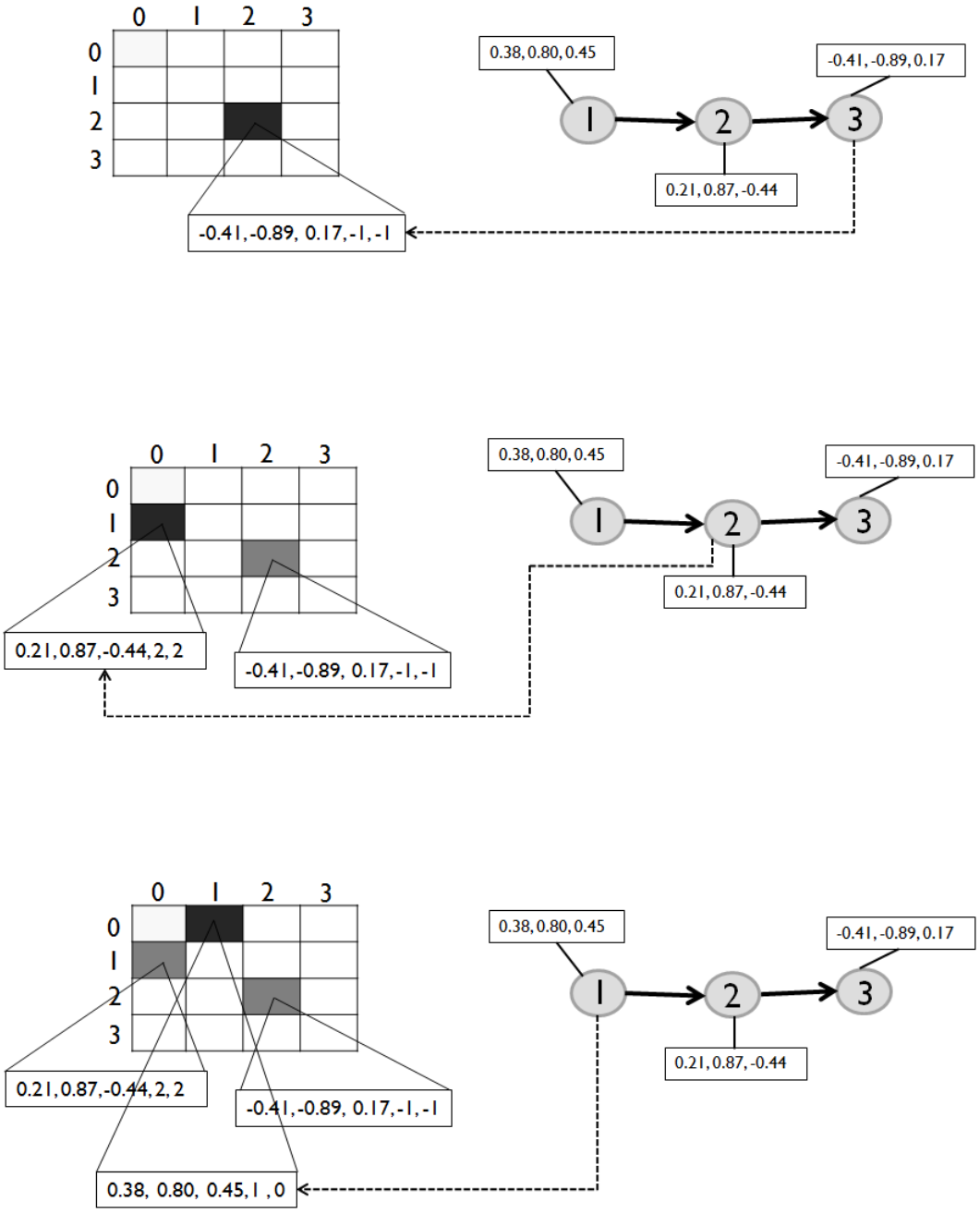


Figure 4.5 : Example of computation of $\mathcal{M}^{\#}$ for the graph related to structural data in Figure 4.4. The picture shows multiple applications of $\mathcal{M}_{node}$ to the nodes of the graph. First of all, the leaf node 3 is presented to $\mathcal{M}_{node}$, where the null coordinates are represented by (-1, -1). The winning neuron has coordinates (2, 2). This information is used to define the input vector representing node 2. This vector is then presented to $\mathcal{M}_{node}$ and the winning neuron is found to have coordinates (1, 0). Using both this information and the previous one, the input vector for node 1 can be composed and presented to $\mathcal{M}_{node}$. This time the winning neuron is found to have coordinates (0, 1) and it is associated to the whole graph, i.e., $\mathcal{M}^{\#}\mathcal{D} = (0, 1)$.

space $\mathcal{A} \equiv [1...n_1] \times [1...n_2]$ and the winning neuron is represented by the coordinate vector $\mathbf{c} \equiv (c_1, c_2) \in [1...n_1] \times [1...n_2]$ of the neuron which is the most active in this two-dimensional lattice. With the above assumptions, we have

$$\mathcal{M}_{node} : \mathbb{R}^m \times ([1...n_1] \times \cdots \times [1...n_q])^c \rightarrow [1...n_1] \times \cdots \times [1...n_q] \tag{4.11}$$

and the $m + cq$ dimensional input vector $\mathbf{v}$ to $\mathcal{M}_{node}$. Here, $\mathbf{v}$ represents the information about a generic node v and $\mathbf{v}$ is defined as

$$\mathbf{v} = [\mathbf{u}_v \mathbf{x}_{ch_1[v]} \mathbf{x}_{ch_2[v]} ... \mathbf{x}_{ch_c[v]}] \tag{4.12}$$

where $\mathbf{x}_{ch_i[v]}$ is the coordinate vector of the winning neuron for the subgraph pointed by the i-th pointer of v.

Given a DAG $\mathcal{D}$ in order to compute $\mathcal{M}^{\#}(\mathcal{D})$, the SOM $\mathcal{M}_{node}$ must be recursively applied to the nodes of $\mathcal{D}$.

Each neuron with coordinates vector $\mathbf{c}j$ in the q-dimensional lattice has an associated weight vector $\mathbf{w}_{\mathbf{c}_j} \in \mathbb{R}^{m+cq}$. The weights related to each neuron in the q-dimensional lattice $\mathcal{M}_{node}$ can be trained using the two-step process. In the first step, the neuron which is most similar to the input node $\mathbf{v}$, defined as in (4.12), is chosen as follows:

$$\mathbf{c}_{i*}(t) = arg \underbrace{min}_{c_i} \parallel \Lambda(\mathbf{v}(t) - \mathbf{w}_{\mathbf{c}_i}(t)) \parallel \tag{4.13}$$

where t is iteration and Λ is a $(m + cq) \times (m + cq)$ diagonal matrix which is used to balance the importance of the label versus the importance of the pointers. In the second step, the weight vector $\mathbf{w}_{\mathbf{c}_i*}$ is moved closer to the input vector $\mathbf{v}$

$$\mathbf{w}_{\mathbf{c}_r}(t+1) = \mathbf{w}_{\mathbf{c}_r}(t) + \eta(t) f(\Delta_{i*r})(\mathbf{v}(t) - \mathbf{w}_{\mathbf{c}_r}(t)) \tag{4.14}$$

where η is learning rate, $f(\Delta_{i*r})$ is neighborhood function and Δ_{i*r} is the topological distance between $\mathbf{c}_r$ and $\mathbf{c}_i*$. The neighborhood function takes the form of a Gaussian function

$$f(\Delta_{i*r}) = exp\left(-\frac{\Delta_{i*r}^2}{2\sigma^2}\right) \tag{4.15}$$

where σ is the spread function which determines the neighborhood size. As the learning proceeds and new input vectors are given to the map, the learning rate gradually decreases to zero according to the specified learning rate function type. A pseudocode of the training algorithm for $\mathcal{M}^{\#}$ is seen in Table 4.1.

Table 4.1 : Training algorithm for $\mathcal{M}^{\#}$.

Input : Set of training DAGs $T = \mathcal{D}_{i=1,\ldots,N}$, c maximum outdegree of DAGs in T, map $\mathcal{M}_{node}$;
begin
  Randomly set the weights for $\mathcal{M}_{node}$
  repeat
    Randomly select $\mathcal{D} \in T$ with uniform distribution;
    List($\mathcal{D}$) $\leftarrow$ an inverted topological order for vert($\mathcal{D}$)
    for v = first(List($\mathcal{D}$)) : last(List($\mathcal{D}$)) **do**
      train($\mathcal{M}_{node}([\boldsymbol{u}_v \mathcal{M}^{\#}(ch_1[v])\mathcal{M}^{\#}(ch_2[v]) \cdots \mathcal{M}^{\#}(ch_c[v])])$);
    end
end

4.4 Hough Transform for Protein Motif Retrieval

Hough transform (HT) is a feature extracting method using coordinate transformation [92]. It was introduced by P.V.C. Hough in 1962 and patented by IBM. Hough used angle-radius parameters exclusively for retrieving the straight lines. HT was extended for extracting circles by R.O. Duda and P.E. Hart in 1972 [93] and for retrieving parabolas by H. Wechsler and J. Sklansky in 1973 [94]. Later it was generalized as GHT by Ballard for retrieving arbitrary shapes [95]. Basically, the original HT is a voting process where each contour point detected in the image votes for all possible patterns passing through that point. As an example in the implementation to detect straight lines, votes are accumulated in an array $A(\rho, \theta)$, where θ is the angle made by the normal to the straight line with the x-axis and ρ is the perpendicular distance of the straight line from the origin. The representation of the straight line in (ρ, θ) form is

$$x\cos\theta + y\sin\theta = \rho \tag{4.16}$$

This accumulator array $A(\rho, \theta)$ is called the Hough Space (HS). The number of votes for each cell in $A(\rho, \theta)$ represents the number of pixels in the searched pattern extracted from the image. In this process each pixel in the image space is mapped to a sinusoidal curve of (4.16) in the HS. So HT is a transformation from a point to a curve [96].

The HT allows the identification of geometric objects by transformation of the points in an image in a parameter space and is translation and rotation insensitive. Figure 4.6 shows how three points in the left x, y coordinate space can be transformed into lines

in a parameter space in which parameters are slope m and intercept q. Parameter space represents the parameters of the bundle of lines which cross each particular point (three points in the example). If the same transformation is carried out for every point of the segment on the left in Figure 4.6, then the parameter space will consist of a bundle of lines all of which will intersect in a particular point of the mq space corresponding to the actual slope and intercept of the line containing three points. The HT consists of performing a vote procedure in a quantized parameter space and retrieving the peaks which will have a high probability of corresponding to descriptors of the actual instances of the wanted shapes in the query image. The same principle

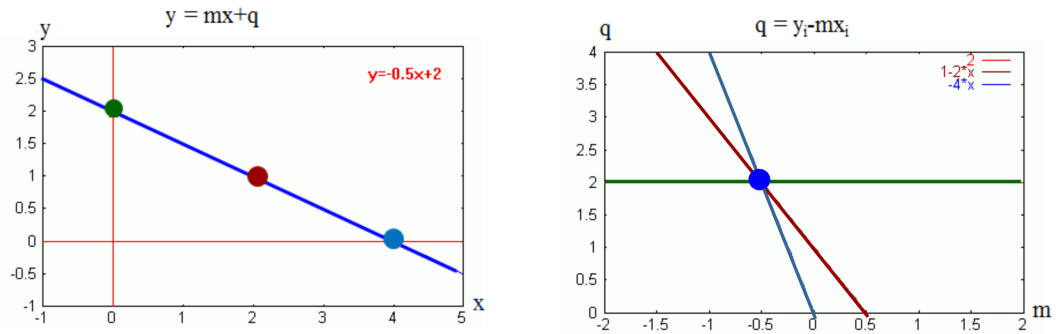


Figure 4.6 : The principle of Hough transform for lines: geometric space (left) and parameter space (right).

also works for other shapes. Figure 4.7 shows the case of a circle [97], in which a point on the circle in the geometric space corresponds a circle in the parameter space. The parameters x and y positions in the geometric space correspond the coordinate of the of the center in the parameter space and the intercept coordinate of the circles represents the coordinate of the center in the geometric space. Geometric spaces are

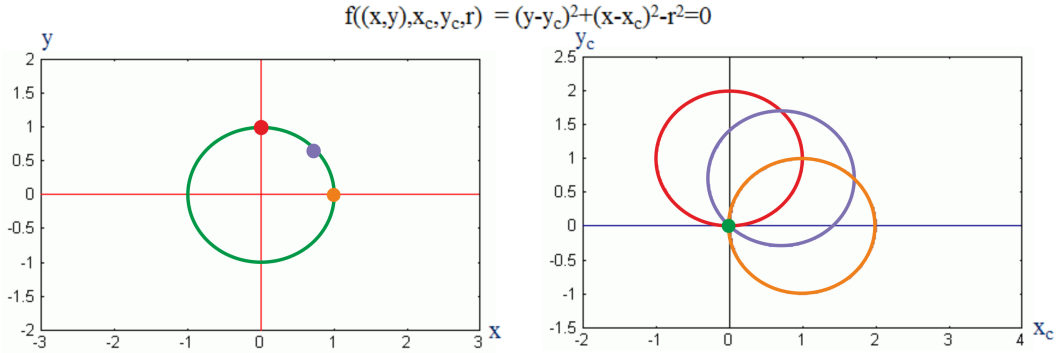


Figure 4.7 : Retrieval of circles of known radius: geometric space (left) and parameter space (right).

used as parameter spaces for retrieval of general objects, whatever the shape, by fixing a Reference Point (RP) internal to the objects. This extension is called Generalized

Hough Transform (GHT) [95]. In GHT arbitrary shapes are represented in the HS which consists of the parameters of the rigid motion - in 2D (x, y, θ, s) x, y representing translation, θ rotation and s a scaling factor. For each evidence extracted from the image, like in the original Hough approach, a mapping rule is defined which determines the value of the parameters of rigid motion (locus of points in HS) compatible with the evidence. Figure 4.8 shows an arbitrary shape and Reference Table (RT) parameters (α, r). For retrieving the arbitrary shape the process is that [98]

1. Pick a RP (e.g., (x_c, y_c))
2. Draw a line from the RP to the boundary
3. Compute ϕ (i.e., perpendicular to gradient's direction)
4. Store the RP (x_c, y_c) as a function of ϕ (i.e., build the RT)
5. Quantize the parameter space

$$P[x_{c_{min}} ... x_{c_{max}}][y_{c_{min}} ... y_{c_{max}}]$$

6. For each edge point (x, y)
   (a) Using the gradient angle ϕ, retrieve from the RT all the (α, r) values indexed under ϕ
   (b) For each (α, r) compute the candidate RPs:

$$x_c = x + rcos(\alpha)$$
$$y_c = y + rsin(\alpha)$$

   (c) Increase counters by giving vote to the possible RP locations:

$$++(P[x_c][y_c])$$

7. Possible locations of the object contour are given by local maxima in $P[x_c][y_c]$
8. If $P[x_c][y_c]$ has the maximum number of votes, then the object contour is located at (x_c, y_c)

Figures 4.9 and 4.10 are pictorial 2D representations of applying GHT to discrete objects such as proteins made up of different SSs. In Figure 4.9 arrows represent

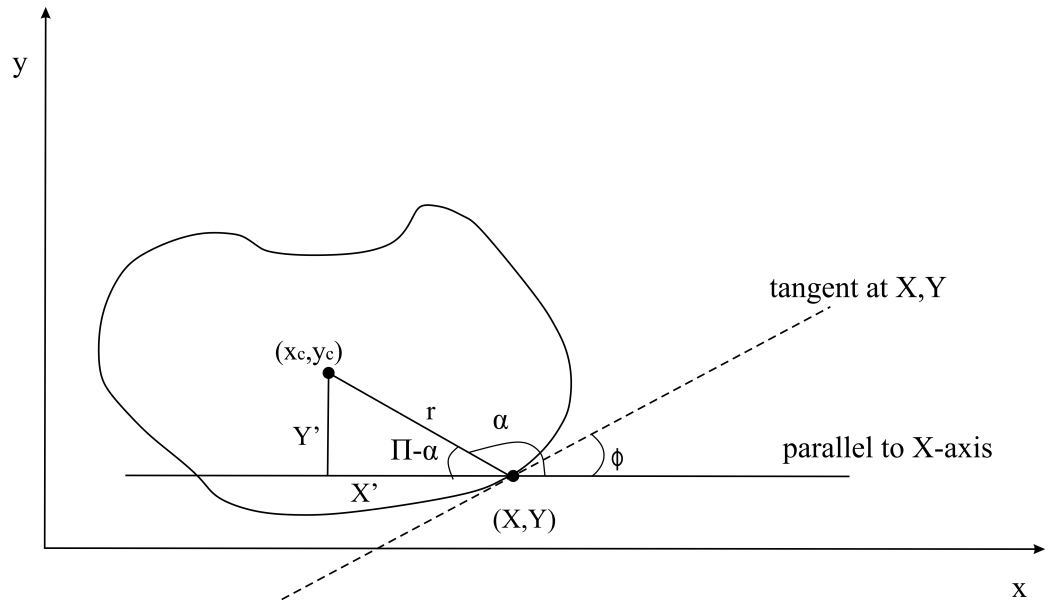


Figure 4.8 : Retrieval of arbitrary shapes using RT parameters (α, r).

protein SSs and the motif consists of two helices and three strands. At the top left of the figure query protein and the RP related to this protein are shown. RP is generally determined as the geometric mean of the SSs. At the bottom left mapping rule is seen. To determine the mapping rule, for each SS (two helices and three strands) the distance between SS midpoint and RP, ρ; and the angle between SS segment and the segment between SS midpoint and the RP, θ are calculated. At the right of the figure votes space is seen and the point having the maximum number of votes is determined as the RP. In Figure 4.10 query protein and mapping rule are the same as in Figure 4.9. In the right of this figure, each vote is given to different points. So there is no peak formation.

In this thesis the GHT is exploited for comparison and search of structural similarity between a given motif or domain or entire protein and the proteins of a database like PDB [2]. Note that, if the searched structure is just a component of a protein (like a structural motif or a domain) the same algorithm supports the detection and the statistical distribution of these components.

Here, GHT-based three algorithms are used for motif retrieval. The first algorithm uses single SS, the second one uses SS couple co-occurrences and the third one uses SS triplet co-occurrences for searching a general motif in the macromolecule (protein). In all three algorithms firstly a motif model is created by using SSs of

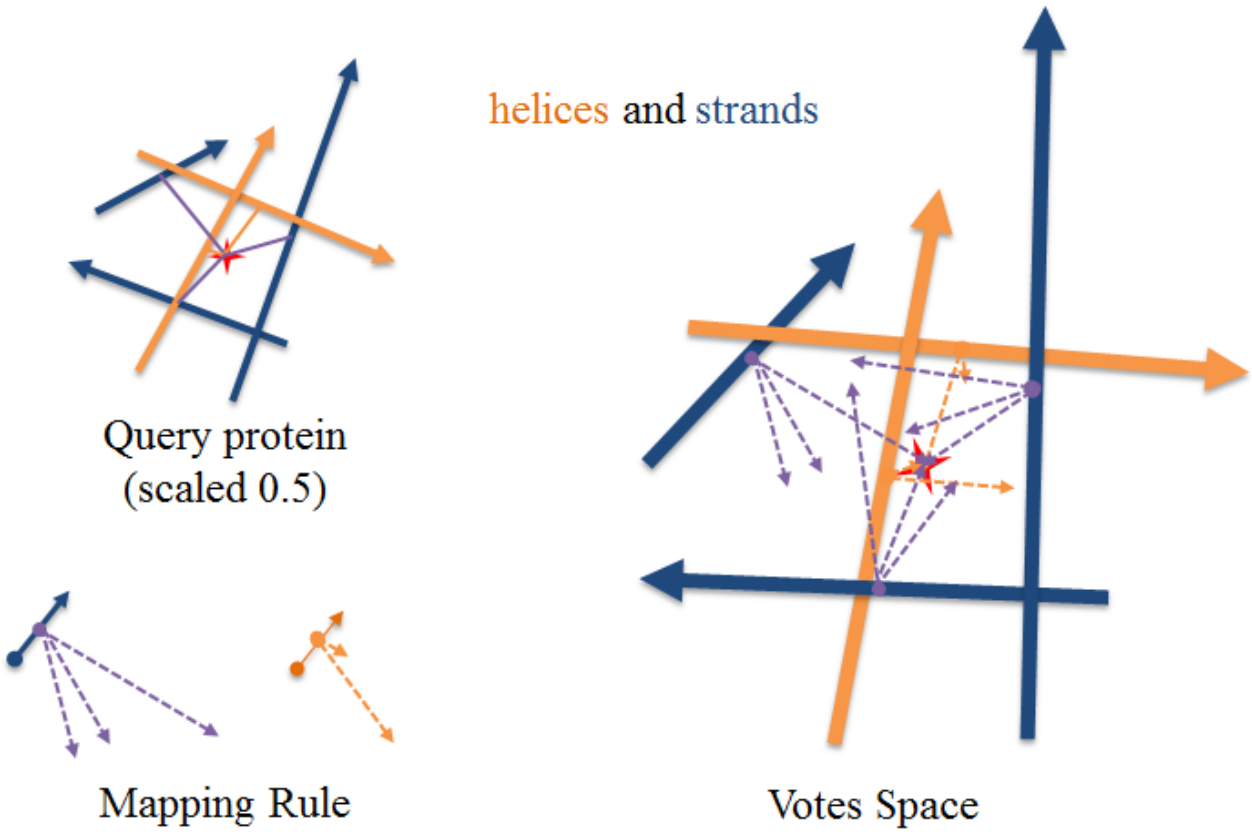


Figure 4.9 : Applying GHT to proteins, positive case with peak formation.

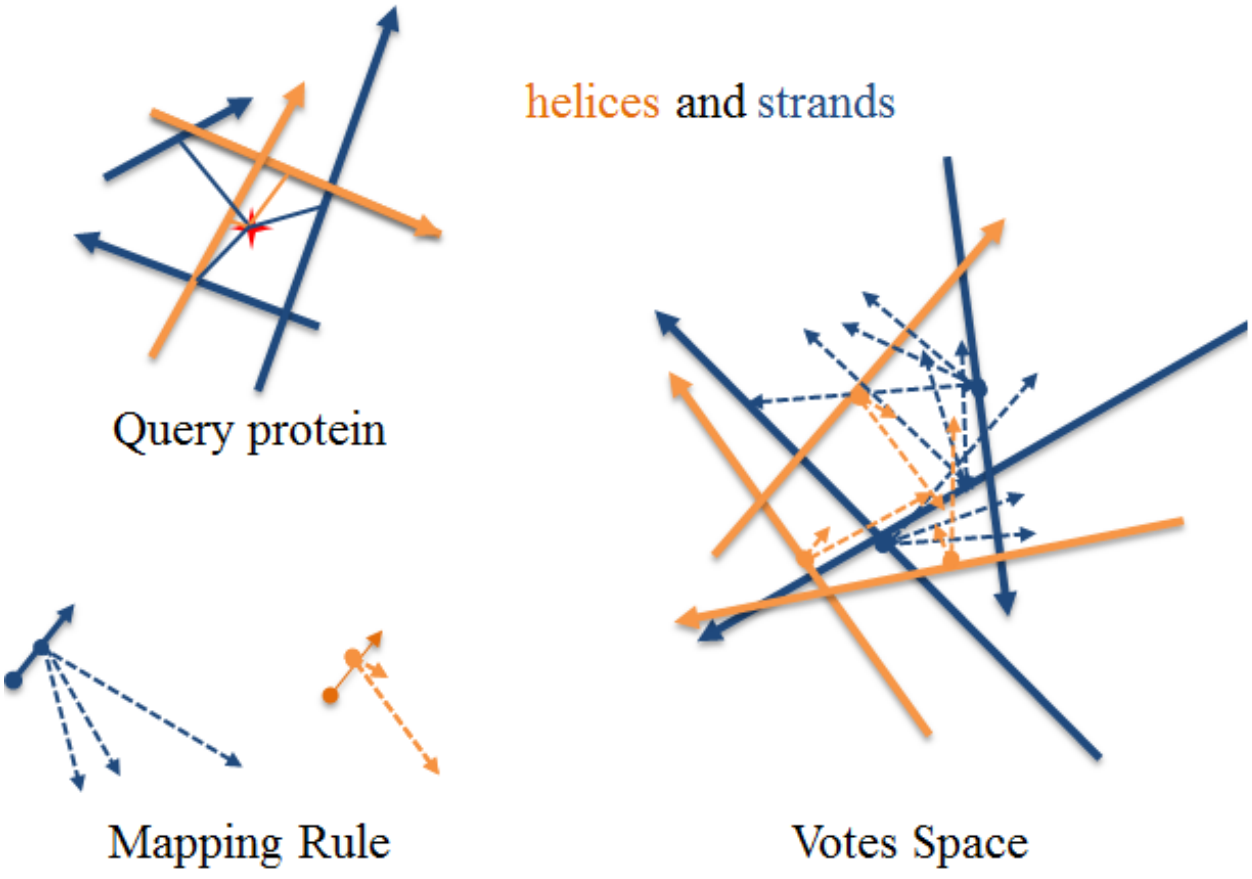


Figure 4.10 : Applying GHT to proteins, no peak formation is detected.

a given macromolecule, e.g. for creating five-SS motif, five SSs of the protein are selected randomly and used.

4.4.1 Selected primitive aggregate: The single secondary structure (SSS)

This method adopts the single SS as primitive for the voting process. The SS being a helix or a sheet is represented by a straight segment on the regression line from all the C_α atoms of the segment. The extremes are determined by the projection of the terminal C_α atoms. The selective component of the RT consists of two parameters, ρ and θ, (see Figure 4.11a); ρ is the segment length between RP and SS midpoint A, and θ is the angle between SS axis and the segment $\overrightarrow{A-RP}$. The mapping rule which determines the candidate RP locations, for a given SS, is a circle on a plane perpendicular to the axis of the SS (see Figure 4.11b), with radius $r = \rho \sin\theta$, having the center along the SS axis and with a displacement $d = \rho \cos\theta$ from midpoint A. Each SS of the protein under scrutiny contributes on a circular locus on the parameter space. The candidate RP locations are detected as the points of intersections of these circles and, in ideal conditions, the number of intersections is just S_1. S_1 is the number of SSs in the query motif. Figures 4.12a and 4.12b show the parameter

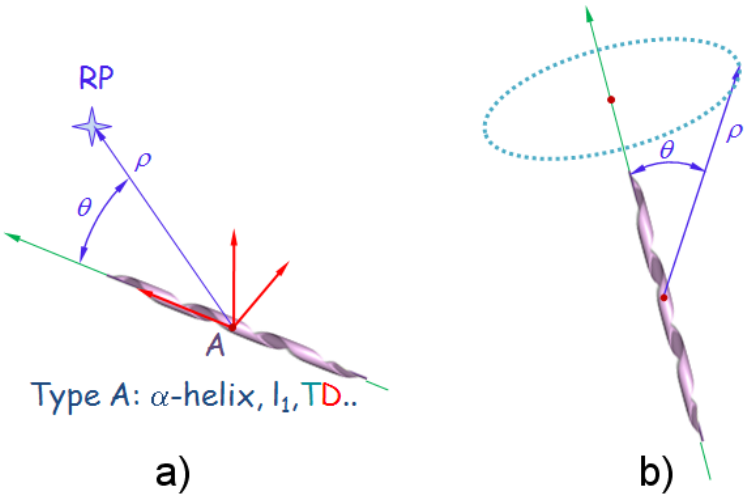
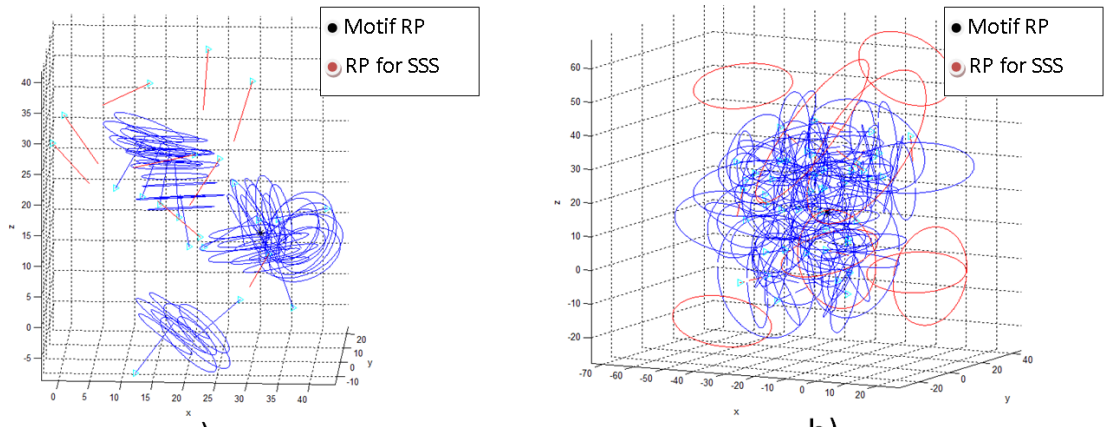


Figure 4.11 : a) Single SS and RT parameters. b) Locus of candidate RP positions in parameter space.

space resulting from the search of a homogeneous motif consisting of four SSs (β-sheets) and five heterogeneous SSs (three α-helices and two β-sheets). In these two examples the peaks, consisting of four blue circle intersections and three red and two blue intersections are located for four-SS and five-SS motifs, respectively. For what concerns the implementation, it is worth to point out that in order to detect the



a) b)

Figure 4.12 : a) Parameter space formed by applying the SSS strategy on the 22 SSs of 1FNB protein. b) The same on the 46 SSs of 7FAB protein. RP and maximum circle intersections are almost coincident for both cases.

peaks of intersections, the voting space is smoothed by the accumulation of nearby votes (within a given radius). In this application time complexity is $O(n^2)$, where n is the number of votes in the vote space (note that only the relevant votes are stored in memory, there is not a matrix with all the possible cells). After smoothing, the peaks are detected avoiding to pick high votes that however are not the top of a peak but lie close to one such peak [99–101].

4.4.2 Selected primitive aggregate: The secondary structure couple co-occurrences (SSC)

In this method a couple of SSs set up a local reference system, e.g. having the origin in the middle point of the first SS, the y-axis on its SS axis and the x-axis on the plane defined by the y-axis and the midpoint of the second SS, then the z-axis is orthonormal to the previous two. In this reference system, the motif RP coordinate is determined, and for each couple of SSs of the protein under scrutiny that matches a motif couple, the candidate RP location is uniquely fixed.

The number of motif couples and protein couples is given by 2-combinations of m and N respectively: $C(m,2)$ and $C(N,2)$. Here, m is the number of motif SSs and N is the number of protein SSs. As consequence, the computational complexity of the motif retrieval process is $O(N^2m^2)$ [102]. For every couple in the motif, a tuple is introduced in the RT where the selective component that characterizes the couple co-occurrence is composed by the three parameters (see Figure 4.13) [103]: Md, Ad and φ. Md is the Euclidean distance between middle points of two SSs, Ad is the shortest distance

between middle points of two SSs axis and φ is the angle between two SSs translated to present common extreme. These three parameters are used as comparison parameters. The motif couple parameters are stored in the RT. Around the axis of a SS a local reference system can rotate but fixing an external point (i.e. the middle point of the selected second SS) no degree of freedom remains and the RP position is precisely fixed. In Figure 4.13 a couple of SSs (A and B) is represented. The local reference system is evidenced together with the three quoted parameters and the correspondent RP position. To build the RT the barycenter of the motif is defined for each couple

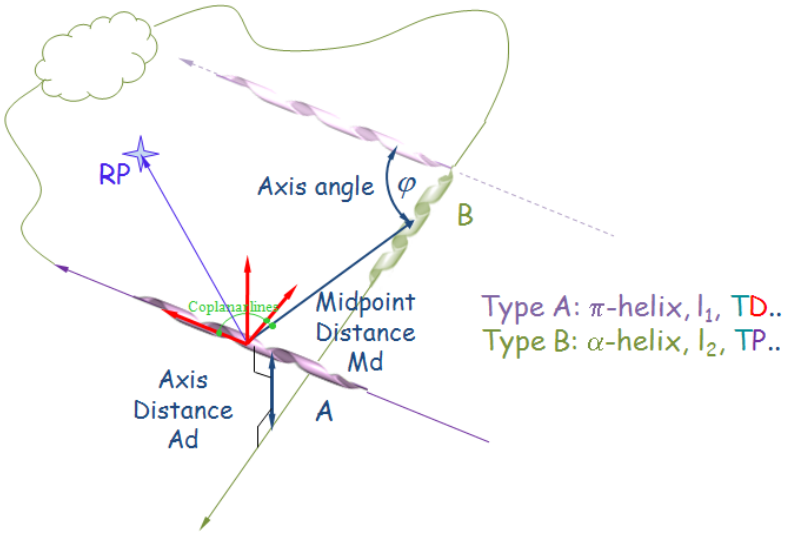


Figure 4.13 : The co-occurrence couple local reference system.

of the motif model. So for each couple, the RP coordinates must be determined with this local reference system. These coordinates constitute the tuples of the RT. The cardinality of the RT will be the number of motif couples, C. A macromolecule has N_{SS} couples:

$$N_{SS} = (N,2) = \frac{N!}{(N-2)!2!} \tag{4.17}$$

where N is the number of protein SSs. For each of these couples three parameters Md, Ad and φ are computed. Every couple of macromolecule is compared to the couples in the motif. If the couples have the same parameter values, a vote is given to the candidate RP which is calculated by using the above coordinates (mapping rule). For each motif couple the mapping rule is reduced to a single location. In Table 4.2 a pseudocode is given for searching all possible motifs in a set of M proteins and in Figure 4.14 a graphic sketch of this process is given. Here, the aim is searching the motif model on the top right in the macromolecule on the left. To do this, couple parameters are used. In this figure only the parameter Md is represented. Md values

($d1$, $d2$, $d3$) for the three couples in the motif are stored in the RT. If the motif couple parameters are equal to macromolecule couple parameters the RP is determined according to the mapping rule related to that motif couple. On the top of Figure 4.14 a complete instance of the model is present, so the correspondent RP position will gather three contributions (only one is graphically shown). Instead on the bottom right, an instance constituted of two SSs is present, and only one contribution is given in the correspondent RP position. Being $m = h + s$ the number of SSs in the motif (h the

Table 4.2 : Algorithm for the retrieval of all possible r motifs contained in a set of M proteins using SSC method.

```
    Input  : Protein DSSP files; N_i: number of protein SSs; m: number of motif SSs
    Output : Locations of candidate motifs in the accumulator A_RP, representing the para-
             meter space
1   for i=1 to M do
2       Calculate all m combinations of N_i: r=C(N_i,m)
3       for j=1 to r do
4           Find the motif barycenter RP
5           Calculate the number of motif couples: c = C(m,2)
6           Calculate the number of protein couples: p = C(N_i,2)
7       end
8           for k=1 to c do
9               Compute the motif couple parameters: Md_k, Ad_k and φ_k (RT constituents)
10          end
11           for l=1 to p do
12              Compute the protein couple parameters: Md_l, Ad_l and φ_l
13           end
14              for k=1 to c do
15                  if match (Md_k, Ad_k, φ_k and Md_l, Ad_l and φ_l) then A_RPl = A_RPl + 1
16              end
17              Compute the peaks in HS
18              Assign the position with the expected votes as candidate RP
19  end
```

number of helices and s the number of strands) and considering both homogeneous and heterogeneous couples, the cardinality of the RT is given by: $S_2 = (m^2 - m)/2$ (precisely they are divided in $h \times s$ heterogeneous votes) and $(h^2 - h)/2$ homogeneous with helices and $(s^2 - s)/2$ homogeneous with strands.

S_2 is the expected peak intensity, and when the motif is heterogeneous, also the quoted peak decomposition can be useful for discrimination purposes. We remark that,

also the single contribution of a couple can be weighted by the couple composition itself [99–101].

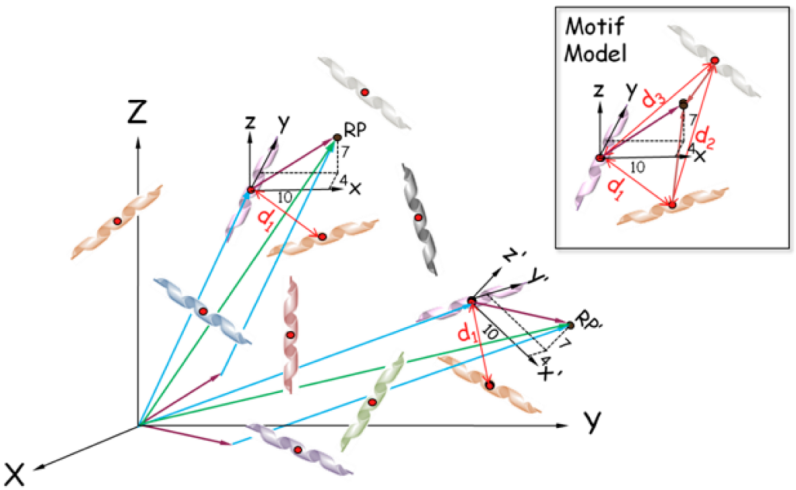


Figure 4.14 : The voting process for a couple of helices. On the top right a sketch of motif having three SSs, and on the left RP locations of two compatible couples.

4.4.3 Selected primitive aggregate: The secondary structure triplet co-occurrences (SST)

In 3D, middle points of three SSs can be joined and an imaginary triangle is composed. So, through the SS triplets a local reference system is setup, e.g. having the origin in the triangle barycenter, the y-axis passing through the farthest vertex, the x-axis lying on the triangle plane and orthonormal to y-axis, and the z-axis following the triangle plane normal (see Figure 4.15). With this reference system the motif RP coordinates are determined, and also in this case for each triplet of SSs of the protein under scrutiny that matches a motif triplet, the candidate RP location is uniquely fixed [104]. For every triplet in the motif, a tuple is introduced in the RT in which the selective component that characterizes the triplet is composed of three parameters: the lengths of the triangle edges. For each motif triplet the mapping rule is reduced to a single location.

For the experimentation, firstly a motif is defined selecting some SSs of the protein and then the number of triangles in this motif is calculated using (3.2). Here, m and t are the number of SSs of the motif and the number of motif triangles respectively.

$$t = C(m,3) = \frac{m!}{(m-3)!3!} \tag{4.18}$$

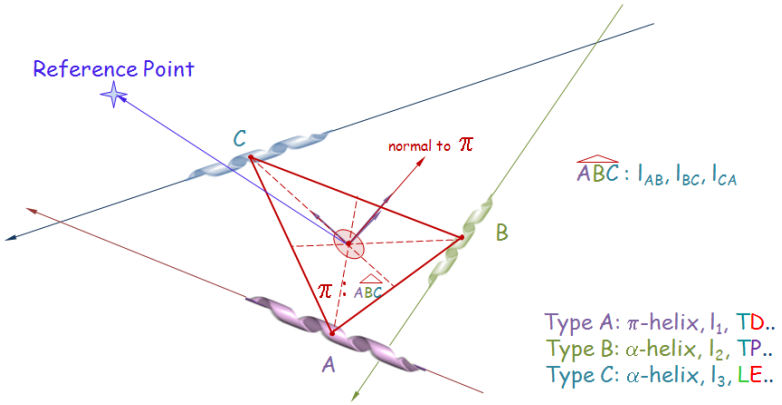


Figure 4.15 : Local reference system representation for the A, B, C triplet. The comparison parameters are the length of triangle edges (i.e. the midpoints' distances). Other discriminant parameters can be considered such as: type of SS (π-helices, α-helices, β-strands, etc.), SS lengths (i.e. number of amino acids), types of amino, etc.

For every triangle in the motif, three edges' lengths are computed and these parameters are stored in the RT. To build the RT, the barycenter of the motif is defined for each motif triangle, and the position of the RP referring to the local triangle reference system is also stored in RT. So the mapping rule is figured out with these definitions. For each triangle this information constitutes the tuple of the RT. The cardinality of the RT will be the number of motif triangles t. Then the number of all possible triangles, T, in the protein is computed using (4.19). In this equation N represents the number of SSs in the protein. For each triangle in the protein, edge lengths are calculated. Then motif triangles and protein triangles are compared. For every correspondence a vote is given to the point which is determined by the mapping rule contained in the RT. Table 4.3 shows a pseudocode of this algorithm for searching all possible motifs in a set of M proteins. The time/space computational complexity of the motif retrieval process, in this approach, is $O(N^3m^3)$.

$$T = C(N,3) = \frac{N!}{(N-3)!3!} \tag{4.19}$$

In this case the cardinality of the RT is given by: $S_3 = C(m,3) = m(m^2-3m+2)/6$ precisely they are divided in $hs(h+s-2)/2$ heterogeneous and $C(h,3)$ homogeneous with helices and $C(s,3)$ homogeneous with splines. S_3 is the expected peak intensity, and as previously with SSC when the motif is heterogeneous, the quoted peak decomposition can be useful for discrimination purposes. Note that also in this case the single triplet contribution can be weighted on the basis of its composition.

Table 4.3 : Algorithm for the retrieval of all possible r motifs contained in a set of M proteins using SST method.

```
   Input  : Protein DSSP files; N_i: number of protein SSs; m: number of motif SSs
   Output : Locations of candidate motifs in the accumulator A_RP, representing the para-
            meter space
1    for i=1 to M do
2       Calculate all m combinations of N_i: r=C(N_i,m)
3       for j=1 to r do
4          Find the motif barycenter RP
5          Calculate the number of motif triangles: c = C(m,3)
6          Calculate the number of protein triangles: p = C(N_i,3)
7       end
8          for k=1 to c do
9             Compute the lengths of motif triangle: d1_k, d2_k and d3_k (RT constituents)
10         end
11          for l=1 to p do
12             Compute the lengths of protein triangle: d1_l, d2_l and d3_l
13          end
14             for k=1 to c do
15                if match (d1_k, d2_k, d3_k and d1_l, d2_l and d3_l) then A_RPl = A_RPl + 1
16             end
17             Compute the peaks in HS
18             Assign the position with the expected votes as candidate RP
19   end
```

Figure 4.16 shows an example of five-SSs motif including one π-helices and four α-helices.

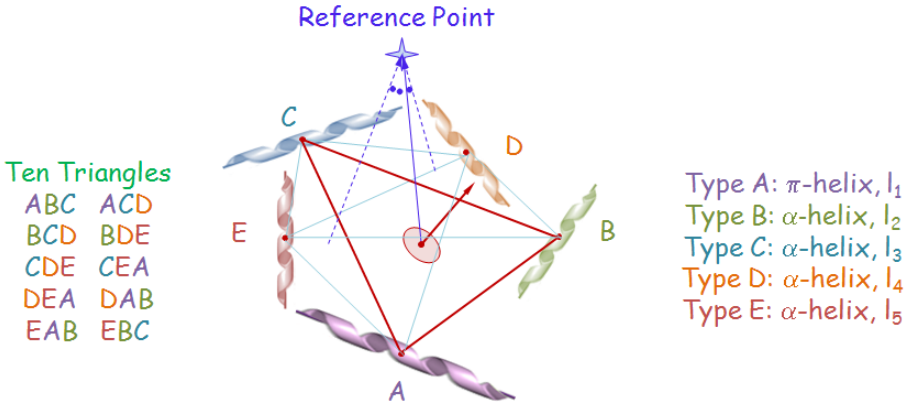


Figure 4.16 : Representation of a motif composed of five SSs (e.g. one π-helices and four α-helices, as shown on the right). In this case $t = 10$, as shown in the list on the left. Just one mapping is drawn completely in the protein, the corresponding RP location will receive one contribution for each of the ten triangles. In the picture ABC contribution is outlined the and only two others corresponding to triangles AEC and CDB are sketched.

Firstly a RP is determined for the motif (generally fixed on the motif barycenter, in this case it has been selected outside for evidencing graphically the voting process). In this case, there are ten triplets and these triplets compose ten triangles. For every triangle, the barycenter is computed and then the motif RP is located referring to the local coordinate system. This constitutes the mapping rule for the voting procedure to detect the candidate RP location(s). When comparing motif triangles and protein triangles, for every correspondence a vote is given to the candidate RP position defined with this mapping rule [99–101, 105].

5. SIMULATION RESULTS

5.1 Protein Database

The Research Collaboratory for Structural Bioinformatics Protein Data Bank (RCSB PDB) provides access to the data in the PDB, the single archive of experimentally determined structures of nucleic acids, proteins and complex assemblies [106]. The PDB was established at Brookhaven National Laboratories in 1971 as an archive for biological macromolecular crystal structures. In the beginning the archive held seven structures, and with each year a handful more were deposited [107]. The public archive currently contains >109,000 entries, derived data files and related data dictionaries. Data is obtained by X-ray crystallography, NMR (Nucleic Magnetic Resonance) spectroscopy and electron microscopy. In each of these methods, the scientists use many pieces of information to create the final atomic model. Primarily, the scientists have some kind of experimental data about the structure of the molecule. For X-ray crystallography, this is the X-ray diffraction pattern; for NMR spectroscopy, it is information on the local conformation and distance between atoms that are close to one another; and for electron microscopy, it is an image of the overall shape of the molecule.

Most of the structures included in the PDB archive were determined using X-ray crystallography. For this method, the protein is purified and crystallized, then subjected to an intense beam of X-rays. The proteins in the crystal diffract the X-ray beam into one or another characteristic pattern of spots, which are then analyzed to determine the distribution of electrons in the protein. The resulting map of the electron density is then interpreted to determine the location of each atom. The PDB archive contains two types of data for crystal structures. The coordinate files include atomic positions for the final model of the structure, and the data files include the structure factors (the intensity and phase of the X-ray spots in the diffraction pattern) from the structure determination. X-ray crystallography can provide very detailed atomic information,

showing every atom in a protein or nucleic acid along with atomic details of ligands, inhibitors, ions, and other molecules that are incorporated into the crystal. However, the process of crystallization is difficult and can impose limitations on the types of proteins that may be studied by this method. For example, X-ray crystallography is an excellent method for determining the structures of rigid proteins that form nice, ordered crystals. Flexible proteins, on the other hand, are far more difficult to study by this method because crystallography relies on having many molecules aligned in exactly the same orientation, like a repeated pattern in wallpaper. Flexible portions of protein will often be invisible in crystallographic electron density maps, since their electron density will be smeared over a large space [2].

NMR spectroscopy may be used to determine the structure of proteins. The protein is purified, placed in a strong magnetic field, and then probed with radio waves. A distinctive set of observed resonances may be analyzed to give a list of atomic nuclei that are close to one another, and to characterize the local conformation of atoms that are bonded together. This list of restraints is then used to build a model of the protein that shows the location of each atom. The technique is currently limited to small or medium proteins, since large proteins present problems with overlapping peaks in the NMR spectra. In the PDB archive, you will typically find two types of coordinate entries for NMR structures. The first includes the full ensemble from the structural determination, with each structure designated as a separate model. The second type of entry is a minimized average structure. These files attempt to capture the average properties of the molecule based on the different observations in the ensemble. You can also find a list of restraints that were determined by the NMR experiment. These include things like hydrogen bonds and disulfide linkages, distances between hydrogen atoms that are close to one another, and restraints on the local conformation and stereochemistry of the chain [2].

Electron microscopy is also used to determine structures of large macromolecular complexes. A beam of electrons is used to image the molecule directly. Several tricks are used to obtain 3D images. If the proteins can be coaxed into forming small crystals or if they pack symmetrically in a membrane, electron diffraction can be used to generate a 3D density map, using methods similar to X-ray diffraction. If the molecule is very symmetrical, such as in virus capsids, many separate images may

be taken, providing a number of different views. These views are then aligned and averaged to extract 3D information. Electron tomography, on the other hand, obtains many views by rotating a single specimen and taking several electron micrographs. These views are then processed to give the 3D information. For a few particularly well-behaved systems, electron diffraction produces atomic-level data, but typically, electron micrographic experiments do not allow the researcher to see each atom. Electron micrographic studies often combine information from X-ray crystallography or NMR spectroscopy to sort out the atomic details. Atomic structures are docked into the electron density map to yield a model of the complex. This has proven very useful for multimolecular structures such as complexes of ribosomes, tRNA and protein factors, and muscle actomyosin structures [2].

Data is obtained by using the methods described above and submitted by biologists and biochemists from all around the world to be freely accessible on internet via its member organizations' websites and is updated weekly. The mission is to maintain a single protein archive of macromolecular structural data (see Figure 5.1). Each structure published in PDB receives a four character alphanumeric identifier or accession number like 1FNB, 4GCR. PDB file format is a textual file format describing the three dimensional structures of molecules held in the PDB. PDB also provides atomic coordinates, sequences, side chains, SSs and atomic connectivity of the molecules. Here, structure files may be viewed using various free and commercial visualization programs and web browsers' plug-ins.

In this thesis, the dataset including primary structure attributes, and tested by GAL and SOM was taken from Ding and Dubchak [37]. The dataset is available on *http://ranger.uta.edu/ chqding/protein/*. The original training dataset and test dataset contain 313 and 385 proteins respectively. However, four of these proteins do not have sequence records (in training dataset 2SCMC and 2GPS, in test dataset 2YHX_1 and 2YHX_2). Accordingly we have 311 proteins for training dataset and 383 proteins for test dataset. None of the proteins in the test dataset has $>35\%$ sequence identity to those in the training dataset [37]. All these proteins belong to 27 folds including four structural classes. Table 5.1 shows these folds. Of these 27 fold types, types 1-6 belong to all-α structural class, types 7-15 to all-β class, types 16-24 to α/β class and types 25-27 to $\alpha+\beta$. So, the classification of 27 folds is one level deeper than that of four

Figure 5.1 : PDB website

structural classes [108–111]. Hence, it is more challenging and difficult to conduct prediction among the 27-fold types than among the four structural classes [29, 112].

The dataset including SS attributes is taken from PDB and derived by University of Naples Parthenope, Computer Vision and Pattern Recognition Laboratory. This dataset includes 20 proteins, and available on *http://opolat.cumhuriyet.edu.tr/data.rar* (See Table 5.2).

Table 5.1 : The most common 27 SCOP folds, structural classes that folds belong to and the number of proteins contained in training and test sets.

Fold No.	Fold name	Str. class	Train	Test
1	Globin-like	all-α	13	6
2	Cytochrome c	all-α	7	9
3	DNA-binding 3-helical bundle	all-α	12	20
4	4-helical up-and-down bundle	all-α	7	8
5	4-helical cytokines	all-α	9	9
6	Alpha; EF-hand	all-α	6	9
7	Immunoglobulin-like β-sandwich	all-β	30	44
8	Cupredoxins	all-β	9	12
9	Viral coat and capsid proteins	all-β	16	13
10	ConA-like lectins/glucanases	all-β	7	6
11	SH3-like barrel	all-β	8	8
12	OB-fold	all-β	13	19
13	Trefoil	all-β	8	4
14	Trypsin-like serine proteases	all-β	9	4
15	Lipocalins	all-β	9	7
16	(TIM)-barrel	α/β	29	48
17	FAD (also NAD)-binding motif	α/β	11	12
18	Flavodoxin-like	α/β	11	13
19	NAD(P)-binding Rossmann-fold	α/β	13	27
20	P-loop containing nucleotide	α/β	10	12
21	Thioredoxin-like	α/β	9	8
22	Ribonuclease H-like motif	α/β	10	12
23	Hydrolases	α/β	11	7
24	Periplasmic binding protein-like	α/β	11	4
25	β-grasp	$\alpha+\beta$	7	8
26	Ferredoxin-like	$\alpha+\beta$	13	27
27	Small inhibitors, toxins, lectins	$\alpha+\beta$	13	27

Table 5.2 : List of the proteins tested by SSC and SST methods. The SS segments are obtained by DSSP.

PDB ID	Description	Number of SSs
2Z98	FMN-dependent NADH-azoreductase	14
2Z9C	FMN-dependent NADH-azoreductase	15
2Z9B	FMN-dependent NADH-azoreductase	16
4GCR	GAMMA-B CRYSTALLIN	18
3E9O	Pre-mRNA-splicing factor 8	21
1FNB	FERREDOXIN-NADP+ REDUCTASE	22
3E9L	Pre-mRNA-processing-splicing factor 8	23
2PZN	Aldose reductase	24
3C3U	Aldo-keto reductase family 1 member C1	26
2Z7G	Adenosine deaminase	28
2Z7K	Queuine tRNA-ribosyltransferase	33
2PRL	Dihydroorotate dehydrogenase, mitochondrial	34
2QX8	Ribosyldihydronicotinamide dehydrogenase [quinone]	35
2QMY	Ribosyldihydronicotinamide dehydrogenase	36
3C94	Exodeoxyribonuclease I	37
2QX9	Ribosyldihydronicotinamide dehydrogenase [quinone]	38
3C95	Exodeoxyribonuclease I	39
3DC7	Putative uncharacterized protein lp_3323	43
3DHP	Alpha-amylase 1	44
7FAB	IGG1-LAMBDA NEW FAB (LIGHT CHAIN)	46

5.2 Protein Classification Results using Primary Structures

5.2.1 One-versus-others (OvO) method and performance measures

OvO prediction method is a simple and effective method [9, 113] for multi-class problems. To explain this method, suppose that there are K classes. Firstly we transform the multi-class problem to two-class problem. One class contains all the proteins belonging to the i-th fold which are labeled as *positive*, and the other class contains all other proteins that are labeled as *negative*. So we construct K binary classifiers to predict the protein folds. For example, in the first classifier one class

contains the first fold's proteins and the other class contains the other $K-1$ folds' proteins.

In the recognition process, new query protein is tested at each of the K binary classifiers to determine if it belongs to the given class or not. This leads to K scores from the K classifiers. Ideally only one of the K classifiers will show a positive result and the other classifiers show negative results, assigning the query protein to a unique fold [37].

In this section for $K = 27$, we try to solve 27-class protein fold classification problem. We made various tests to classify the protein fold patterns. To calculate the classification success rate for individual fold we used sensitivity (true positive rate, TPR) (5.1) and we generalized the sensitivity formula for 27-class to calculate overall success rate (5.2).

$$Ind.FoldSuccessRate = \frac{\sum TruePositive}{\sum TruePositive + \sum FalseNegative} \tag{5.1}$$

$$OverallSuccessRate = \frac{\sum_{i=1}^{27} TruePositive}{\sum_{i=1}^{27} TruePositive + \sum_{i=1}^{27} FalseNegative} \tag{5.2}$$

In some tests we did not use OvO method and we calculated classifier's performance using accuracy as below:

$$Accuracy = \frac{\sum TruePositive + \sum TrueNegative}{\sum Positive + \sum Negative} \tag{5.3}$$

5.2.2 Protein classification results by using GAL

In this section protein folds were classified using GAL network. Here, the dataset mentioned in Section 5.1 was used. The dataset produced by [37] includes 311 and 383 proteins (125 dimensional) for training and test sets, respectively. In the first experiment 27-class protein folds were classified using one classifier (without OvO) and the success rate was calculated using (5.3). The algorithm was tested with 3000 iterations. For the classification, GAL produced a network including 242 nodes and

the proteins were classified in 4.32 seconds. For each test the average performance over 50 runs was reported. The results related to this experiment is shown in Table 5.3.

Table 5.3 : Performance of GAL for 27-class fold classification problem.

Fold	Success Rate(%)	Fold	Success Rate(%)	Fold	Success Rate(%)
1	66.7	10	16.7	19	37.0
2	55.6	11	37.5	20	41.7
3	45.0	12	21.1	21	25.0
4	62.5	13	25.0	22	66.7
5	66.7	14	50.0	23	57.1
6	22.2	15	28.6	24	75.0
7	45.4	16	64.6	25	37.5
8	16.7	17	58.3	26	25.9
9	41.7	18	38.5	27	48.1
Overall Success Rate					44.1

To get unbiased training data for the classification of proteins we used 10-fold cross validation with GAL network. The training set is partitioned into ten folds with each fold containing almost equal number of patterns. Among the 10 sets, one of them is assigned as testing data to validate the data and the rest is used as training data. The process of cross-validation is repeated 10 times, where each of the 10 sets is used once as the validation model. Tests were done with 3000 iterations as previous. For each of ten folds, the network generated an average of 53 nodes. Classification process took an average of 1.1 seconds for each fold. As we have already expected, the success rate was increased. The results related to this experiment are shown in Table 5.4.

Table 5.4 : Performance of GAL for 27-class fold classification problem by using 10-fold cross validation technique.

Folds	1	2	3	4	5	6	7	8	9	10
Success Rate(%)	45.1	67.6	69.6	69.6	68.6	50.0	56.7	58.0	51.4	34.3
Overall Success Rate										57.1

As seen from Tables 5.3 and 5.4 proteins were classified with low success rate. Then, OvO method was used with GAL network to increase classification performance. The results related to this experiment are shown in Table 5.5. According to this table,

seven of 27 folds were classified perfectly (100%) and the minimum classification performance is 52.6% for the 12th fold.

Table 5.5 : Performance of GAL for 27-class fold classification problem by using OvO prediction method.

Fold	Success Rate(%)	Fold	Success Rate(%)	Fold	Success Rate(%)
1	100.0	10	66.7	19	81.5
2	100.0	11	87.5	20	75.0
3	80.0	12	52.6	21	87.5
4	87.5	13	100.0	22	91.0
5	100.0	14	75.0	23	100.0
6	77.8	15	85.7	24	100.0
7	70.5	16	85.4	25	75.0
8	75.0	17	75.0	26	66.7
9	84.6	18	84.6	27	100.0
Overall Success Rate					81.2

To further improve the classification performance, another experiment was done using OvO and 10-fold cross validation technique together; and the overall success rate was increased as expected. Here, proteins were classified according to their folds with an 87.7% success rate. But the computation time was increased for the average of ten runs for each of 27 folds because of using either OvO method or 10-fold cross validation technique. The results related to this experiment is shown in Table 5.6 and the comparison of the results are shown in Table 5.7.

Table 5.6 : Performance of GAL for 27-class fold classification problem by using 10-fold cross validation and OvO prediction method.

Folds	1	2	3	4	5	6	7	8	9	10
Success Rate(%)	91.5	93.0	91.3	91.3	87.1	85.3	83.4	88.4	91.4	74.3
Overall Success Rate										87.7

To determine the effectiveness of the features, we made some tests as Ding and Dubchak did [37]. Firstly we used only C (amino acid composition) attribute to be contained in the feature vectors. Then we appended S (predicted SS) attribute to C, so

Table 5.7 : Prediction methods used with GAL and the success rates related to them.

Test Methods	Success Rate (%)
Accuracy	44.1
10-fold CV + Accuracy	57.1
OvO + Sensitivity	81.2
10-fold CV + OvO +Sensitivity	87.7

we used C+S to be the elements of the feature vectors, progressively in the last test we used all six attributes to form the feature vectors. To select attributes with this strategy does not claim to give optimum results. Because we started to test with C attribute as in [37]. In the test process, iteration number was determined as 3000. The results are shown in Table 5.8.

Above, to determine the most significant attribute and decrease the dimension of the feature vector we had considered the feature blocks including 20 or 21 features (Table 5.8). Here, we individually considered each of the 125 features with dynamic programming and we calculated the divergence values of each one. So we put in order 125 features according to their significance. Then we classified the proteins using the best 30, 40, 50 and 60 features with GAL. We did not run the classifier for more than 60 features because as seen from Table 5.9 we got better classification performance (81.5%) than that of Table 5.8 (81.2%). Test results related to divergence analysis are shown in Table 5.9.

Table 5.8 : True positive rates for each individual fold and overall success rates for the OvO method using GAL. Dimension of the feature vector is incremented gradually by including the six attributes with the order 'C', 'S', 'H', 'V', 'P' and 'Z'.

Fold no	C	CS	CSH	CSHV	CSHVP	CSHVPZ
1	83.3	100.0	100.0	100.0	100.0	100.0
2	77.8	88.9	88.9	100.0	100.0	100.0
3	60.0	75.0	75.0	85.0	85.0	80.0
4	87.5	87.5	100.0	100.0	100.0	87.5
5	100.0	100.0	100.0	100.0	100.0	100.0
6	66.7	66.7	66.7	55.6	55.6	77.8
7	65.9	65.9	72.7	68.2	70.5	70.5
8	50.0	66.7	66.7	75.0	75.0	75.0
9	92.3	76.9	76.9	76.9	84.6	84.6
10	66.7	66.7	83.3	83.3	83.3	66.7
11	62.5	75.0	87.5	100.0	87.5	87.5
12	36.8	52.6	47.4	47.4	47.4	52.6
13	75.0	75.0	100.0	100.0	100.0	100.0
14	75.0	100.0	100.0	100.0	100.0	75.0
15	85.7	71.4	85.7	85.7	71.4	85.7
16	79.2	83.3	83.3	83.3	87.5	85.4
17	66.7	83.3	83.3	91.7	83.3	75.0
18	61.5	69.2	69.2	76.9	69.2	84.6
19	66.7	59.3	59.3	66.7	51.9	81.5
20	83.3	75.0	75.0	66.7	75.0	75.0
21	62.5	75.0	87.5	87.5	87.5	87.5
22	91.7	91.7	91.7	83.3	91.7	91.7
23	85.7	85.7	100.0	100.0	100.0	100.0
24	75.0	100.0	100.0	100.0	100.0	100.0
25	50.0	75.0	75.0	75.0	87.5	75.0
26	51.9	59.3	59.3	55.6	70.4	66.7
27	100.0	92.6	92.6	88.9	96.3	100.0
Success Rate(%)	71.3	75.2	77.5	78.1	79.4	81.2

Table 5.9 : Performance of the GAL classifier using different dimensional feature vectors formed by divergence analysis.

Fold No	Dim = 30	Dim = 40	Dim = 50	Dim = 60
1	100.0	100.0	100.0	100.0
2	100.0	100.0	100.0	100.0
3	85.0	95.0	90.0	85.0
4	100.0	100.0	87.5	87.5
5	100.0	100.0	100.0	100.0
6	77.8	66.7	77.8	77.8
7	70.5	68.2	70.5	68.2
8	75.0	75.0	75.0	83.3
9	92.3	92.3	92.3	92.3
10	83.3	83.3	100.0	83.3
11	87.5	87.5	87.5	100.0
12	68.4	57.9	57.9	63.2
13	100.0	100.0	100.0	100.0
14	75.0	50.0	75.0	75.0
15	71.4	57.1	71.4	85.7
16	87.5	85.4	81.3	75.0
17	91.7	83.3	91.7	100.0
18	76.9	69.2	76.9	76.9
19	74.1	81.5	70.4	74.1
20	83.3	83.3	75.0	75.0
21	87.5	87.5	75.0	75.0
22	83.3	91.7	100.0	100.0
23	71.4	85.7	100.0	100.0
24	75.0	75.0	100.0	75.0
25	75.0	75.0	100.0	75.0
26	59.3	55.6	55.6	59.3
27	88.9	81.5	96.3	100.0
Success Rate(%)	80.7	79.1	80.9	81.5

5.2.3 Protein classification results by using SOM

In this section experiments were done firstly on a dataset including 125 dimensional 120 proteins which belong to three different folds, namely "Flavodoxin-like", "Ribonuclease H-like motif" and "TIMbeta/alpha-barrel". We used these three folds to compare the performance of SOM with the performance of SOM-SD in [90]. 10-fold cross-validation is implemented to classify protein folds in SCOP. By using 10-fold cross-validation, the datasets are partitioned into 10 sets having 12 samples in each. So, each time 108 proteins and 12 proteins were used for training data and test data, respectively. Here, the network was trained on 9×9 neurons with a neighborhood spread $\sigma = 1$, considering learning rate $\eta = 0.5$ and $\lambda = 500$ iterations. After the training and test processes the accuracy rate was calculated using (5.3). Performance was calculated as the average of 10 sets' performance.

We compared our results with the results in [90]. Both works basically use SOM but apply it in different ways. [90] uses SOM-SD but we use classical SOM. The difference between these methods is in the used features' types. SOM uses 125 features obtained from six attributes and the dimension of the data is fixed (125 dimensional data) but SOM-SD in [90] uses a new data type called PGI which includes variable number of features. In that work features are directions of the SSs in the protein so the dimension of the data in [90] is variable, because the number of SSs in the proteins are variable. So these two methods are applied differently to proteins. For classification of proteins SOM and SOM-SD use 120 and 45 proteins belonging to same three folds, respectively. Comparison results related to these methods are shown in Table 5.10. According to this table, SOM has better classification performance (93.3%) compared to SOM-SD (86.4%) although using less nodes.

Table 5.10 : Comparison results in terms of number of nodes, number of used proteins and accuracy rate for the proposed SOM and SOM-SD classifiers.

Methods	Number of Nodes	Number of Proteins	Accuracy Rate(%)
SOM	9×9	120	93.3
SOM-SD	200×200	45	86.4

Then, SOM is used to classify the 27 well-known SCOP folds. For 27-class problem, SOM was trained on 18×18 neurons with a neighborhood spread $\sigma = 1$, considering

learning rate $\eta = 0.5$ and $\lambda = 3000$ iterations. To calculate the classification performance we used sensitivity (true positive rate). While we test using OvO we used sensitivity to calculate success rate for individual fold (5.1) and we used generalized sensitivity for 27 folds to calculate overall success rate (5.2). Table 5.11 shows SOM's statistical performance values related to each fold. Here, as previous, it is difficult to classify the proteins of the 12th fold, they are classified with low success rate; but the proteins of the first, 13th and 27th folds are classified perfectly. According to this table, SOM is not as good as GAL for 27-class protein fold classification, but it has a success rate comparable with that of existing methods in literature.

Table 5.11 : Performance of 18×18 SOM with OvO for 27-class fold classification problem.

Fold	Success Rate(%)	Fold	Success Rate(%)	Fold	Success Rate(%)
1	100.0	10	50.0	19	63.0
2	88.9	11	75.0	20	66.7
3	80.0	12	52.6	21	87.5
4	87.5	13	100.0	22	75.0
5	100.0	14	75.0	23	71.4
6	66.7	15	85.7	24	75.0
7	72.7	16	72.9	25	62.5
8	58.3	17	66.7	26	55.6
9	84.6	18	61.5	27	100.0
Overall Success Rate					73.4

5.3 Protein Classification Results using Secondary Structures

5.3.1 Protein classification results by using SOM-SD

This section shows preliminary results obtained by SOM-SD using a set of proteins represented by PGI as input. In particular, for each SS, only its direction is considered. The dataset is composed of 45 proteins classified by SCOP as belonging to the class "Alpha and beta proteins (α/β)". Three folds have been considered, namely "Flavodoxin-like", "Ribonuclease H-like motif" and "TIMbeta/alpha-barrel", and for

each fold 15 proteins have been chosen. The task consists of grouping proteins belonging to the same fold.

The network performances are reported in terms of clustering performance, classification performance and retrieval performance. The clustering performance is a measure on how well clusters are formed. The classification performance is a measure on how well the clustering corresponds to the desired clustering by comparing the target values of vertices mapped to the same location. The retrieval performance is a measure of confidence of the clustering result, so that if there are many vertices with different targets mapped to the same location, the confidence in the clustering is low, hence producing a low retrieval performance. SOM-SD was trained on 200×200 neurons (output nodes) with a neighborhood spread of $\sigma = 60$, considering different learning rates $\eta = [1\ 1.25\ 1.5]$ and different iterations $\lambda = [40\ 60\ 80]$. For each test the average performance over 50 runs is reported. The dataset for each test was composed by randomly picking the 70% of the patterns for the training phase and the remaining 30% for the testing phase.

The test has been conducted considering the whole protein as pattern, i.e., each protein is represented by a PGI. It can be observed in Table 5.12 that the results are quite good in term of clustering performance. Even though this measure does not take into account the desired clustering outcome, the result is supported by the good retrieval performance which reflects a reduced confusion in the mappings of each pattern. The classification performance, reflecting the performance with respect to the desired clustering outcome, shows less accurate results but with an interesting peak 86.42% [90].

Table 5.12 : Performance of a 200×200 SOM–SD for three-class fold classification problem.

	Test Set								
Learning Rate	1			1.25			1.5		
Iterations	40	60	80	40	60	80	40	60	80
Retrieval	74.39	81.67	79.63	92.72	92.35	93.40	92.36	92.34	93.85
Classification	75.58	76.37	77.64	85.11	84.14	84.17	85.03	85.78	86.42
Clustering	0.80	0.80	0.80	0.79	0.80	0.79	0.79	0.80	0.79

5.3.2 Protein motif retrieval by using GHT

In this section three group of tests were processed for structural block comparison using a few proteins in PDB. To calculate the error rate, relative error rate formula given below was used.

$$ErrorRate = 100 \times \frac{|MotifRP - CandidateRP|}{MotifRP} \tag{5.4}$$

First group experiments: Two examples of searching motifs composed of four and five SSs are discussed here. In the first, a motif composed of five SSs (3 α-helices and 2 β-strands) is selected randomly from the protein 7FAB and is searched in the same protein. The protein under analysis, 7FAB, contains 46 SSs (9 α-helices and 37 β-strands). To apply SSS method, a mapping rule is figured out using ρ and θ values for each SS in the motif. In this experiment the number of SSs in the motif is five so, the expected number of votes in this method is five. To apply the SSC method, firstly comparison parameters (Md, Ad and ϕ) are determined related to motif couples and the mapping rule is figured out using ρ and θ values for each couple in the motif. Then, motif couples and protein couples are compared and if there is a correspondence a vote is given to the candidate RP. In this method the expected number of votes is the number of motif couples. In this situation, it is 10. For the SST method, motif triangles are compared to protein triangles using comparison parameters (edge values of the motif triangles) and if there is a correspondence a vote is given to the candidate RP. In this method expected number of votes is the number of motif triangles, so it is 10. Table 5.13 summarizes the performances in terms of the precision in locating and computation time. Second column represents the motif RP, third one represents the candidate RP location and the fourth one represents the displacement of the RP location with respect to the position of motif center of gravity and the last column is the searching time. It can be seen from Table 5.13 that SSC and SST have better computation time compared to SSS; but SST has the best retrieval performance compared to others.

The second one relates a well-known motif: The Greek Key which is formed by just four β-strands. The protein under analysis is 1FNB containing 22 SSs (9 α-helices and 13 β-strands). Table 5.14 summarizes the performances of these three methods.

Table 5.13 : Performances of searching five-SSs motif in 7FAB protein.

Methods	Motif RP	Candidate RP	Error Rate(%)	Search Time
SSS	[-17.59 9.51 15.21]	[-17.56 9.46 15.17]	0.28	108sec
SSC	[-17.48 9.17 15.48]	[-17.40 9.14 15.48]	0.34	42.51sec
SST	[-17.48 9.17 15.48]	[-17.48 9.17 15.48]	0.00	48.89sec

Table 5.14 : Performances of searching four-SSs motif in 1FNB protein.

Methods	Motif RP	Candidate RP	Error Rate(%)	Search Time
SSS	[31.33 1.14 12.01]	[31.41 1.16 11.94]	0.32	35.2sec
SSC	[31.38 1.08 11.69]	[31.33 1.08 11.79]	0.33	3.86sec
SST	[31.38 1.08 11.69]	[31.38 1.08 11.69]	0.00	5.76sec

For the first group experiments, tables show that the SST strategy has the best precision. The worst case is the SSS strategy mainly for the computation time, certainly because of the cumbersome mapping rule and cumbersome detection of the pick in the circles intersection.

Second group experiments: Here, SSC and SST methods were processed. Two sets of experiments were done. In the first set of experiments, 4GCR, 1FNB and 7FAB proteins containing 18, 22 and 46 SSs respectively were used. Three motifs composed of four SSs were selected randomly from these three proteins. First motif contains the 6th, 9th, 13rd, 17th SSs of 4GCR protein, second motif contains the 2nd, 8th, 15th 20th SSs of 1FNB protein and the third motif contains the 9th, 17th, 32nd, 40th SSs of 7FAB protein. Here SSC and SST methods are tested, and Table 5.15 and 5.16 show the results related to these tests.

In the second set of these experiments, different proteins and the motifs formed by five SSs were used. These proteins are 2Z9B, 3C94 and 3DHP containing 16, 37 and 44 SSs respectively. Three motifs composed of five SSs were selected randomly from these three proteins. First motif contains the 3rd, 5th, 7th, 12th, 15th SSs of 2Z9B protein, second motif contains the 10th, 19th, 21st, 27th, 32nd SSs of 3C94 protein

Table 5.15 : Results for searching the motif formed by four SSs in the 4GCR, 1FNB and 7FAB proteins by using SSC.

PDB ID	Motif RP	Candidate RP	#Max votes	Search Time
4GCR	[14.30 13.46 22.43]	[14.30 13.46 22.43]	6	0.004sec
1FNB	[23.55 8.10 15.57]	[23.55 8.10 15.57]	6	0.006sec
7FAB	[-30.17 14.39 13.23]	[-30.17 14.39 13.23]	6	0.009sec

Table 5.16 : Results for searching the motif formed by four SSs in the 4GCR, 1FNB and 7FAB proteins by using SST.

PDB ID	Motif RP	Candidate RP	#Max votes	Search Time
4GCR	[14.30 13.46 22.43]	[14.30 13.46 22.43]	4	0.007sec
1FNB	[23.55 8.10 15.57]	[23.55 8.10 15.57]	4	0.007sec
7FAB	[-30.17 14.39 13.23]	[-30.17 14.39 13.23]	4	0.011sec

and the third motif contains the 6th, 11th, 22nd, 30th, 39th SSs of 3DHP protein. Here SSC and SST methods are tested, and Tables 5.17 and 5.18 show the results related to these tests.

In these two sets of tests, the RP locations were determined with precision and had exactly the expected number of votes/contributions. Moreover no spurious peaks have been detected, and no displacement from the true RP location could be measured. The motif location perfectly coincides with the true RP location. The given computation times in the tables are related to a desktop computer with a processor Intel Core 2 Duo 6600, 2.4 GHz, 2 GB RAM.

Table 5.17 : Results for searching the motif formed by five SSs in the 2Z9B, 3C94 and 3DHP proteins by using SSC.

PDB ID	Motif RP	Candidate RP	#Max votes	Search Time
2Z9B	[7.38 26.97 6.14]	[7.38 26.97 6.14]	10	0.006sec
3C94	[14.03 28.67 26.88]	[14.03 28.67 26.88]	10	0.011sec
3DHP	[3.62 55.52 19.69]	[3.62 55.52 19.69]	10	0.008sec

Table 5.18 : Results for searching the motif formed by five SSs in the 2Z9B, 3C94 and 3DHP proteins by using SST.

PDB ID	Motif RP	Candidate RP	#Max votes	Search Time
2Z9B	[7.38 26.97 6.14]	[7.38 26.97 6.14]	10	0.007sec
3C94	[14.03 28.67 26.88]	[14.03 28.67 26.88]	10	0.011sec
3DHP	[3.62 55.52 19.69]	[3.62 55.52 19.69]	10	0.012sec

Third group of experiments: Here, a set of proteins (see Table 5.2) has been randomly selected among the PDB structures having a number of N_i of SSs ranging from 14 to 46 (a number of residue from 174 to 496); in Table 5.2 the selected set is detailed.

All possible structural blocks with m equal to three, four and five, have been retrieved for the SSC and SST approaches. Table 5.19 reports the number of experiments for the SSC and SST cases $\sum_{i=1}^{M} C(N_i, m)$, M is the number of the proteins selected from PDB, (column two: $C(N_i, m)$) and the cumulative and average time performances.

Table 5.19 : Performances and protein parameters for the tested set. Time ranges under the columns 3-4 are highlighted within square brackets.

Number of motif SSs: m	Total number of motifs	Average search time and range per motif for SSC (msec)	Average search time and range per motif for SST (msec)
3	105971	1.1 [0.6-1.5]	7.3 [0.9-11.7]
4	918470	1.4 [0.5-1.8]	11.2 [1.2-16.9]
5	6455009	1.7 [0.5-2.2]	17.3 [1.4-24.4]

In all about 7.5 million cases, the matching of candidate motifs with the RT tuples has been verified with a tolerance in the comparison parameters of $\varepsilon = 1\%$. In all cases, the collected RP locations had exactly the expected number of votes. Moreover, no spurious peaks have been detected for the SSC and SST cases. And, no displacement from the true RP position has been measured. The motif location perfectly coincided to the detected RP location.

6. CONCLUSIONS

Proteins are the most important macromolecules because they form much of the functional and structural machinery in every cell in all organisms. Function of the protein is determined by its spatial structure so that it is important to learn structure-function relationship in the protein universe by comparing their structures and retrieving similar models. In this thesis we focused on two problems related to proteins; protein fold classification and motif retrieval (structural block comparison) by using primary and secondary protein structures. Primary protein structures were used with GAL and SOM to classify the proteins according to their folds. Secondary protein structures were used with SOM-SD to classify the protein folds and with GHT to compare structural blocks for motif retrieval. Shortly, we used GAL, SOM and SOM-SD to classify the proteins, and used GHT to compare structural blocks.

Classifying the proteins according to their folds is one of the important study areas of the molecular biology. We can say that the proteins having the same structure perform the same function. So, the classification of proteins is important. In this thesis primarily protein fold classification problem is handled. To classify the proteins neural network based three methods were used; GAL, SOM and SOM-SD.

Firstly, GAL network was used to classify the protein folds. GAL is an incremental neural network for supervised learning, and determines the number of nodes during training if need arises. The network grows when it learns and shrinks when it forgets. GAL represents the distribution of feature vectors according to the minimum distance measure. Computational loads of training and classification processes of GAL are rather low. Moreover, there is not any parameter to be determined before the training. For application of GAL method to proteins, features extracted from the sequences were used. These features related to primary protein structures represent amino acids' physicochemical properties, namely amino acid composition, predicted SS, hydrophobicity, normalized van der Waals volume, polarity and polarizability. Of these attributes, amino acid composition has 20 dimensions and the other five attributes

have 21 dimensions. Thus, in total a feature vector combining six attributes has 125 dimensions. The dataset used with GAL was derived from SCOP by Ding and Dubchak [37]. This dataset, including 27 folds, contains 694 proteins having less than 35 percent sequence identity with each other. 311 of these proteins were used for training and the remaining 383 proteins were used for test. We do not approve the separation of dataset in this way, but after Ding and Dubchak, in a series of studies, this dataset was used like that. We had to use the same training and test sets to compare our results with those in the literature. However, by using K-fold cross validation, we achieved more realistic performance. The algorithm was tested with 3000 iterations. For the classification, GAL produced a network including 242 nodes and the proteins were classified in 4.32 seconds with a 44.1% success rate. Due to the poor success rate and imbalanced data 10-fold cross validation technique was used with GAL and the proteins were classified with a 57.1% success rate. To increase the classification performance OvO method was employed with GAL. With this method 27-class fold classification problem was transformed to two-class classification problem, so 27 binary classifiers were formed and the dataset was tested with these classifiers. Test results showed that the proteins were classified with a 81.2% success rate. To further increase the classification performance GAL network was tested with 10-fold cross validation and OvO techniques together, and this method achieved 87.7% success rate.

To determine the effectiveness of the attributes we made some tests using GAL. Firstly we used only C (amino acid composition) attribute to be contained in the feature vectors. Then we appended S (predicted SS) attribute to C, so we used C+S to be the elements of the feature vectors, progressively in the last set we used all six attributes and we tested by using GAL. The test results showed that the most important attribute is the amino acid composition. This attribute has a good performance (71.3%) even tested alone.

To decrease the feature vector dimension without changing the success rate divergence analysis was applied. This analysis calculates divergence values of the features and put them in order according to their importance. Here, after divergence analysis, the most significant 30, 40, 50 and 60 features were determined, and they were presented to GAL network. For the most significant 60 features, we obtained 81.5% classification performance, so we did not test anymore. As a result, using this method we decreased

the feature vector dimension from 125 to 60 without decreasing success rate, and so we reduced the computational load.

To classify the proteins according to their folds, secondly, SOM was used. Kohonen's SOM is a type of unsupervised learning. It uses competitive learning algorithm. In this algorithm the network neurons compete to be activated and eventually only one neuron wins the race. The goal is to train the network so that nearby outputs correspond to the nearby inputs, and to project the high dimensional data onto low dimensional map in an adaptive way. In the tests of this part we used two-dimensional rectangular grid of nodes. The dataset used with GAL was also used with SOM. Here, primarily three-fold, namely "Flavodoxin-like", "Ribonuclease H-likemotif" and "TIM beta/alpha-barrel" from α/β structural class, and then 27-fold classification problems were handled. The network was trained on 9×9 neurons with a neighborhood spread $\sigma = 1$, considering learning rate $\eta = 0.5$ and $\lambda = 500$ iterations for three-fold classification problem, and 18×18 neurons with a neighborhood spread $\sigma = 1$, considering learning rate $\eta = 0.5$ and $\lambda = 3000$ iterations for 27-fold classification problem. To calculate the classification performance sensitivity (true positive rate) was used. As a result of tests, the protein fold classification problem was solved with 93.3% and 73.4% success rates for three-fold and 27-fold cases, with SOM by using OvO.

Thirdly, SOM-SD was used for protein fold classification problem. This method differs from classic SOM in the way of used data. While applying SOM-SD a new data structure called PGI derived from secondary protein structures was used. Here, every protein was shown using PGI representation. In this representation, the chain sequence of SSs in the protein is recorded as a list which is mapped on a sphere surface and a directed graph is obtained. Nodes of the graph include the orientation information related to corresponding SS. The method was tested on the same three folds as in the SOM. These folds include 45 proteins (15 proteins in each fold). According to test results these three folds were classified with a 86.4% success rate.

When we consider all the test results, we can say that GAL network outperforms SOM for 27-class fold classification problem, and SOM outperforms SOM-SD for three-class fold classification problem. Besides, these methods can successfully compete with other methods in the literature.

In this thesis besides fold classification problem, we dealt with motif retrieval by structural block comparison. To do that GHT-based three methods (SSS, SSC and SST) were used. SSS uses the single SS, SSC uses the SS couple co-occurrences and SST uses the SS triplets. For each of the three methods a mapping rule is figured out to determine the location of reference point of the motif. For SSS this mapping rule is applied to all SSs in the motif; for SSC, this mapping rule is applied if there is a correspondence between motif couples and protein couples and a vote is given to this point. And as in the SSC, for SST, the mapping rule is applied if there is correspondence between motif triangles and protein triangles. To test these methods three group tests were processed. In the first group experiments, two motifs formed by four and five SSs were selected from 1FNB and 7FAB proteins, respectively, and SSS, SSC and SST methods were employed. Test results showed that SSC and SST have better computation time compared to SSS; but SST has the best retrieval performance compared to others. In the second group experiments, SSC and SST methods were tried for four-SS and five-SS motifs retrieval with six proteins selected randomly from PDB. According to test results, the RP locations related to the motifs were determined with precision and had exactly the expected number of votes. For the third group experiments, 20 proteins were selected from PDB randomly, and all possible three-SS, four-SS and five-SS motifs were searched by using SSC and SST methods for motif retrieval. Test results showed that the RP locations were determined precisely. Moreover, no spurious peaks have been detected. For the computation time SSC method has better performance compared to SST.

In future works, to increase the success rate of protein fold classification problem, improvements in the learning algorithms of GAL, SOM and SOM-SD networks will be searched. The methods can be tested on larger datasets. Moreover, the dimension of the feature vector derived from primary protein structures, in this thesis 125, can be reduced by using different dimension reduction techniques and the classifier can be tested with low-dimensional data for high performance and low computation time. For better classification performances, new feature extraction methods and new network structures can be searched.

REFERENCES

[1] **Hashemi, H.B.**, **Shakery, A. and Naeini, M.P.** (2009). Protein fold pattern recognition using Bayesian ensemble of RBF neural networks, *Proceedings of the International Conference of Soft Computing and Pattern Recognition*, pp.436–441.

[2] **Protein Data Bank**, `http://www.rcsb.org`, last access date: 07.06.2010.

[3] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2013). *Searching structural blocks by SS exhaustive matching*, Lecture Notes in Bioinformatics. Leif Peterson, Giuseppe Russo, Francesco Masulli (Eds.), pp. 57-69.

[4] **Cantoni, V.**, **Ferone, A. and Petrosino, A.** (2012). Protein motif retrieval through secondary structure spatial co-occurrences, *New Tools and Methods for Pattern Recognition in Complex Biological Systems*, *35*(5).

[5] **Kabsch, W. and Sander, C.** (1983). Dictionary of protein secondary structure: Pattern recognition of hydrogen bonded and geometrical features, *Biopolymers*, *22*(12), 2577–2637.

[6] **Heinig, M. and Frishman, D.** (2004). STRIDE: a web server for secondary structure assignment from known atomic coordinates of proteins, *Nucleic Acids Research*, *32*(2), W500–W502.

[7] **Eisenberg, D.** (2003). The discovery of alpha-helix and beta-sheet, the principal structural features of proteins, *Proc. of the National Academy of Sciences of the United States of America*, *100*(20), 11207–11210.

[8] **Murzin, A.G.**, **Brenner, S.E.**, **Hubbard, T. and Chothia, C.** (1995). SCOP: A structural classification of proteins database for the investigation of sequences and structures, *Journal of Molecular Biology*, *247*(4), 536–540.

[9] **Dubchak, I.**, **Muchnik, I.**, **Mayor, C.**, **Dralyuk, I. and Kim, S.H.** (1999). Recognition of a protein fold in the context of the structural classifications of proteins (SCOP) classification, *Proteins: Structure, Function and Bioinformatics*, *35*(4), 401–407.

[10] **Can, T. and Wang, Y.F.** (2003). A robust and efficient method for protein structure alignment based on local geometrical and biological features, *Proceedings of the IEEE Computer Society Conference on Bioinformatics*, pp.169–179.

[11] **Camoglu, O.**, **Kahveci, T. and Singh, A.** (2003). PSI: Indexing protein structures for fast similarity search, *Bioinformatics*, *19*(1), 81–83.

[12] **Chionh, C.H.**, **Huang, Z.**, **Tan, K.L. and Yao, Z.** (2003). Augmenting SSEs with structural properties for rapid protein structure comparison, *Proceedings of the Third IEEE Symposium on Bioinformatics and Bioengineering*, pp.341–348.

[13] **Chi, P.H.**, **Scott, G. and Shyu, C.R.** (2004). A fast protein structure retrieval system using image based distance matrices and multi dimensional index, *International Journal of Software Engineering and Knowledge Engineering, Special Issue on Software and Knowledge Engineering Support in Bioinformatics*, 522–532.

[14] **Zotenko, E.**, **Dogan, R.I.**, **Wilbur, W.J.**, **O'Leary, D.P. and Przytycka, T.M.** (2007). Structural footprinting in protein structure comparison: The impact of structural fragments, *BMC Structural Biology*, *7*(53).

[15] **Krissinel, E. and Henrick, K.** (2004). Secondary structure matching (SSM), a new tool for fast protein structure alignment in three dimensions, *International Union of Cyrstallography*, 2256–2268.

[16] **Cantoni, V. and Mattia, E.** (2012). Protein structure analysis through Hough transform and range tree, *New Tools and Methods for Pattern Recognition in Complex Biological Systems*, *35*(5), 39–46.

[17] **Kneller, D.G.**, **Cohen, F.E. and Langridge, R.** (1990). Improvements in protein secondary structure prediction by an enhanced neural network, *Journal of Molecular Biology*, *214*(1), 171–182.

[18] **Rost, B. and Sander, C.** (1993). Prediction of protein secondary structure at better than 70% accuracy, *Journal of Molecular Biology*, *232*(2), 584–599.

[19] **Frishman, D. and Argos, P.** (1997). Seventy-five percent accuracy in protein secondary structure prediction, *Proteins-Structure Function and Genetics*, *27*(3), 329–335.

[20] **Cuff, J.A.**, **Clamp, M.E.**, **Siddiqui, A.S.**, **Finlay, M. and Barton, G.J.** (1998). JPred: a consensus secondary structure prediction server., *Bioinformatics*, *14*(10), 892–893.

[21] **Jones, D.T.** (1999). Protein secondary structure prediction based on position-specific scoring matrices, *Journal of Molecular Biology*, *292*(2), 195–202.

[22] **Hua, S. and Sun, Z.** (2001). A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach, *Journal of Molecular Biology*, *308*(2), 397–407.

[23] **Fai, C.Y.**, **Hassan, R. and Mohamed, S.** (2012). Protein secondary structure prediction using optimal local protein structure and support vector machine, *International Journal of Bio-Science and Bio-Technology*, *4*(2), 35–43.

[24] **Liang, L.** (2012). Predicting the secondary structure of proteins using new ways of classification, *The 4th International Conference on Intelligent Human-Machine Systems and Cybernatics*, 212–215.

[25] **Klein, P. and Delisi, C.** (1986). Prediction of protein structural class from the amino acid sequence, *Biopolymers*, *25*(9), 1659–1672.

[26] **Zhang, C.T. and Chou, K.C.** (1992). An optimization approach to predicting protein structural class from amino acid composition, *Protein Science*, Cambridge University Press, 401–408.

[27] **Chou, K.C. and Zhang, C.T.** (1993). A new approach to predicting protein folding types, *Journal of Protein Chemistry*, *12*(2), 169–178.

[28] **Chou, K.C. and Zhang, C.T.** (1994). Predicting protein folding types by distance functions that make allowances for amino acid interactions., *Journal of Biological Chemistry*, *269*(35), 22014–22020.

[29] **Chou, K.C.** (1995). A novel approach to predicting protein structural classes in a (20-1)-D amino acid composition space, *Proteins*, *21*, 319–344.

[30] **Dubchak, I.**, **Muchnik, I.**, **Holbrook, S.R. and Kim, S.H.** (1995). Prediction of protein folding class using global description of amino acid sequence, *Proceedings of the National Academy of Sciences*, *92*(19), 8700–8704.

[31] **Cai, Y.D.**, **Liu, X.J.**, **Xu, X.B. and Chou, K.C.** (2002). Prediction of protein structural classes by support vector machines, *Computers & chemistry*, *26*(3), 293–296.

[32] **Luo, R.Y.**, **Feng, Z.P. and Liu, J.K.** (2002). Prediction of protein structural class by amino acid and polypeptide composition, *European Journal of Biochemistry*, *269*(17), 4219–4225.

[33] **Sun, X.D. and Huang, R.B.** (2006). Prediction of protein structural classes using support vector machine, *Amino Acids*, *30*, 469–475.

[34] **Chen, K.E.**, **Kurgan, L.A. and Ruan, J.** (2008). Prediction of protein structural class using novel evolutionary collocation-based sequence representation, *Journal of Computational Chemistry*, *29*(10), 1596–1604.

[35] **Reczko, M. and Bohr, H.** (1994). The DEF data base of sequence based protein fold class predictions., *Nucleic acids research*, *22*(17), 3616.

[36] **Edler, L.**, **Grassmann, J. and Suhai, S.** (2001). Role and results of statistical methods in protein fold class prediction, *Mathematical and Computer Modelling*, *33*(12), 1401–1417.

[37] **Ding, C.H.Q. and Dubchak, I.** (2001). Multi-class protein fold recognition problem using support vector machines and neural networks, *Bioinformatics*, *17*(4), 349–358.

[38] **Bologna, G. and Appel, R.D.** (2002). A comparison study on protein fold recognition, *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 5, IEEE, pp.2492–2496.

[39] **Bindewald, E.**, **Cestaro, A.**, **Hesser, J.**, **Heiler, M. and Tosatto, S.C.E.** (2003). MANIFOLD: protein fold recognition based on secondary structure, sequence similarity and enzyme classification, *Protein Engineering*, *16*(11), 785–789.

[40] **Markowetz, F.**, **Edler, L. and Vingron, M.** (2003). Support vector machines for protein fold class prediction, *Biometrical Journal*, *45*(3), 377–389.

[41] **Tan, A.C.**, **Gilbert, D. and Deville, Y.** (2003). Multi-class protein fold classification using a new ensemble machine learning approach, *Genome Informatics*, *14*, 206–217.

[42] **Huang, C.D.**, **Lin, C.T. and Pal, N.R.** (2003). Hierarchical learning architecture with automatic feature selection for multiclass protein fold classification, *NanoBioscience, IEEE Transactions on*, *2*(4), 221–232.

[43] **Igel, C.**, **Gebert, J. and Wiebringhaus, T.** (2004). Protein fold class prediction using neural networks with tailored early-stopping, *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 3, IEEE, pp.1693–1697.

[44] **Okun, O.** (2004). Protein fold recognition with k-local hyperplane distance nearest neighbor algorithm, *Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics, Pisa, Italy*, Citeseer, pp.51–57.

[45] **Shi, S.Y.M.**, **Suganthan, P.N. and Deb, K.** (2004). Multiclass protein fold recognition using multiobjective evolutionary algorithms, *Computational Intelligence in Bioinformatics and Computational Biology, 2004. CIBCB'04. Proceedings of the 2004 IEEE Symposium on*, IEEE, pp.61–66.

[46] **Chinnasamy, A.**, **Sung, W.K. and Mittal, A.** (2005). Protein structure and fold prediction using tree-augmented naive Bayesian classifier, *Journal of Bioinformatics and Computational Biology*, *3*(04), 803–819.

[47] **Huang, C.D.**, **Liang, S.F.**, **Lin, C.T. and Wu, R.C.** (2005). Machine learning with automatic feature selection for multi-class protein fold classification, *Journal of information science and engineering*, *21*(4), 711–720.

[48] **Nanni, L.** (2006). A novel ensemble of classifiers for protein fold recognition, *Neurocomputing*, *69*(16), 2434–2437.

[49] **Nanni, L.** (2006). Ensemble of classifiers for protein fold recognition, *Neurocomputing*, *69*(7), 850–853.

[50] **Shen, H.B. and Chou, K.C.** (2006). Ensemble classifier for protein fold pattern recognition, *Bioinformatics*, *22*(14), 1717–1722.

[51] **Chen, K. and Kurgan, L.** (2007). PFRES: protein fold classification by using evolutionary information and predicted secondary structure, *Bioinformatics*, *23*(21), 2843–2850.

[52] **Shamim, M.T.A.**, **Anwaruddin, M. and Nagarajaram, H.A.** (2007). Support Vector Machine-based classification of protein folds using the structural properties of amino acid residues and amino acid residue pairs, *Bioinformatics*, *23*(24), 3320–3327.

[53] **Abual-Rub, M.S. and Abdullah, R.** (2008). A survey of protein fold recognition algorithms, *Journal of Computer Science*, *4*(9), 768–776.

[54] **Guo, X. and Gao, X.** (2008). A novel hierarchical ensemble classifier for protein fold recognition, *Protein Engineering Design and Selection*, *21*(11), 659–664.

[55] **Krishnaraj, Y. and Reddy, C.K.** (2008). Boosting methods for protein fold recognition: an empirical comparison, *Bioinformatics and Biomedicine, 2008. BIBM'08. IEEE International Conference on*, IEEE, pp.393–396.

[56] **Damoulas, T. and Girolami, M.A.** (2008). Probabilistic multi-class multi-kernel learning: on protein fold recognition and remote homology detection, *Bioinformatics*, *24*(10), 1264–1270.

[57] **Shen, H.B. and Chou, K.C.** (2009). Predicting protein fold pattern with functional domain and sequential evolution information, *Journal of Theoretical Biology*, *256*(3), 441–446.

[58] **Chen, P.**, **Liu, C.**, **Burge, L.**, **Mahmood, M.**, **Southerland, W. and Gloster, C.** (2009). Protein fold classification with genetic algorithms and feature selection, *Journal of bioinformatics and computational biology*, *7*(05), 773–788.

[59] **Ghanty, P. and Nikhil, R.P.** (2009). Prediction of protein folds: Extraction of new features, dimensionality reduction, and fusion of heterogeneous classifiers, *IEEE Transactions on Nanobioscience*, *8*(1), 100–110.

[60] **Jazebi, S.**, **Tohidi, A. and Rahgozar, M.** (2009). Application of classifier fusion for protein fold recognition, *Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on*, volume 7, IEEE, pp.171–175.

[61] **Dehzangi, A.**, **Amnuaisuk, S.P. and Dehzangi, O.** (2010). Using random forest for protein fold prediction problem: An empirical study, *J. Inf. Sci. Eng.*, *26*(6), 1941–1956.

[62] **Dehzangi, A.**, **Amnuaisuk, S.P.**, **Manafi, M. and Safa, S.** (2010). Using rotation forest for protein fold prediction problem: An empirical study, *8th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, 217–227.

[63] **Valavanis, I.K.**, **Spyrou, G.M. and Nikita, K.S.** (2010). A comparative study of multi-classification methods for protein fold recognition, *International Journal of Computational Intelligence in Bioinformatics and Systems Biology*, *1*(3), 332–346.

[64] **Wang, R. and Gao, X.** (2010). A Two-Layer Learning Architecture for Multi-Class Protein Folds Classification, *Interdisciplinary Research and Applications in Bioinformatics, Computational Biology, and Environmental Sciences*, 39.

[65] **Dehzangi, A.and Karamizadeh, S.** (2011). Solving protein fold prediction problem using fusion of heterogeneous classifiers, *INFORMATION, An International Interdisciplinary Journal*, *14*(11), 3611–3622.

[66] **Kavousi, K.**, **Moshiri, B.**, **Sadeghi, M.**, **Araabi, B.N. and Moosavi-Movahedi, A.A.** (2011). A protein fold classifier formed by fusing different modes of pseudo amino acid composition via PSSM, *Computational Biology andChemistry*, *35*(1), 1–9.

[67] **Kavousi, K.**, **Sadeghi, M.**, **Moshiri, B. and Araabi, B. N.and Moosavi-Movahedi, A.A.** (2012). Evidence theoretic protein fold classification based on the concept of hyperfold, *Mathematical Biosciences*, *240*(2), 148–160.

[68] **Yang, T.**, **Kecman, V.**, **Cao, L.**, **Zhang, C. and Huang, J.Z.** (2011). Margin-based ensemble classifier for protein fold recognition, *Expert Systems with Applications*, *38*(10), 12348–12355.

[69] **Suvarnavani, K.**, **Rafiah, S.B. and Kamisetti, N.R.** (2011). Multiclass classification for protein fold prediction using Smote, *International Journal of Advanced Research in Computer Science and Software Engineering*, *2*(11), 290–296.

[70] **Chmielnicki, W. and Stapor, K.** (2012). A hybrid discriminative/generative approach to protein fold recognition, *Neurocomputing*, *75*(1), 194–198.

[71] **Bae, S.E.**, **Jung, S.**, **Ahn, I. and Son, H.S.** (2013). Protein fold classification with backbone torsional characters using multi-class linear discriminant analysis, *J Proteomics Bioinform*, *6*, 148–152.

[72] **Suryanto, C.H.**, **Hino, H. and Fukui, K.** (2013). Combination of multiple distance measures for protein fold classification, *Pattern Recognition (ACPR), 2013 2nd IAPR Asian Conference on*, IEEE, pp.440–445.

[73] **Lin, C.**, **Zou, Y.**, **Qin, J.**, **Liu, X.**, **Jiang, Y.**, **Ke, C. and Zou, Q.** (2013). Hierarchical classification of protein folds using a novel ensemble classifier, *PloS one*, *8*(2), e56499.

[74] **Aram, R.Z. and Charkari, N.M.** (2015). A two-layer classification framework for protein fold recognition, *Journal of Theoretical Biology*, *365*, 32–39.

[75] **Gatti, R.** (2010). Pattern recognition techniques applied to protein docking problems, *Ph.D. thesis*, Pavia University, Pavia, Italy.

[76] **Whitford, D.** (2005). *Proteins, structures and functionsl*, John Wiley and Sons, Ltd.

[77] **Gross, E. and Meienhofer, J.**, (1979). The Peptides analysis, synthesis, biology: Major methods of peptide bond formation, Academic Press, New York, 1. edition.

[78] **Jmol: An open source java viewer for chemical structures**, `http://www.jmol.org`, last access date: 07.06.2010.

[79] **Quaternary structure**, `http://themedicalbiochemistrypage.org/protein-structure.html`, last access date: 22.06.2015.

[80] **Horn, B.K.P.** (1984). Extended Gaussian images, *Proc. IEEE*, *72*(12), 1671–1686.

[81] **Kang, S.B. and Ikeuchi, K.** (1993). The complex EGI: a new representation for 3-D pose determination, *IEEE-T-PAMI*, *15*(7), 707–721.

[82] **Shum, H.**, **Hebert, M. and Ikeuchi, K.** (1996). On 3D shape similarity, *In Proceedings of the IEEE-CVPR*, 526–531.

[83] **Cohen, A.** *et al.* (1986). *Biomedical signal processing*, CRC press.

[84] **Ölmez, T. and Dokur, Z.** (2009). *Uzman sistemlerde örüntü tanıma: Yapay sinir ağları, genetik algoritmalar, bulanık mantık, makina öğrenmesi*, Lecture Notes, ITU.

[85] **Alpaydin, E.** (1994). GAL: Networks that grow when they learn and shrink when they forget, *Int. Journal of Recognition and Artificial Intelligence*, *8*, 391–414.

[86] **Kohonen, T.** (1982). Self-organized formation of topologically correct feature maps, *Biological Cybernatics*, *43*(1), 59–69.

[87] **Kohonen, T.** (1994). Self-organizing maps, *Springer Series in Information Science*.

[88] **Ozbudak, O. and Dokur, Z.** (2014). Protein fold classification using Kohonen's self-organizing maps, *Proceedings of International Work-Conference on Bioinformatics and Biomedical Engineering*, Granada, Spain.

[89] **Hagenbuchner, M.**, **Sperduti, A. and Tsoi, A.C.** (2003). A self-organizing map for adaptive processing of structured data, *Neural Networks, IEEE Transactions*, *14*(3), 491–505.

[90] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2012). Protein structural block representation and search through unsupervised NN, *Proceedings of the 22nd International Conference on Artificial Neural Networks*, Lausanne, Switzerland, pp.515–522.

[91] **Sperduti, A.** (2000). A tutorial on neurocomputing of structures, *Knowledge-Base Neurocomputing*, 117–152.

[92] **Hough, P.V.C.** (1962). Methods and means for recognizing complex patterns, *US Patent 3069654*.

[93] **Duda, R.O. and Hart, P.E.** (1972). Use of the Hough transformation to detect lines and curves in pictures, *Comm. ACM*, *15*(1), 11–15.

[94] **Wechsler, H. and Sklansky, J.** (1977). Automatic detection of ribs in chest radiographs, *Pattern Recognition*, *9*, 21–30.

[95] **Ballard, D.H.** (1981). Generalizing the Hough transform to detect arbitrary shapes, *Pattern Recognition*, *13*(2), 111–122.

[96] **Naik, S.K. and Murthy, C.A.** (2004). Hough transform for region extraction in color images, *In Proceedings of ICVGIP*, pp.252–257.

[97] **Kimme, C.**, **Ballard, D.H. and Sklansky, J.** (1975). Finding circles by an array of accumulators, *Communs Ass. Comput. Mach.*, *18*, 120–122.

[98] **Generalized Hough transform**, `http://http://www.cse.unr.edu/~bebis/CS791E/Notes/GeneralizedHough.pdf`, last access date: 07.07.2015.

[99] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2012). Protein structural motifs search in protein data base, *Proceedings of the 13th International Conference on Computer Systems and Technologies*, ACM, pp.275–281.

[100] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2012). Structural Analysis of Protein Secondary Structure by GHT, *Proceedings of the 21st International Conference on Pattern Recognition*, pp.1767–1770.

[101] **Ferone, A. and Ozbudak, O.**, (2013). Comparison of GHT-Based Approaches to Structural Motif Retrieval, New Trends in Image Analysis and Processing–ICIAP 2013, Springer, pp.356–362.

[102] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2012). Motif retrieval bu exhaustive matching and couple co-occurrences, *Proceedings of the 9th International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics*, Houston, Texas.

[103] **Dror, O.**, **Benyamini, H.**, **Nussinov, R. and Wolfson, H.** (2003). MASS: multiple structural alignment by secondary structures, *Bioinformatics*, *19*(1), 95–104.

[104] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2012). Search of protein structural blocks through secondary structure triplets, *3rd International Conference on Image Processing Theory, Tools and Applications*, Istanbul, pp.222–226.

[105] **Cantoni, V.**, **Ferone, A.**, **Ozbudak, O. and Petrosino, A.** (2013). Protein motifs retrieval by SS terns occurrences, *Pattern Recognition Letters*, *34*(5), 559–563.

[106] **Rose, P.W.**, **Bi, C.**, **Bluhm, W.F.**, **Christie, C.H.**, **Dimitropoulos, D.**, **Dutta, S.**, **Green, R.K.**, **Goodsell, D.S.**, **Prlić, A.**, **Quesada, M.**, **Quinn, G.B.**, **Ramos, A.G.**, **Westbrook, J.D.**, **Young, J. Zardecki, C.**, **Berman, H.M. and Bourne, P.E.** (2013). The RCSB Protein Data Bank: new resources for research and education, *Nucleic Acids Research*, *41*(D1), D475–D482.

[107] **Berman, H. M.and 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*(1), 235–242.

[108] **Chou, K.C. and Zhang, C.T.** (1995). Review: Prediction of protein structural classes, *Crit. Rev. Biochem. Mol. Biol.*, *30*, 275–349.

[109] **Cai, Y.D.** (2001). Is it a paradox or misinterpretation, *Proteins*, *43*, 336–338.

[110] **Zhou, G.P.** (1998). An intriguing controversy over protein structural class prediction, *Journal of Protein Chemistry*, *17*, 729–738.

[111] **Zhou, G.P. and Assa-Munt, N.** (2001). Some insights into protein structural class prediction, *Proteins*, *44*, 57–59.

[112] **Chou, K.C. and Maggiora, G.M.** (1998). Domain structural class prediction, *Protein Eng.*, *11*, 523–538.

[113] **Brown, M.P.S.**, **Grundy, W.N.**, **Lin, D.**, **Cristianini, N.**, **Sugnet, C.W.**, **Furey, T.S.**, **Ares, M.J. and Haussler, D.** (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines, *Proc. Natl. Acad. Sci. USA*, *97*(1), 262–267.

CURRICULUM VITAE

Name Surname: Ozlem Polat

Place and Date of Birth: Sivas 1981

Adress: Cumhuriyet University, Faculty of Technology, Central Campus, Sivas

E-Mail: ozlem.polat@cumhuriyet.edu.tr

B.Sc.: 2005, Yildiz Technical University, Faculty of Electrical & Electronics, Electronics and Communication Engineering

M.Sc.: 2009, Istanbul Technical University, Electronics & Communication Engineering, Electronics Engineering

Professional Experience and Rewards:

December 2006 - December 2013: Research Assistant, Istanbul Technical University.

December 2013 - present: Research Assistant, Cumhuriyet University.

PUBLICATIONS/PRESENTATIONS ON THE THESIS:

- Cantoni V., Ferone A., Petrosino A. and **Polat O.**, 2014. Pattern Recognition Tools for Proteomics, *The European Physical Journal Plus*, 129(130), 1-2, ISSN 2190-5444, DOI 10.1140/epjp/i2014-14130-3.

- **Ozbudak O.** and Dokur Z., 2014. Protein Fold Classification using Kohonen's Self-Organizing Map. *2nd International Work Conference on Bioinformatics and Biomedical Engineering*, April 7-9, Granada, Spain.

- **Ozbudak O.**, Dokur Z. and Cantoni, V., 2013. Protein Structural Block Comparison by Using Hough Transform. *The Journal of Electrical, Electronics, Computer and Biomedical Engineering*, 3(6), 91-97.

- Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2013. Searching Structural Blocks by SS Exhaustive Matching, *Lecture Notes in Bioinformatics*, 5381, Leif Peterson, Giuseppe Russo, Francesco Masulli (Eds.): CIBB 2012, LNBI 7845, 57-69. ©Springer-Verlag Berlin Heidelberg.

- Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2013. Protein Motifs Retrieval By SS Terns Occurrences, *Journal of Pattern Recognition Letters*, Elsevier, 34, 559-563, ISSN:0167-8655.

- Ferone A. and **Ozbudak O.**, 2013. Comparison of GHT-based Approaches to Structural Motif Retrieval. A. Petrosino, L. Maddalena, P. Pala (Eds.): *ICIAP 2013 Workshops*, LNCS 8158, 356–362, ©Springer-Verlag Berlin Heidelberg.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012: Motif Retrieval by Exhaustive Matching and Couple Co-occurrences", *9th International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics*, CIBB'12, July 12-14, Houston, Texas, ISBN:9788890643712.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012. Structural Biology and Pattern Recognition, *Convegno GIRPR'12*, May 21-23, Siena, Italy.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012. Search of Protein Structural Blocks through Secondary Structure Triplets, *3rd International Conference Image Processing Theory, Tools and Applications, IPTA'12*, October 15-18, Istanbul, Turkey, ISBN:978-1-4673-2584-4.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012. Structural Analysis of Protein Secondary Structure by GHT, *21st International Conference on Pattern Recognition, ICPR'12*, November 11-15, Japan, ISBN:978-4-9906441-1-6.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012. Structural Motif Representation and search through Unsupervised NN, *22nd International Conference on Artificial Neural Networks, ICANN'12*, September 11-14, 2012, Lausanne, Switzerland.

▪ Cantoni V., Ferone A., **Ozbudak O.** and Petrosino A., 2012. Protein Structural Motifs Search in Protein Data Bank, *International Conference on Computer Systems and Technologies, CompSysTech'12*, June 22-23, Ruse, Bulgaria, **Best Paper Award on session "Application Aspects**.

OTHER PUBLICATIONS/PRESENTATIONS:

▪ **Ozbudak O.**, Tukel M. and Seker S., 2010. Fast Gender Classification, *IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, December 28-29, Coimbatore, India.

▪ **Ozbudak O.**, Kirci M., Cakir Y. and Gunes E.O., 2010. Effects of the Facial and Racial Features on Gender Classification, *IEEE MELECON*'10, April 26-28, Valetta, Malta.

▪ Kocaman B., Kirci M., Gunes E.O., Cakir Y. and **Ozbudak O.**, 2009. On Ear Biometrics, *IEEE EUROCON'09*, May 18-23, St. Petersburg, Russia.