T.C.
BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI
YÜKSEK LİSANS PROGRAMI

PROTEİN HOMOLOJİ TESPİTİNDE BİR ÜST SINIFLANDIRMA YAKLAŞIMI

A DATA FUSION APPROACH IN PROTEIN HOMOLOGY DETECTION

AYDIN CAN POLATKAN

YÜKSEK LİSANS TEZİ

ANKARA 2007

PROTEİN HOMOLOJİ TESPİTİNDE
BİR ÜST SINIFLANDIRMA YAKLAŞIMI

A DATA FUSION APPROACH IN
PROTEIN HOMOLOGY DETECTION

AYDIN CAN POLATKAN

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
BİLGİSAYAR MÜHENDİSLİĞİ Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2007

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma, jürimiz tarafından

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI 'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Başkan (Danışman) : .....
(Prof. Dr. Hayri SEVER)

Üye : .....
(Prof. Dr. Mehmet Reşit TOLUN)

Üye : .....
(Yrd. Doç. Dr. Mustafa SERT)

ONAY

Bu tez 22/01/2007 tarihinde Enstitü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

...../...../.....
Prof.Dr. Emin AKATA
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

TEŞEKKÜRLER

Gerek bu çalışmanın gerçekleştirilmesinde gerekse akademik hayatta beni yönlendiren, getirdiği yeni yaklaşımlar ve bu yaklaşımları geliştirme konusunda sürekli fikir ve kaynak sağlayan biricik tez danışmanım, Prof. Dr. Hayri Sever'e gönülden teşekkür ederim.

Tez çalışmam da bana verdiği ilham ve her daim sağladığı destekten dolayı, hiç erinmeden ne zaman yardım istesem gerek Ankara'da gerekse Malatya'da imdadıma yetişen sevgili Yrd. Doç. Dr. Hasan Oğul'a gönülden teşekkür ederim.

Tüm hayatım boyunca olduğu gibi tez çalışmalarımda da sevgilerini ve desteklerini hiçbir zaman esirgemeyen sevgili babam Vahit Polatkan, annem Nursel Polatkan, ağabeylerim Selkan ve Kağan Polatkan'a gönülden teşekkür ederim.

ÖZET

PROTEİN HOMOLOJİ TESPİTİNDE BİR ÜST SINIFLANDIRMA YAKLAŞIMI

Aydın Can POLATKAN

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Hesaplamalı biyoloji alanında sınıflandırma problemleri için makine öğrenme teknikleri sıkça ve geniş şekilde kullanılmaktadır. Bu teknikler, girdi olarak sabit uzunluklu nitelik vektörleri istemektedir. Bilindiği üzere proteinler farklı uzunluklara sahip olduklarından dolayı, tüm protein dizilimlerini sabit sayıda nitelik ile göstermek gerekir.

Bu amaçla geliştirilen etkili yöntemlerden biri protein dizilimlerinin n-peptit birleşimleridir. Yöntem n uzunluktaki her alt dizginin dizilim içerisindeki görülme yüzdesini ifade eder. Alan karmaşıklığını azaltmak amacıyla, n'nin artan değerleri için, kullanılan aminoasit alfabesi, sonuç vektörün günümüz bellek kaynaklarıyla uyumlu olmasını sağlayacak şekilde düzenli olarak küçültülmüştür.

Kullanılan bu çözümde birleşime ait bütün özellik girdileri sadece bir sınıflandırıcıya toplu olarak verilmekteydi. Bu tezde, bu özellik girdileri n-peptit birleşimlere ve küçültülen amino asit alfabelerine göre farklı gruplara ayrılıp, farklı sınıflandırıcılara verilmiştir böylece soyutlanarak daraltılan arama uzayında, gezinen birden fazla tekniğe, bir üst sınıflandırma yaklaşımı denenmiştir. Amaç doğru şekilde yakınsanan ve bizi birbirinden farklı çözüm bölgelerine ulaştıran tekniklere üstsel sınıflandırma yaklaşımı ile daha iyi sonuçlar alabilmektir. Bu yaklaşımda farklı sınıflandırıcıların çıktı değerlerini değerlendirmek üzere ortalama alma, ağırlıklı ortalama alma ve öğrenme kümesinde en başarılı olanı seçme gibi değişik durumlar karşılaştırılmıştır.

Her bir yöntem hesaplamalı biyolojinin önemli ve güncel problemlerinden biri olan uzak homoloji tespiti üzerinde test edilmiş ve sonuçlar karşılaştırmalı olarak sunulmuştur.

Sonuçlara bakıldığında eğitim kümesinde en başarılı olan sınıflandırıcının sonucunun doğru kabul edildiği durumun diğerlerine göre daha etkili olduğu gözlenmiştir. Sonuçlar arasındaki istatistiksel anlamlılığı dikkatlice incelemek için tüm yöntemler arasında öğrenci T-testleri yapılmış ve testlerin sonuçları yorumlanmıştır. Denenen bütün üst sınıflandırma yaklaşımları yalnız bir sınıflandırıcı kullanılan duruma göre daha etkili bellek kullanımına sahiptir. Destek vektör makineleriyle test edilen bu üst sınıflandırma yaklaşımının sadece uzak homoloji tespitinde değil diğer sınıflandırma problemlerinde de başarılı olacağı düşünülmektedir.

Anahtar Sözcükler: Protein Homoloji Tespiti, N-peptit Birleşimler, Destek Vektör Makineleri, Sınıflandırma, Üst Sınıflandırma.

Danışman: Hayri SEVER, Prof. Dr., Çankaya Üniversitesi, Bilgisayar Mühendisliği Bölümü

ABSTRACT

A DATA FUSION APPROACH IN PROTEIN HOMOLOGY DETECTION

Aydın Can POLATKAN

Baskent University

Computer Engineering

Machine learning techniques are frequently and extensively used for classifying problems in the field of computational biology. These techniques require constant length feature vectors as inputs. As far as it is known that proteins are in different lengths, therefore all proteins are needed to be represented with a constant number of features.

One of the effective methods developed for this goal is n-peptite combinations of the protein strings. These methods are represented with the availability percentage of each of the n-length substrings inside the sequence. To reduce the space complexity, for increasing values of n, amino acid alphabet is reduced regularly for the resulting feature vectors to conform available memory resources today.

In this solution, all feature inputs were given to a single classifier. In this thesis, these feature inputs are classified into specific significant groups, according to the n-peptite compositions and reduced amino alphabets. These groups are given to several different classifiers to achieve a data fusion approach with a few techniques that are wandering in the narrowed search space by abstraction. Aim is to have better results with techniques that are converging in exact and leading to different regions of a solution. In that approach, to evaluate the output values of different classifiers, various cases like averaging, weighted averaging and choosing the most successful one in the training set are compared.

Each of these methods was tested on remote homology detection problem which is one of the major and actual problems of computational biology and results are presented relatively.

As the results are considered, the case in which the output of the most successful training set is granted, observed as the more accurate one. To explore the statistical significance of differences between results, paired samples T-tests were carried out between all methods. Furthermore, all data fusion approaches tested, through out the thesis has more efficient memory usage according to the single classifier case. The data fusion approach which has been tested with support vector machines is also thought to be efficient for not only protein homology detection problems but also other problems of classification.

Keywords: Protein Homology Detection, N-peptite Compositions, Support Vector Machines, Classification, Data Fusion.

Supervisor: Hayri SEVER, Prof. Dr., Çankaya University, Department of Computer Engineering

İÇİNDEKİLER

ŞEKİLLER

ÇİZELGELER

SİMGELER ve KISALTMALAR LİSTESİ

Kısaltma	Açıklama
A	Adenin
AORT	Ağırlıklı Ortalama
C	Citosin
ÇDH	Çoklu Dizin Hizalama
DNA	Deoksiribonükleik Asit
DVM	Destek Vektör Makinesi
G	Guanin
İDH	İkili Dizin Hizalama
MAX	Maksimum
MBE	Minimum Biricik Eşleşme
mRNA	Motor Ribonükleik Asit
ORT	Ortalama
OSA	Olasılıklı Sonek Ağaçları
PAM	Noktadan Kabul Mutasyon
PDH	Protein Data Bank
RNA	Ribonükleik Asit
ROC	Receiver Operating Characteristics
rRNA	Ribozomal Nükleik Asit
SCOP	Structural Classification of Proteins
SMM	Saklı Markov Modeli
T	Timin
tRNA	Transfer Ribonükleik Asit
U	Urasil

1 – GİRİŞ

1. Giriş

Yüksek çıktıya sahip genom sıralama projelerinin birçoğu veritabanlarında ham sıra verisinin yüksek oranlarda yığılmasıyla sonuçlanmıştır. Deneysel zorluklar ve proteinlerin yapısal ve fonksiyonel analizinde karşılaşılan engeller yüzünden son yıllarda bilinen protein dizileri ve deneysel olarak belirlenen yapıların arasındaki fark durmadan artmaktadır. Bu durum, protein dizi verisine sahip genel veritabanları üzerinde proteinlerin yapısı ve fonksiyonları üzerinde otomatik çıkarımlar yapan hesaplama araçlarının ve yöntemlerinin ortaya çıkmasına neden olmuştur.

1.1. Tezin Kapsamı ve Düzeni

Bu tez otomatik protein sınıflandırma sistemlerinde kullanılacak protein dizilerinin biri olan n-peptit birleşimlerini ele almaktadır. Tez kapsamında bu gösterim üzerinde bir üst sınıflandırma yaklaşımı denenmiştir. Bu yaklaşımda farklı sınıflandırıcıların çıktı değerleri değerlendirilmek üzere karşılaştırılmıştır. Yöntemler hesaplamalı biyolojinin önemli ve güncel problemlerinden biri olan uzak homoloji tespiti üzerinde uygulanmış, test edilmiş ve sonuçlar verilmiştir.

Bu bölümün bir sonraki kısmı tez genelinde çeşitli bölümlerde sık olarak değinilen ve bağlantılı biyoloji bilimine ait bazı kavramlara ve tanımlara giriş yapmaktadır. Bölüm üzerinde deneylerin yapıldığı veri kümelerini oluşturmak için kullanılan açık veritabanlarına giriş yaparak sonlanmaktadır.

İkinci bölüm üst sınıflandırma yaklaşımının, bilgi belge yönetimi üzerinde nasıl uygulandığını anlatmaktadır. Sınıflandırma algoritmaların, arama yöntemlerinin sonuçlarının birleştirilmesi, önerilen, kullanılan yapılar ve bu şekilde yapılan üst sınıflandırma yaklaşımı hakkında geniş bilgi verilmektedir. Bu tezin özünde protein

homoloji tespitinde kullanılan, üst sınıflandırma yöntemlerinin benzerlerinin bilgi belge yönetiminde kullanılan örnekleri verilmiştir.

Üçüncü bölüm makine öğrenmesi ve destek vektör makinelerinin anlatıldığı bölümdür. Bölüm içerisinde sırasıyla verilerin sayısallaştırılması, özellik seçimi, çıkarımı ve son olarak mevcut yöntemler tanıtılmış ve anlatılmıştır.

Dördüncü bölüm dizi gösterimlerinin tanımlandığı ve anlatıldığı bölümdür. N-peptit gösterimlerin ilk tanımlandığı yer bölüm 4.1'dir. Benzerlik tabanlı gösterimler bölüm 4.2'de tanımlanmaktadır. Bunların arasında bölüm 4.2 altında alt başlıklarla maksimum biricik eşleşme modeli ve ikili olasılıklı sonek ağaçları tanımlanmıştır. Dizi haritalamanın literatürdeki tarihsel görüngesini vermek amacıyla bu çalışma için özgün olmayan sıraya dizme tabanlı karşılaştırma teknikleri bölüm 4.2.1'de alt başlıklar ile anlatılmıştır.

Beşinci bölüm, biyolojinin önemli ve güncel problemlerinden biri olan uzak homoloji tespitini yapılan önceki çalışmalara da değinerek anlatmaktadır. Uzak homoloji tespiti için yapılan üst sınıflandırma yaklaşımları yine bu bölümde sistemler ve yöntemler başlığı altında anlatılmaktadır.

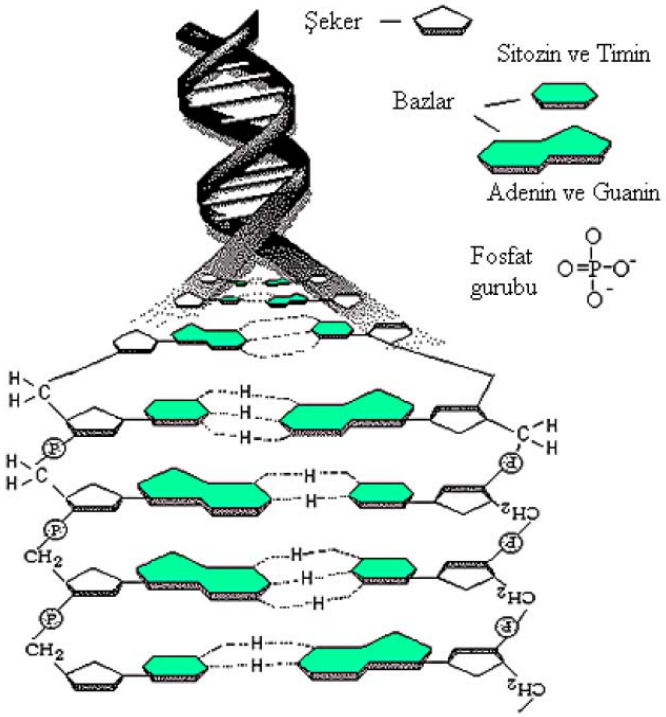
Son olarak altıncı bölümde tez içinde, ortak karşılaştırmalı veri kümeleri üzerinde yürütülen testler ve titiz analizler sonunda ortaya çıkan sonuçlar ve bu sonuçların ikili ve genel karşılaştırmaları açık ve anlaşılır bir şekilde, yorumlarıyla beraber sunulmuştur.

1.2. Gerekli Biyoloji Bilgisi

Nükleik asitler genetik bilginin depolanması ve ifade edilmesinden sorumlu moleküllerdir. Kimyasal olarak değerlendirildiğinde iki farklı çeşit nükleik asit mevcuttur. Bunlardan biri deoksiribonükleik asit olarak nitelendirilen DNA, diğeri ise ribonükleik asit olarak nitelendirilen RNA'dır. Her iki nükleik asit de yapılarından nükleoititler bulundurmaktadırlar. Makromoleküler yapıda şeker ve fosfat birimleri

fosfodiester bağı ile birbirine bağlanarak molekülün ana omurgasını oluşturur. Azotlu bazlar ise iki omurgayı bir arada tutmaktadır. Nükleik asitler birden fazla yapı taşının bir araya gelmesiyle oluşmaktadır. Bu yapı taşları, şekerler, pürin ve primidin bazları, nükleozitler, nükleotitler ve polinükleotitler olarak verilir.

Ebeveynlerimizden miras aldığımız ve çocuklarımıza verdiğimiz genetik bilgiler eksik oksijenli çekirdek asidi ya da diğer bir adıyla DNA olarak verilen uzun moleküller tarafından taşınmaktadır. DNA, iki uzun tele sahiptir, bu tellerden her biri kimyasal çekirdekler, fosfat, dioksiriboz şekerler ve nükleotidlerin bir dizi şeklinde birbirine bağlanması ile oluşmaktadır. DNA yapısında bulunan nükleotidler dört çeşittir, Adinin, Ganin, Sitisin, Timin sırasıyla A, G, S ve T olarak kısaltılır. Bir DNA molekülü, bir çift helis oluşturmak üzere birbirine anti paralel konumlanan iki telin birleşiminden oluştur. Bu adaptasyon katı temel eşleşme kurallarına sahiptir. Örneğin, A sadece T ile, G sadece C ile eşleşebilir. Bu nedenle, gerçekte her tel diğerinin tamamlayıcı dizidir. DNA helisinde bir tele kalıp diğerine ise kılavuz adı verilir. DNA molekülün görünümü Şekil 1.2.1'de gösterilmiştir.



Şekil 1.2.1 – DNA molekülü

Hücre içerisinde gerçekleşen birçok kimyasal reaksiyon, başlıca proteinlerin sonucudur. Herhangi bir canlı için, DNA molekülünün temel rolü, protein sentezini belirleyerek hücre içerisindeki faaliyetleri kontrol etmektir. Ancak, bir DNA doğrudan protein üretmez, bunun yerine sırayla protein sentezini kodlaması için RNA teli formunda bir kalıp üretir. Genelde, bilgi, DNA moleküllerinden proteinlere RNA molekülleri sayesinde gönderilir.

DNA molekülü içerisinde saklı bilginin proteine dönüştürülmesine transkripsiyon denilmektedir. Transkripsiyon gerçekleşirken, DNA çift sarmalı açılarak, sarmallardan biri kalıp görevini üstlenir ve bu sarmala anti paralel olarak RNA sentezi gerçekleştirilir. Diğer bir deyişle transkripsiyon, DNA molekülünden, RNA kalıbının üretilmesi olayıdır.

RNA'lar ribonükleotitlerin birbirine bağlanması ile meydana gelen tek zincirli nükleik asitlerdir. DNA ile kıyaslandıkları zaman boyları daha kısadır. Tüm hücrelerde bol miktarda bulunan RNA'nın üç türü vardır.

1. mRNA, diğer adı ile haberci RNA, DNA'da saklı genetik bilginin, protein yapısına aktarılmasında kalıp görevi yapan aracı moleküldür. Hücrede mevcut bütün RNA miktarının yaklaşık %5'ini oluşturmaktadır. DNA molekülünden aldığı genetik şifre, sentezlenecek proteinin aminoasit sırasını tayin eder. Her mRNA molekülü, DNA üzerinde bulunan belirli bir gen dizisine karşı tamamlayıcı özelliğe sahiptir.

2. tRNA, diğer adı ile transfer RNA, hücrede mevcut RNA'lar arasında en küçüğüdür. Tüm hücre RNA'sının %15'ini oluştururlar. Doğada yer alan 20 aminoasidin her biri için en az bir tRNA molekülü bulunmaktadır. Adaptörlük görevi yapan tRNA'lar bağlandıkları aminoasidi, mRNA üzerindeki şifreye göre polipeptit zincirine aktarırlar. Üç bazdan meydana gelen ve kodon adı verilen şifrenin tRNA üzerindeki tamlayıcılığına ise antikodon adı verilir.

Çizelge 1.2.1 – Aminoasitler ve Kısaltmaları

Aminoasit	Üç Harfli Kodu	Tek Harfli Kodu
polar olmayan aminoasitler (hydrophobic)		
Glycine	Gly	G
Alanine	Ala	A
Valine	Val	V
Leucine	Leu	L
Isoleucine	Ile	I
Methionine	Met	M
Phenylalanine	Phe	F
Trytophan	Trp	W
Proline	Pro	P
polar aminoasitler (hydrophilic)		
Serine	Ser	S
Threonine	Thr	T
Cysteine	Cys	C
Tyrosine	Tyr	Y
Asparagine	Asn	N
Glutamine	Gln	Q
elektrikle yüklü aminoasitler (negatif ve hydrophilic)		
Aspartic asit	Asp	D
Glutamic asit	Glu	E
elektrikle yüklü aminoasitler (positif ve hydrophilic)		
Lysine	Lys	K
Arginine	Arg	R
Histidine	His	H

3. rRNA, diğer adı ile ribozomal RNA olarak bilinir. Ribozom hücrede protein sentezinin yapıldığı yerlerdir. Ribozomal RNA'lar ribozomların ana yapı elementi olup, ribozom ağırlığının yaklaşık olarak %65'ini teşkil etmektedirler. rRNA'lar katalitik bir çevre oluşturmak ve tRNA'ların kendilerine bağlı aminoasitlerle beraber bu çevreye girebilmesi için sayısız ribozomsal proteinle birleştirilmiştir. Ribozomlar protein sentezinin tüm faaliyetlerini katalize eder.

Transkripsiyondan sonra ortaya çıkan RNA, DNA çiftinin kalıp teline tamamlayıcı, kalıp olmayan diğer teline ise eş olmaktadır. DNA molekülünün kalıp olmayan teli, kod teli olarak nitelendirilir çünkü bu teldeki tüm daha sonra mRNA sayesinde proteinlere dönüştürülür. Ancak U ile verilen Urasil, RNA yapısında T ile yani Timin ile değiştirilir. Protein sentezi RNA'lar tarafından yönlendirilir ve bu işleme çeviri adı verilmektedir. Bu işlem RNA'nın üç sınıfına da gereksinim duymaktadır, fakat kendine özgü aminoasitlerin tam eklenmesi için kullanılan kalıp mRNA'dır. Ökaryotlarda, mRNA dizisini, proteinin birincil dizisine dönüştürmek için mRNA çekirdekten dışarıya doğru sitoplâzmaya gönderilir. Daha sonra RNA, üç harften oluşan kodon serilerine çevrilir, her kodon özel bir aminoasit gösterir. Aminoasit listeleri ve kısaltmaları Çizelge 1.2.1'de verilmiştir.

Hücrenin yönetimi esnasında gerçekleşen bütün olaylarda, DNA molekülleri üzerindeki şifreler kullanılır. Sitoplazmaya bilgi aktarılır. Olaylarda bu bilgilere göre düzenlenir. DNA zinciri üzerinde arka arkaya dizilmiş her üç nükleotit, bir anlam ifade etmektedir. İşte DNA ve RNA moleküllerindeki bu üçlü dizilimlere (AAG, CTC vb.) kodon ya da şifre denilmektedir. DNA molekülünde dört adet nükleotit bulunduğu ve bunların bir kodon oluşturabilmek için üçer üçer gruplandırıldığı düşünüldüğünde, toplam 4^3 = 64 farklı kodon meydana gelmektedir. DNA molekülleri bütün mesaj ve emirleri işte bu 64 farklı şifreyi kullanarak vermektedir. Kodonlar genetik şifrenin en küçük birimleridir. RNA molekülündeki kodonlara antikodon denilmektedir. Doğada bulunan ve protein yapısına katılan 20 aminoasit kodonlar ile ifade edilmektedir. DNA üzerindeki kodonlar kullanılarak sitoplazmada protein sentezi yapılmaktadır. Protein sentezi için verilen zincirdeki kodonlar bir cümle gibidir, bu bilgiler şifreli olarak mRNA tarafından alınır. DNA'nın bir protein

için kaç nükleotit kullanacağı, ilgili proteinin kaç aminoasitten oluştuğuna bağlıdır. Proteinlerdeki aminoasit sayısı 9 ile 700 arasında değişebilmektedir, bu nedenle de mRNA'ların nükleotid ve kodon sayıları farklı olmaktadır.

Tüm olası kodon dizileri ve ilgili aminoasit gösterimleri Çizelge 1.2.2'de verilmiştir. Bazı aminoasitlerin birden fazla üçlü kodon tarafından kodlandığı deneysel olarak ispatlanmıştır, bu yüzden genetik kod bozulmaktadır. Başlangıç kodonu adıyla verilen özel bir kodon transkripsiyonun başlangıç noktasını belirtir, yine transkripsiyon özel bir kodon olan bitiş kodonu ile sonlandırılır.

Çizelge 1.2.2 – Genetik Kod

	Orta				
İlk	U	C	A	G	Son
U	Phe	Ser	Tyr	Cys	U
	Phe	Ser	Tyr	Cys	C
	Leu	Ser	Stop	Stop	A
	Leu	Ser	Stop	Trp	G
C	Leu	Pro	His	Arg	U
	Leu	Pro	His	Arg	C
	Leu	Pro	Gln	Arg	A
	Leu	Pro	Gln	Arg	G
A	Ile	Thr	Asn	Ser	U
	Ile	Thr	Asn	Ser	C
	Ile	Thr	Lys	Arg	A
	Met	Thr	Lys	Arg	G
G	Val	Ala	Asp	Gly	U
	Val	Ala	Asp	Gly	C
	Val	Ala	Glu	Gly	A
	Val	Ala	Glu	Gly	G

Kısaca özetlemek gerekirse, bilgi depo molekülü (DNA) içerdiği bilgiyi, transfer molekülü (RNA) ile fonksiyonel ve kodlamayan ürün olan proteine geçirir. Örneğin, aşağıdaki DNA dizisi verilsin,

AGTAATCTCGTTACT

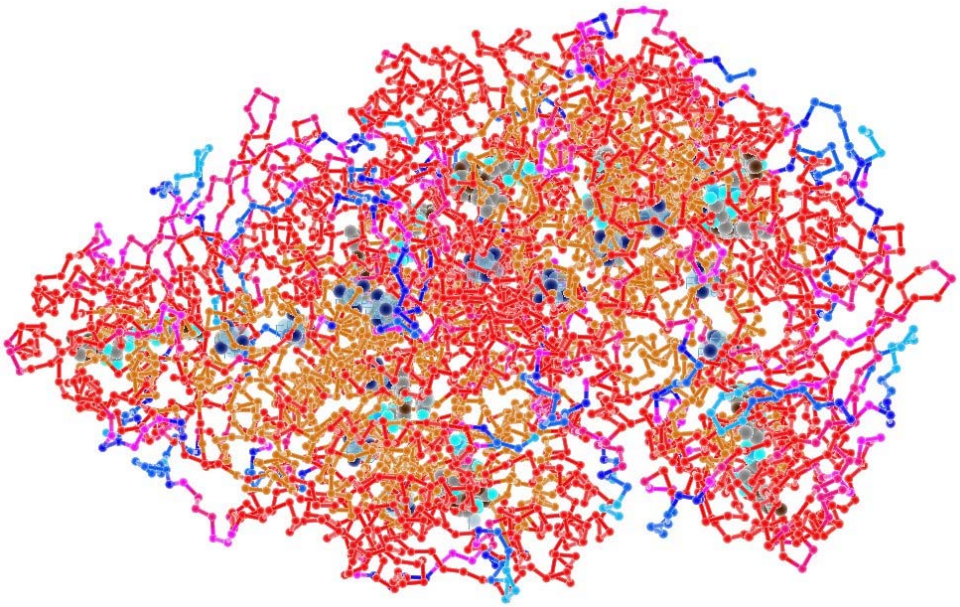
daha sonra RNA, DNA kalıbı ile aynı olacaktır, tek fark DNA'daki T'ler, U'lar ile yer değiştirecektir;

AGU AAU CUC GUU ACU

olarak ortaya çıkan RNA kodunun protein dizilimi ise aminoasitlerinin kısaltmalarının yerine konmasıyla aşağıdaki dizilim ortaya çıkacaktır.

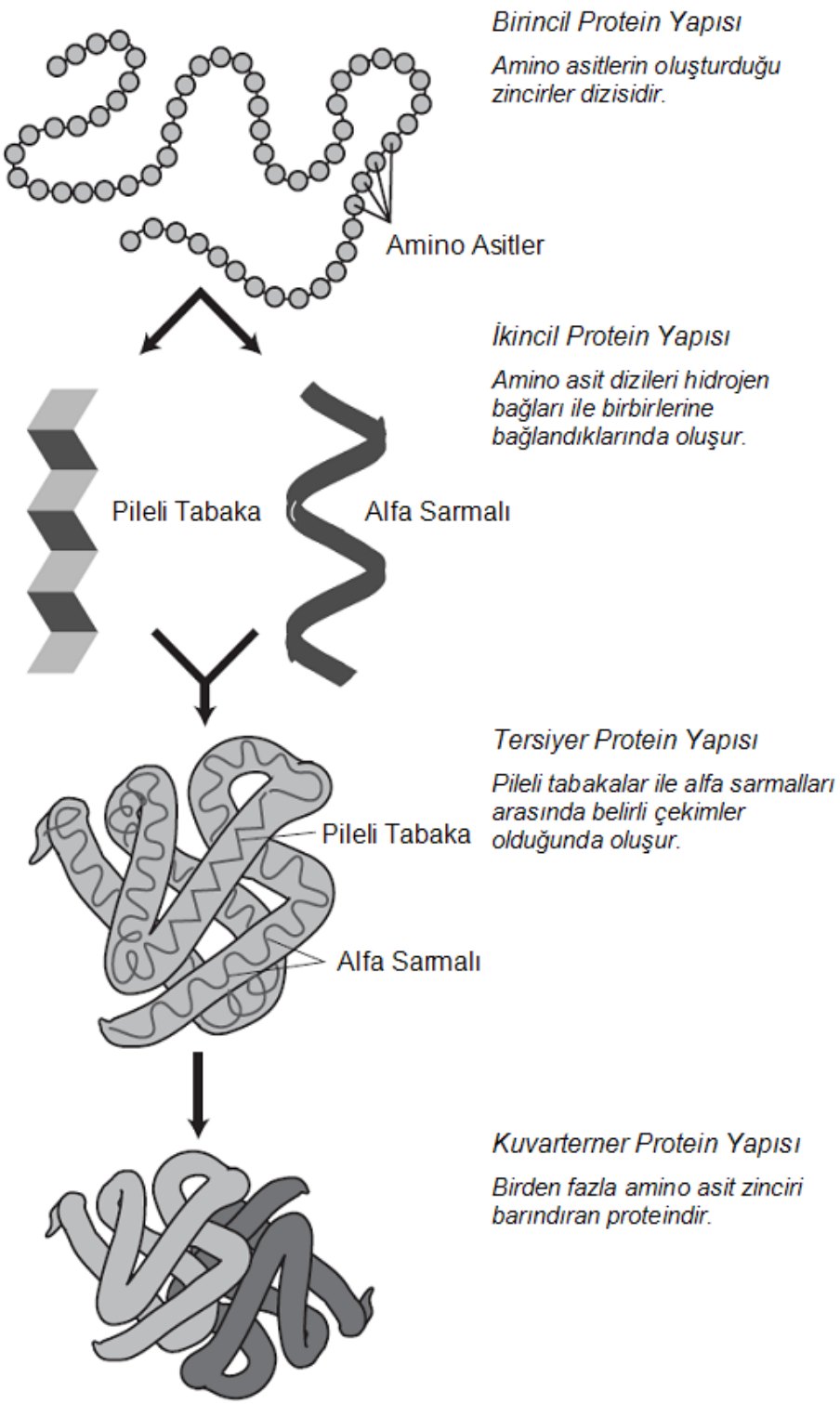
S N L V T.

Proteinler, doğada bulunan 300'den fazla aminoasitten 20 tanesinin birbirine peptit bağlarıyla bağlanarak oluşturduğu dev yapılı moleküllerdir. Üç boyutlu bir protein yapısı modeli Şekil 1.2.2'de gösterilmiştir.



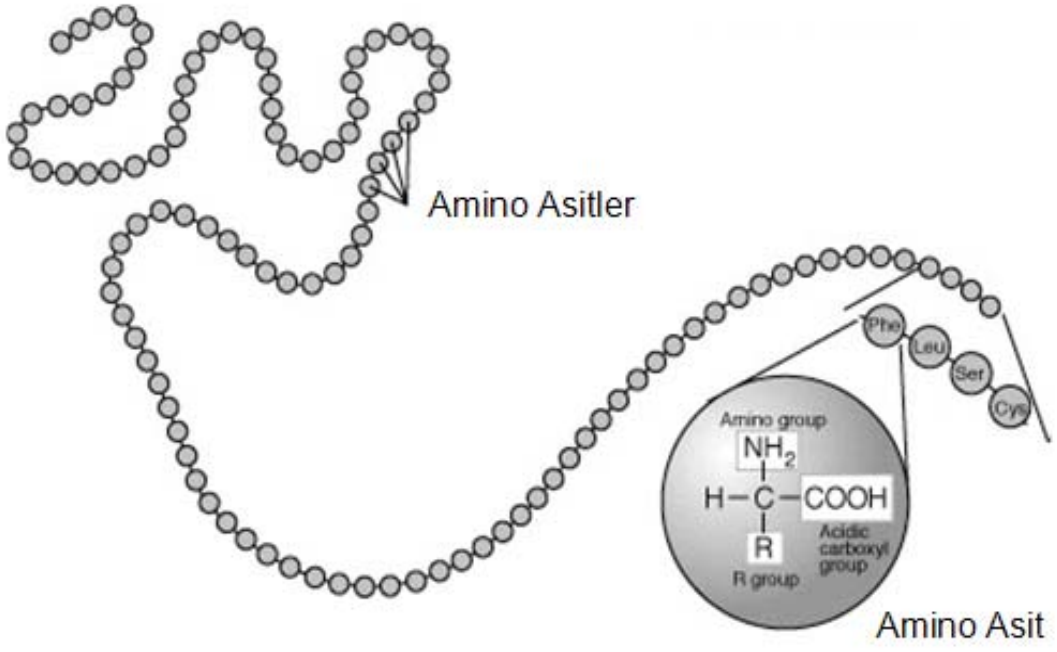
Şekil 1.2.2 – Üç boyutlu bir protein yapısı modeli

Protein yapısındaki aminoasitler (ya da polipeptitler) birbirlerine peptit bağları ile bağlıdır. Protein yapısı dört temel seviyede gösterilmektedir. Bu gösterim aşağıda verilen Şekil 1.2.3'te mevcuttur.



Şekil 1.2.3 – Protein yapıları

Aminoasitlerin peptit bağları ile zincir boyunca oluşturduğu bu dizi, proteinin primer yapısını oluşturmaktadır. Birincil protein yapısı Şekil 1.2.4'te gösterilmiştir.

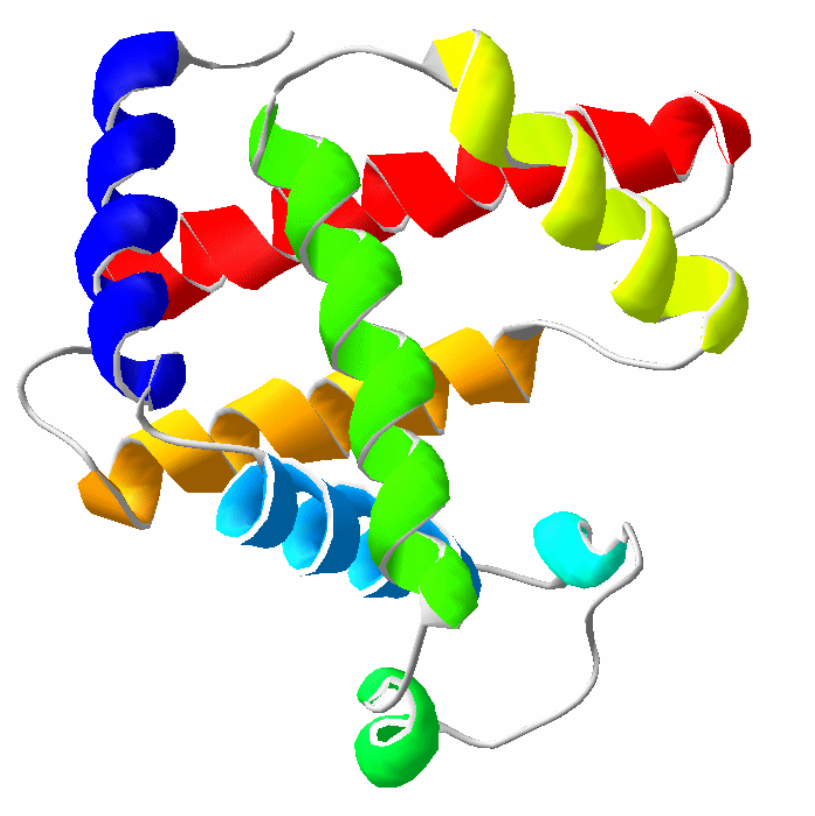


Şekil 1.2.4 – Birincil protein yapısı, aminoasitlerin oluşturduğu zincirler

Proteinler basitçe dizilmiş düz aminoasit zincirleri değildirler. Her protein, kendine has biyolojik özelliklerini belirlemekte çok önemli rol oynayan, çok karmaşık konformasyonlar oluşturmak üzere, kıvrılıp katlanırlar. Bu üç boyutlu yapının çoğu proteindeki peptitler arası zayıf etkileşimler sonucudur.

Bir genelleme yapılacak olursa, proteinler iki geniş sınıf altında toplanabilir. Bu sınıflardan biri fibröz proteinlerdir diğeri ise globüler proteinlerdir. Fibröz proteinler aynı zamanda sarmal yapının en basit haliyle görüldüğü proteinlerdir, ince, uzun ve iplik şeklinde bir yapıya sahiptirler. Globüler proteinler ise sıkı olarak kıvrımlı ve spiral bir yapıdadırlar. Polipeptit zincirlerinin küresel şekilde katlanmış olduğu globüler proteinler, konformasyon bakımından fibröz proteinlerden daha karmaşıklardır.

Bir protein zincirinin kısmı bölgelerindeki, peptit bağlarının sergilediği çift bağ karakteri, o proteinin konformasyonlarını belirler. Bir proteinde, polipeptit zincirinin farklı bölgelerinde birincil aminoasit zinciri tarafından farklı konformasyonlar saptanabilir.



Şekil 1.2.5 – İkincil protein yapısı

California Teknoloji Enstitüsü'nden Robert B. Carey ve Linus Pauling'in 1951'de gösterdiği gibi belirli derecelerdeki kıvrılma, molekül için hidrojen bağlarının alfa (α) sarmal (heliks) denilen yapıyı oluşturmasına ve sabitleştirilmesine izin verir. Bir sarmal düzgün bir silindirin etrafına sarılmış bir kurdele olarak gözükebilir. Bir proteinde sarmalın her tam dönüşü yaklaşık olarak polipeptit zincirinin 3.6 aminoasit ünitesini kapsar. Zincirin bu sarmal şekli bir aminoasidin amino grubu ile polipeptit zincirinde ilerdeki 3. aminoasidin (-ki bu aminoasit sarmalın ekseni yönünde düşünülürse yanındaki aminoasit olmaktadır) oksijeni arasında oluşan hidrojen bağları ile oluşturulur. Peptit zincirinin aminoasitlerinin uzaydaki düzenlenişleri ya da konformasyonları "ikincil yapı" olarak adlandırılır. İkincil

yapının keşfedilen ilk örneği sarmaldır. İkincil protein yapısındaki alfa sarmalları Şekil 1.2.5'te görülmektedir. Polipeptit zincirindeki peptitlerin diğer bir basit düzenlenmesi ikincil yapının ikinci ana tipini oluşturur. Beta (β) yapısı olarak bilinen bu yapı çoğunlukla "pileli tabaka" olarak adlandırılır.

Ortak ikincil yapı elemanları, α-sarmalları, pileli β-tabakaları, kıvrımlar, köprüler, sarımlar ve ilmeklerdir. Bu yapı elemanlarından en açık olanları, α-sarmalları ile β-tabakalarıdır. Kalan yapı elemanlarını tanımlamak pek de kolay değildir, bundan dolayı çoğunlukla "diğerleri" ya da "spiraller" olarak anılırlar.

İkincil yapının üzerine oturtulan üç boyutlu katlanma tipi tersiyer yapı olarak adlandırılır. Pratikte tersiyer yapıyı belirlemek zordur. Myoglobin proteini, küçük düzensiz (sarmal olmayan) kıvrılma bölgeleri ile bağlanan sekiz helis içeren bir polipeptit zincirinden oluşur. Her düzensiz kıvrılma bölgesinde polipeptit zincirinin üç boyutlu kıvrılması değişerek proteinin tipik katlanmasını böylece de tersiyer yapının iyi bir örneğini oluşturur.

Eğer bir globülar protein, kendi başına bağımsız olarak katlanmış iki ya da daha fazla polipeptit zincirin, genellikle zayıf bağlar ile gevşek olarak birbirine tutunması ile oluşmuş ise, bu durumda kuvarterner yapıdan söz edilmektedir. Bir proteinin birincil yapısında (diğer bir deyişle aminoasit dizilimi) yer alan çeşitli aminoasitler onun tersiyer ve kuvarterner yapısını oluşturmasına iştirak etmektedir.

Bir proteinin üst seviye yapıları, o proteinin fonksiyonlarını, faaliyetlerini ve çevre ile olan etkileşimlerini tanımlamada çok büyük öneme sahiptirler. Proteinlerin görevleri aslında yaşamın tam kendisidir. Herhangi bir canlının sahip olduğu proteinlerin çoğu enzimlerdir. Enzimler hücre içerisindeki kimyasal reaksiyonları hızlandıran belirli proteinlerdir. Hücre içerisinde, belli bir reaksiyonun oluşabilmesi için kullanılan küçük moleküler araçlardır. Geçici olarak bileşenlere eklenirler ve bileşenlerin belli açıları yakalayabilmesini sağlarlar, böylece istenilen reaksiyon gerçekleşebilir. Ayrıca bir reaksiyonun gerçekleşebilmesi için gereken enerji ihtiyacını azaltırlar, böylece normalden daha düşük sıcaklıklarda enzimler sayesinde reaksiyonlar gerçekleşebilir. Bunlara ek olarak proteinler hücre içerisinde

daha birçok rolü de üstlenebilmektedir. Proteinler hücre içerisindeki görevlerine göre kategorilere ayrılabilirler, birkaç kategoriye örnek vermek istersek, enzimler, yapısal proteinler, hormonlar, nakil proteinleri, vb.nden bahsedebiliriz. Proteinler, görevlerinden ziyade fonksiyonlarına göre de kategorilere ayrılabilirler, mesela nakil proteinleri kendi içinde oksijen taşıyıcılar ve yağlı asit taşıyıcılar gibi örneklendirilebilirler.

Önceden de bahsettiğimiz yapılar proteininin fonksiyonları belirlemektedir. Ama bundan daha önemli bir soru vardır ki, “proteinin yapısını ne belirler?” şeklinde olacaktır. 1973 yılında Anfinsen tarafından gösterilmiştir ki, proteinlerin birincil yapıları (aminoasit dizilimleri) üçüncü dereceden yapılarında en büyük belirleyici etkendirler. Ancak, primer yapı hakkındaki bilgi yeterli olmamaktadır; esas çözümün saklı olduğu çevre, proteinin üçboyutlu konformasyonunda büyük rol oynamaktadır.

1.3. Veri Kaynakları

Günümüzde birçok biyolojik veriyi saklayan, depolayan ve verimli bir şekilde geri alan veritabanları bulunmaktadır. Bu veritabanlarının bir bölümü çok özel amaçlara yönelik kullanılacak bilgiler depolamak üzere özelleştirilirken, bir bölümü de genel kullanım amaçlıdır ve her türlüyü bilgiyi kullandırmak üzere depolamaktadır. Bu alanda mevcut birçok veritabanına, internet aracılığı ile kendi web siteleri üzerinden erişilmektedir. Veritabanlarının hemen hemen hepsi düzenli olduğundan, veriler direk veritabanından alınıp, yerel iş istasyonlarına indirilebilinmektedir. Biz proteinler ile ilgilendiğimiz için, proteinler ile alakalı önemli veritabanları ekte listelenmiş ve kısaca tanıtılmıştır.

Tez boyunca, deneysel verileri sağlamak veya sonuçları karşılaştırmak için veritabanlarından birçoğuna sıkça erişilmiştir. Uzak homoloji testleri için gereken protein aileleri ve süperaile tanımları SCOP veritabanından indirilmiştir (Murzin et al., 1995). Protein dizileri fasta formatında, PDH ve SWISSPROT veritabanlarından indirilmiştir (Berman et al., 2000; Bairoch ve Apweiler, 1999). Deneysel verilerin

sağlandığı bu veritabanlarının dışında, CATH, NCBI ve Pfam veritabanları ve bu veritabanlarına ait araçlar da, deneylerde kullanılan verilerin doğruluğunu ve güvenilirliğini kontrol etmek için sıkça kullanılmıştır.

2 – ÜST SINIFLANDIRMA YAKLAŞIMI

2. Üst Sınıflandırma Yaklaşımı

Bilgi geri getirim sistemleri, doğrudan ya da dolaylı olarak, erişim yöntemlerini temel almaktadır. Basit modellerin örnekleri, olasılıklı veya Bayes sınıflandırma (Robertson ve Sparck Jones, 1976; Van Rijsbergen, 1979) ve vektör uzayı modelini içermektedir. Daha birçok model ileri sürülmüş ve günümüzde de kullanılmaktadır (Van Rijsbergen, 1986; Deerwester et al., 1990; Fuhr, 1992; Turtle ve Croft, 1992).

2.1 Sorguların Birleştirilmesi

Gösterimlerin birleştirilmesi ile ilgili deneysel çalışmalar göstermiştir ki, genelde, birden fazla gösterim, geri getirme etkisini artırmaktadır. Bunun yanı sıra bir kanıt kaynağı diğer bir kanıt kaynağından zayıf ise (ilgiyi ölçme yetisi düşük), bu durum kanıt toplama işlemi içinde yansıtılmalıdır, aksi durumda etki ortadan kalkmaktadır. Bu gözlemler basit olasılıklı yapı içerisinde tutarlıdır.

Ne var ki, ilgiyi ölçmek, doküman gösterimlerinden daha da fazlasını kapsamaktadır. Araştırmacıların bilgi ihtiyaçlarının gösterimleri olan sorgular, P(R|D,Q) hesaplama işleminin büyük bir kısmını oluşturmaktadır. Sorgunun, gerçek bilgi hakkında sahip olduğu her küçük ek kanıt, geri getirmede fiilen büyük farklılıklar yaratabilir. Bu çok öncelerden beri kabul edilmiştir ve ilgi geri besleme (Salton ve McGill, 1983), sorgu genişleme (Xu ve Croft, 1996; Mitra et al, 1998) gibi tekniklerin de temelinde yatmaktadır. Aynı zamanda geri besleme ve sorgu genişleme, bilgi ihtiyacının gösterimi için alternatif gösterimler yaratan teknikler olarak da görülebilmektedir. Kavramlar dizini gibi geleneksel sorgu ileri sürme araçları da aynı şekilde görülmektedir. Öncelerde, kavram diziniyle beraber yapılan geri getirme ve terim kümeleme kullanılarak yapılan otomatik sorgu genişleme deneylerinden (Spark Jones, 1971) bile, kavramlar dizininin sınıfları veya terim

kümeleri, kelime-tabanlı sorguları ile birleştirilmiş alternatif gösterimleri olarak ele alınmaktaydı.

Alternatif doküman gösterimlerinin bazılarının kullanılabilmesi için dahi, en azından sorgunun aynı gösterimler kullanılarak kısmen de olsa açıklanması beklenmektedir. Salton et al. 1983'de ilk sorguya alıntıları eklemek için ilgi geri besleme tekniğini kullanmışlardır, sonuç olarak da sıralamayı iyileştirebilmek için doküman gösterimlerinde alıntıları kullanabilmişlerdir. Crounch et al. 1990'da sorguya kontrol edilmiş kelimeleri eklemek için geri besleme tekniğini kullanmışlardır. Xu ve Croft 1996'da ağırlıklı ortalama kullanılarak ilk kelime tabanlı gösterimlerle birleştirilmiş sorgunun cümle tabanlı gösterimini ortaya koyabilmek için otomatik sorgu genişleme tekniğini kullanmışlardır. Callen et al. 1995a, alternatif sorgu gösterimlerinin otomatik yapımı için bir dizi strateji tanımlamışlardır.

Alternatif sorguların olduğu fikri sadece temelde yatanın, araştırmacı ile ilişkilendirilmiş bilgi ihtiyacı olduğu, sanıldığında anlam kazanmaktadır. Verilen bir sorgu, bu bilgi ihtiyacının gürültülü ve tamamlanmamış bir gösterimi olarak kabul edilmektedir. Birden çok sorgu yaparak, ilgi hakkında daha çok kanıt parçacığı yakalanmaktadır. Bilgi ihtiyacı hakkındaki en iyi bilgi kaynağı elbette ki araştırmacının kendisi olmaktadır. Birden fazla çalışma, tek bir araştırmacının birden fazla sorgusunu yakalamak veya birden fazla araştırmacının aynı bilgi ihtiyacını tanımlamasının etkilerine bakmış veya etkilerini gözlemlemiştir. McGill et al. 1979'da sıralama algoritmalarını etkileyen etmenler üzerinde çalışmalar yürütmüşlerdir. Bu çalışmalar ışığında, aynı bilgi ihtiyacı tanımlandığında, farklı aramalarda, aracılar (belirli bir arama sistemini kullanmakta uzman olan insanlar) geri getirilen dokümanlar arasında sürpriz bir şekilde azda olsa üst üste çakışmaların olduğunu fark etmişlerdir. Saracevic ve Kantor, 1988'de farklı aracıların, bilgi ihtiyacının aynı tanımı üzerinden Boolean arama yapmaları sonucu geri getirilen kümelerde az da olsa üst üste çakışmalar olduğunu bulmuşturlar. Buna ek olarak, ilişkili olarak değerlendirilen dokümanın şansının geri getirilen kümede olma, bulunma sayısıyla orantılı olduğu gözlemlenmiştir.

Yapılan çalışmaları temel alarak, Turtle ve Croft 1991'de bilgi ihtiyacındaki çoklu gösterimler fikrini apaçık şekilde ortaya koyan bir geri getirim modeli önermişlerdir. Kelime tabanlı ve Boolean sorgularını birleştiren deney sonuçlarını geri getirim etkililiğini iyileştirmek için bildirmişlerdir. Rajashekar ve Croft, 1995'te bu çalışmaları, kelime tabanlı sorgularla, farklı tiplerde elle dizinleme üzerine kurulu başka iki sorguyla birleştirip, genişletmişlerdir. Sorgu gösterim ikilililerini birleştirmek, tutarlı performans ilerlemeleri kaydetmiştir. Bütün üç gösterimin ağırlıklı birleşimleri de en iyi geri getirim etkililiğini sağlamıştır.

Belkin et al. 1993'te, aynı olasılıklı yapı içinde, sorgu birleşimlerinin etkileri hakkında çok daha sistematik bir çalışma sürdürmüştür. Sorgu birleşimi ile geri getirim etkililiğinin yeteri kadar çok iyileştirilebileceğini doğrulamışlardır ancak sorguların birleşimindeki etkilik, birleştirilen sorguların tek başına etkililiğine bağımlıdır. Diğer bir deyişle, ilgi hakkında fazla kanıta sahip olmayan sorguların birleşimdeki ağırlığı , birleşim performansını iyileştirmek için düşük olmaktadır. Çünkü kötü sorgu gösterimleri, daha iyi gösterimlerle birleştirildiğinde, birleşim sorgunun etkililiğini aşağı düşürmektedir. Sonralarda, daha kapsamlı bir çalışmada, Belkin et al. 1995'te, esasen aynı sonuçları elde etmiştir ve sorgu birleşimiyle, farklı sistemlerin çıktılarını birleştirme stratejilerini (diğer adıyla veri füzyonu) karşılaştırmışlardır.

2.2 Sınıflandırma Algoritmalarının Birleştirilmesi

Sınıflandırma algoritmaları olasılıklı geri getirim veya vektör uzayı yaklaşımı gibi aynı genel yapı içerisinde ya da farklı yapılarla gerçekleştirilebilinir. Sınıflandırma algoritmaları aynı, üst üste binen veya tamamen ayrı veritabanları üzerinde çalışabilirler. Bu bölüm aynı yapı içerisinde gerçekleştirilen ve aynı veri üzerinde çalışan sınıflandırma algoritmalarının çıktılarının birleşimleri üzerinde odaklanmaktadır.

Croft ve Harper 1979'da ortalama performansları çok benzer olmasına karşın, küme-tabanlı sınıflandırma algoritmalarının, kelime-tabanlı sınıflandırma algoritmalarına kıyasla, konu ile ilgili daha fazla farklı doküman geri getirdiğini belirtmişlerdir. Kelime-tabanlı aramanın başarısız olduğu durumlarda alternatif strateji olarak küme-tabanlı aramayı kullanmayı önermişlerdir. Özellikle TREC değerlendirmelerinde, diğer sınıflandırma algoritmaların performansları için benzer gözlemler yapılmıştır (Harman, 1995). Alternatif sınıflandırma algoritmalarının sağlanması yaklaşımı, bazı deneysel geri getirim sistemlerinin tasarımlarına dâhil edilmiştir, özellikle I^3R (Croft ve Thompson, 1987) ve CODER (Fox ve France, 1987). Uyarlanabilir bir ağ kullanarak verilen bir sorgu için en iyi algoritmayı seçmek (Croft ve Thompson, 1984), ve mantıklı bir sonuç çıkarsama ağı kullanarak çeşitli sınıflandırma algoritmalarının verdiği sonuçları birleştirmek için denemeler yapılmıştır (Croft et al., 1989). Turtle ve Croft, 1991'de, olasılıklı yapı içinde en yakın komşu küme arama, kelime-tabanlı aramayla nasıl birleştirilip, tanımlandığını göstermişlerdir.

Sınıflandırma algoritmalarının çıktılarını birleştirilmesi, sınıflandırıcının çıktılarını birleştirilmesi olarak modellenir (Tumer ve Ghosh, 1999). Bir sınıflandırma algoritması, her sorgu için, sınıfların anlamlı ve anlamsız olarak ilişkilendirildiği bir sınıflandırıcı tanımlamaktadır (Van Rijsbergen, 1979). Bu sınıflandırıcılar, anlam geri beslemesini kullanarak eğitilirler ancak konu ile ilgili sınıf için kullanılabilir tek bilgi tipik olarak sorgudan gelmektedir. Gerçekte, bilgi ihtiyacının birçok gösterimin birleştirilmesi yaklaşımı, tam olarak çeşitli sınıflandırıcılar kurmak (her sorgu gösterimi için bir tane) ve sınıflandırıcıların çıktılarını birleştirmek olarak görülmektedir. Bu bakış açısından görülmektedir ki, yapılan deneyler, Rahashekar ve Croft, 1995 ve Beklin et al., 1995, bilgi geri getirimi için birçok sınıflandırıcının birleştirilmesinin işe yarar olduğunu onaylamaktadır.

Genel olarak sinir ağları araştırması ve makine öğrenme alanlarında sınıflandırıcıların birleştirilmesi yaygın bir şekilde çalışılmaktadır. Tumer ve Ghosh, 1999'da, bu alanın iyi bir taramasını vermiştirler. Sınırlı eğitim verisi, büyük ve gürültülü desen uzayı, ağırlık değişimleri, başlangıç durumları ve sınıflandırıcının

içyapısı, sınıflandırıcının farklı çıktılar vermesine neden olabileceği gösterilmiştir. Bu tam olarak araştırmacılarının gözlemlediği sonuçtur (Lee, 1995).

Tumer ve Ghosh, her ne kadar, en uygunu öğrenen ağırlıklı ortalama gibi daha karmaşık birleştirme stratejileri değerlendirilirken, basit bir şekilde sınıflandırıcıların çıktılarının ortalamasını almanın, en yaygın ortak birleştirme stratejisi olduğunu gözlemlemişlerdir. Ortalama alma stratejisini analiz etmişler ve tarafsız, bağımsız sınıflandırıcılar için Bayes hatası üzerine eklenen hatanın, N sınıflandırıcı için N faktör oranında azaldığını göstermişlerdir. Bir sınıflandırıcının çıktısını, her sınıf için olasılıklı dağılımların birleşimi ve gürültü dağılımı (hata) şekilde bir girdi için modellemişlerdir. Hatayı azaltmak, gürültünün değişimini azaltmaya karşılık gelmektedir. Sınıflandırıcının hataları ilgili veriyi getirmemeye veya ilgili olmayan veriyi getirmeye karşılık geldiğinden dolayı, bu hatayı azaltmak geri getirme faydasını iyileştirecektir (Van Rijsbergen, 1979). Tumer ve Ghosh aynı zamanda, basit birleştirme stratejilerinin, tüm sınıflandırıcıların aynı işi yaptığı durumlarda (bilgi geri getirimde geçerli olan durum) en uygun ve karşılaştırılabilir bir başarıya sahip olduğuna değinmişlerdir. Basit birleştirme stratejileri, birleştirilen sınıflandırıcılardan sadece biri bile çok düşük veya dengesiz bir performansa sahip olduğunda, başarısız olmaktadır.

Toplama, ortalama alma ve ağırlıklı ortalama alma gibi basit birleştirme stratejilerinin bilgi geri getirme için yeterli olduğu konusunda bir hayli kanıt mevcuttur. Ağırlıklı ortalama alma, sınıflandırıcılardan birinin çok zayıf bir değere sahip olduğu gibi, bazı durumlarda gereklidir. Bartell et al., 1994'te bilgi geri getirme için sınıflandırıcıların ağırlıklı, doğrusal birleşimlerini öğrenen bir yaklaşımı tanımlamıştır. Fox ve Shaw, 1994'te vektör uzayı modeli ile farklı geri getirim algoritmalarını kullanarak, birleştirme stratejilerinin bir değerlendirmesini ileri sürmüşlerdir. En iyi birleştirme stratejisinin geri getirme algoritmalarının çıktılarının toplamından ibaret olduğunu bulmuşlardır, bu durum final mertebede terimlerin ortalamasını almaya denk olmaktadır. Hull et al., 1996'da, doküman filtreleme problemi için (bu problem oldukça fazla eğitim verisine sahiptir) basit ve karmaşık sınıflandırma birleşimlerini karşılaştırmışlardır, en iyi ilerlemenin basit ortalama alma stratejisi ile edildiğini bulmuşlardır.

Bilgi geri getirim sınıflandırıcıları, belirgin olarak aynı dokümanı ve sorgu gösterimlerini kullanmadıkları sürece çoğunlukla bağımsız değildirler. Geri getirme faydasının iyileştirilmesi bazında en iyi sonuçların bir çoğu, çok farklı gösterimler üzerinde sınıflandırıcıların birleştirilmesiyle meydana gelmektedir. Birbirine çok benzer sınıflandırıcıların birleştirilmesi genellikle performans üzerinden önemli bir iyileştirmeye sebep olmamaktadır. Lee, 1995'te, vektör uzayı modeli içinde farklı ağırlık düzenlemelerini temel alan geri getirim çıktılarının birleştirilmesi üzerine çok geniş kapsamlı bir çalışma yürütmüştür. Bu çıktıları, normalleştirilmiş puanların ortalamasını alarak, birleştirmiştir. Yaptığı deneysel çalışmalar şunu göstermiştir ki, benzer ağırlık düzenlemelerini temel alan sınıflandırıcıları birleştirmek performansı çok az bir oranda artırmaktadır. Sınıfları, tf.idf ağırlıklarının kosinüs normalleştirmesi temel alınarak birleştirmenin diğer normalleştirme düzenlemelerine göre daha etkili olduğunu açık bir şekilde ortaya koymaktadır (ortalama %15 kesin iyileştirme sağlamaktadır). Hull et al., 1996'da birleşimlerin ortaya koyduğu performans kazançları, kullanılan sınıflandırıcı bağıntıları ile sınırlı olduğunu bulmuştur. Bu bağıntı birincil olarak aynı eğitim verisinin kullanımından oluşmaktadır ve aynı doküman gösterimlerini kullanan sınıflandırıcılar içinde en güçlü düzeyde bulunmaktadır.

Tumer ve Ghosh, sınıflandırıcı birleştirme tartışmasında aynı yapı içerisinde aynı kararları verebilmek için sınıflandırıcıların karşılaştırılabilir çıktıya sahip oldukları varsayılmaktadır. Olasılıklı sistemler için, bunun anlamı hepsinin P(R|D,Q) ölçümleri yapılmaya çalışılmaktadır, bu ölçümler basit stratejiler kullanılarak birleştirilebilirler. Önceki olasılık bilgisine sahip olunmadığından ve eğitim verisinin yeterli olmayışından dolayı, bu hesaplamalar kesinlik sağlayamamaktadır ve sınıflandırıcıların çıktıları da tam uyumlu olmamaktadır. Küme-tabanlı bir geri getirme algoritması ile yazı tabanlı bir geri getirme algoritmasını birleştirmek zor olabilir çünkü sınıflandırma için bu algoritmalar tarafından üretilen sayıların, ayırıcı olma olasılığı sayıları ile olan ilişkisi düşük seviyelerde olabilmektedir. Yeterli eğitim verisi ile bu sayılar arasındaki bağlantı ve olasılıklar öğrenilebilir ancak böyle bir veri mevcut değildir. Sınıflandırıcı çıktılarının uyuşmazlığı Lee'nin 1995'te ki makalesinde tartışıldığı gibi vektör uzayı modelinde de ortaya çıkmaktadır. Makale

de, koşturulan farklı erişimlerden elde edilen puanlar uyumluluğu artırabilmek amacıyla her koşu için o koşunun maksimum puanlarıyla normalleştirilmektedir. Tamamen farklı arama sistemlerinin birleştirilmesi gibi bir yaklaşımda bu problem kendini daha fazla göstermektedir.

2.3 Arama Sistemlerinin Birleştirilmesi

Farklı arama sistemlerinin çıktılarını birleştirme fikri DARPA TIPSTER projesi sırasında ortaya çıkmıştır (Harman, 1992) ve TREC değerlendirmeleri ile ilişkilendirilmiştir (Harman 1995). Bu değerlendirmeler, aynı büyük veritabanları üzerinde aynı sorgular üzerinde çalışan birçok arama sistemini kapsamaktadır. Bu aramaların sonuçları araştırmalar için kullanılabilir olarak sunulmuş ve birleştirme stratejileri üzerinde çalışılmıştır. Beklin et al., 1995'te, olasılıklı ve vektör uzayı sistemlerinden aldıkları arama sonuçlarını birleştirmiş ve performans iyileştirmelerini göstermişlerdir. Lee, 1997'de bir TREC değerlendirmesi için seçilmiş altı geri getirim sisteminin sonuçlarını belirlemiştir. Fox ve Shaw, 1994'te ve Lee 1995'te kullanılan birleştirme stratejilerini gözden geçirmiştir. Farklı geri getirim sistemlerinden alınan sonuçlar, aşağıdaki formülde verilen minimum ve maksimum puanlar kullanılarak normalleştirilmiştir.

$$normal\_puanı = \frac{normal\_olmayan\_puan - \min\_puan}{\max\_puan - \min\_puan} \quad (2.3.1)$$

Lee'nin sonuçları en etkili birleştirmelerin (tek bir arama için yaklaşık olarak %30 iyileştirme kaydedilmiştir) ilgili dokümanların benzer kümelerini ancak ilgisiz dokümanların farklı kümelerini geri getiren sistemler arasında olduğunu göstermiştir. Bu durum Lee'nin 1995'te yaptığı gözlem ile alakalıdır. Yani geri getirilen kümelerin düşük oranda üst üste bindiği geri getirim algoritmaları, tüm toplamda benzer performans vermektedir, birleştirmede de en iyi sonucu üretmektedir. Vogt ve Cotrell, 1998'de daha iyi bir birleştirme performansı tahmin eden faktörlerle ilgili bir çalışmada, başka bir TREC değerlendirmesinde mevcut tüm sistemlerin (tümü 61 adet) ikili birleştirmesine bakmışlardır. Bu çalışma

sonucunda da Lee'nin gözlemini, en iyi birleştirmelerin ilgili dokümanın benzer kümelerini, ancak ilgisiz dokümanın farklı kümelerini geri getiren sistemler arasında yapıldığını, onaylamışlardır.

Bu sonuçlar ilintisiz (uncorrelated) sınıflandırıcılar bakımından basit şekilde açıklanabilir. Her sistemin her sorguda geri getirdiği en üstteki 1000 doküman bu çalışmalarda karşılaştırılan doküman kümeleridir. Verilen bir sorgu için ilgili dokümanların sayısı tipik olarak pek de fazla olmadığı bilinmektedir (100-200). Böylece, ilgili dokümanların büyük bir bölümünün, çoğu sistem tarafından geri getirilmesi beklenmektedir. Bu durum gerçekte Lee tarafından geri getirilen ilgili doküman kümeleri arasında bağıntıların analizi ile gösterilmiştir. Çok fazla sayıda ilgisiz doküman olduğundan, ilintisiz sınıflandırıcıların (arama sistemlerinin) ilgisiz dokümanların farklı kümelerini geri getirmesi beklenmektedir, gözlemlenen bu durumdur.

Geri getirilen ilgili doküman kümeler içinde üst üste binme oranının yüksek oluşuna rağmen ilintisiz sistemlerde ilgili dokümanların farklı sınıflandırılması beklenmektedir. Vogt ve Cottrell iyi birleşimler için sınıflandırmadaki farklılığı gözlemlemişler. En iyi birleştirme, birleştirilen her iki sistemde iyi performansa sahipse gerçekleşmektedir ancak sistemlerden sadece biri iyi performansa sahipse yine iyileştirme sağlama ihtimali mevcuttur denilmektedir. Tüm bu gözlemler ancak "sınıflandırıcılar bağımsız ve hatasız ise birleşme en düşük hata payı ile gerçekleşir" ifadesi sağlandığında tutarlı olmaktadır.

Lee, 1997'de farklı arama sistemleri için birleştirme stratejileri ile ilgili başka iki sonuç daha ortaya koymuştur. Bunlardan ilki, arama sistemlerinin sonuçlarını dokümanların puanlarından ziyade sınıflarını kullanarak birleştirmek genelde çok etkili olmamaktadır sonucudur. Puan-sınıf eğrisinin şekli bakımından arama sistemlerinin çok farklı bir karakteristiğe sahip olduğu durumlar hariç tutulmaktadır. İlgili olma olasılığı için bu normalleştirilmiş puanın çoğunlukla sıralamaya (rank) göre daha iyi bir tahminci olduğunun kanıtı olarak yorumlanabilir. Sıralamayı kullanmak, düzgünleştirmenin en şiddetli formlarındandır, sadece birleştirilecek

olan sistem çok farklı bir puanlama karakteristiğine sahip olduğunda hata payını yükseltmektedir.

En iyi birleştirme stratejisi normalleştirilmiş puanların toplamı ve sonra birleştirme içerisinde sıfır olmayan sonuçların toplam sayısı ile çarpılmasıyla ortaya çıkmaktadır, bu da Lee'nin ikindi kuralıdır. Bu, puanları basit bir şekilde toplamaktan biraz daha iyidir ancak daha tutarlı bir paya sahiptir. Bu birleşim formu birden fazla sistemin geri getirdiği dokümanlara aşırı derecede gerek duymaktadır. Bir doküman için sıfır puanın anlamı, ilgili olma olasılığı için doğru bir tahmin olduğundan ziyade o sistemdeki üstteki 1000 doküman içinden geri getirilmediğin anlamındadır.

Yukarıda değinilen denemeler birden fazla arama sisteminin çıktılarını aynı veritabanını kullanarak birleştirmiştir. Üst üste binen veya tamamıyla ayrı veritabanlarını kullanan arama sistemlerinin çıktılarını birleştirmekte mümkün olmaktadır. Bu tip birleşimlere toplama füzyonu (collection fusion), dağıtılmış bilgi geri getirim sistemleri (distributed IR) veya meta-arama (meta-search) denilmektedir. Vorhees et. al., 1995'te birleşim içindeki her bir sistemle, aralarında ilişki kurmaya yönelik ağırlıkları öğrenmek için teknikler üzerinde deneyler rapor etmişlerdir. Bu ağırlıklar, dokümanları son kez sınıflandırmak nasıl karıştırılacağını belirlemeye yönelik, sınıflandırılan doküman kümeleri ile beraber kullanılmaktadır. Bu yaklaşım, Lee'nin 1997'de kullandığı rank birleştirme stratejileri ile bazı benzerliklere sahiptir.

Callan et. al., 1995b göstermiştir ki bir sorgu için veritabanına ait bir değer ölçümü ile puanlara ağırlıklar eklemek, seviyeleri sırayla birleştirmeye göre daha yeterlidir. Hem Voorhees hem Callan, deneyleri için ayrılmış veritabanları kullanmışlardır. Pratikte kullanılan birçok ortamda, örneğin internet üzerinden meta-aramalarda, arama sistemleri tarafından kullanılan veritabanları üst üste binmektedirler. Bu sonuç Lee'nin 1997'de anlattığı, dokümanların kendileriyle ilişkilendirilmiş birden fazla değişiklik gösteren puanlara sahip olması durumuna çok benzemektedir. Web arama sonuçlarının birleştirilmesinde tam ve kesin bir değerlendirme yoktur ancak bu durumda Lee'nin sonuçları uygulanabilir olarak görünmektedir. Bunun anlamı,

en iyi birleştirme stratejisi, her arama motorunun puanını normalleştirmek, normalleştirilen bu puanları her doküman için toplamak ve bulunan bu toplamı dokümanı döndüren arama motorlarının sayısıyla çarpmak olacak, şekilde verilebilecektir. Eğer arama motorlarının kullandığı veritabanlarında üst üste binme oranı düşük ise (indekslenen web sayfalarının miktarı içinde yeterli farklılık mevcut ise), yukarıda verilen adımlardan sonuncusu çok da etkili olmamaktadır.

Birden fazla arama sisteminin çıktılarının birleştirilmesi durumu aynı zamanda mültimedya geri getirmede de uygulanmaktadır (Croft et. al., 1990; Fagin, 1996; Fagin 1998). Bu durumda, tipik olarak yapılacak işlemler, metin araması sonuçlarını ve renk dağılımları, desen vb. gibi görüntü özelliklerini karşılaştıran bir ya da birden fazla sınıflandırma algoritmasının sonuçlarını birleştirmek olacaktır. Tartışılan deneysel sonuçlar, görüntü ve metin geri getirme algoritmalarından gelen puanlar, potansiyel olarak sıfır olmayan puanların sayısını da hesaba katarak, önce normalleştirmeyi sonra da toplanarak birleştirmeyi önermektedir. Maalesef ki, bunun iyi bir seçim olduğuna dair, sınıflandırıcı birleştirmesi hakkındaki teorik sav dışında hiçbir kesin kanıt bulunmamaktadır.

Fagin, 1996 bulanık mantığın standart operatörleri yani *min* ve *max* ile mültimedya veritabanları içinde puanları birleştiren bir algoritma geliştirmiştir. Lee yaptığı deneyler ile bu birleştirme operatörlerinin performansının, toplamaya kıyasla daha kötü performans sergilediğine kanıt sağlamıştır. Ciaccia et. al., 1998'te mültimedya veritabanı ortamında sınıflandırma üzerine çalışmıştır. Birincil olarak birleştirmenin verimliliği ile ilgilenilmiştir, Fagin'in de yaptığı gibi birleştirme operatörlerinin performans sonuçlarını sunmuşlardır.

2.4 Gösterimlerin Birleştirilmesi için Yapılar

Vektör uzayı modeli çok sayıda birleşim problemi için temel olarak alınmıştır. Bu modelde, dokümanlar ve sorgular ağırlıklı terimlerden oluşan vektörler ile tanımlanmıştır. Fox et al. 1988'de farklı "kavram tiplerini" veya dokümanlardan türetilen gösterimleri tanımlamak için yardımcı vektörler kullanmayı önermiştir.

Dokümanları sıralamak için kullanılan, bir doküman ile bir sorgu arasındaki tüm benzerlik, her yardımcı vektör için benzerliklerin doğrusal birleşimleri olarak hesaplanması olarak verilmektedir. Örneğin, eğer dokümanlar, kelimeler, yazarlar, alıntılar kullanılarak gösterilirse, benzerlik fonksiyonu:

$$\begin{aligned} benzerlik(Q,D) = {} & c_{kelime} \,.\, benzerlik\,(Q_{kelime}, D_{kelime}) \\ & + c_{yazar} \,.\, benzerlik\,(Q_{yazar}, D_{yazar}) \\ & + c_{alıntı} \,.\, benzerlik\,(Q_{alıntı}, D_{alıntı}) \end{aligned} \qquad (2.4.1)$$

olmaktadır. Burada c_i değerleri katsayılardır. Bu göstermektedir ki, gösterimleri birleştirilmesi ve arama çıktısının birleştirilmesi yapıları arasındaki farklılık çok azdır. Fox et al. 1988'de benzerlik fonksiyonunu ilgiyi tahmin etmek için kullanmıştır, daha sonra katsayı değerlerini belirleyebilmek için toplama test verisiyle regresyon analizi yapılmaktadır.

Fuhr ve Buckley 1991 yılında belge gösterimlerinin etkili birleştirilmesini öğrenmek için regresyon uygulamıştır. Bu çalışma temelini olasılıklı modelden almaktadır. Direk olarak kelimeleri kullanmak yerine bu nitelikleri temel alan gösterimleri model içindeki olasılıkları hesaplayabilmek için daha fazla eğitim verisi sağlamaktadır. Terim ve doküman niteliklerinin polinom birleşim katsayılarını hesaplamak için en küçük karesel hata ölçütü kullanılmıştır.

Gey, 1994'te, genelde olasılık değerlerini hesaplamada daha uygun olarak lojistik regresyon kullanan lojistik çıkarsama modelini geliştirmiştir. Formül, dokümanlardaki ve sorgulardaki terim karakteristiklerinin doğrusal birleşimi olarak verilmektedir.

Greiff, 1998'de terimler, dokümanlar ve ilişkileri keşfedecek ayırıcı olma özelliklerini kapsayan büyük miktarlardaki verilere bakarak, araştırma ile ilgili veri analizlerini de kullanarak olasılıklı bir model tanımlamıştır. Bu yaklaşım biraz da olsa önceden tanımlanan regresyon modellerine benzerlikler göstermektedir, ancak temelde yatan dağılımlar hakkında sanılar ortaya koymamaktadır. Greiff, 1999'da da yaklaşımını, bilinen kısıtlara göre uygun olasılıklı modeli belirlemek için kullanılan

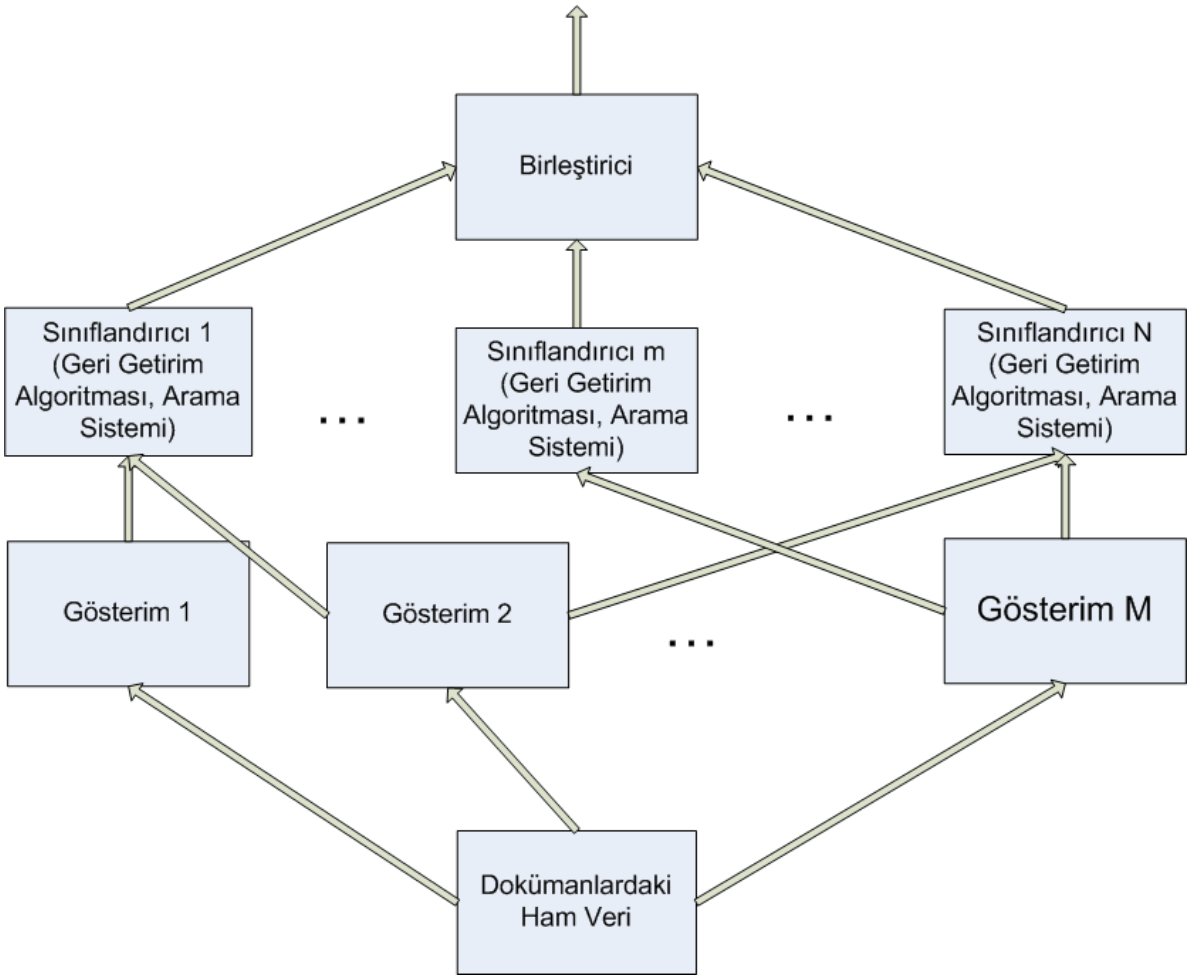
bir yol olan maksimum entropi prensibi temeline dayanarak, birden fazla kanıt (gösterimler) kaynağı ile birleştirmiş ve genişletmiştir. Birleştirilen modeller bazında oldukça basit bir model olmasına rağmen, geliştirilen yapı önceden bahsedilen, tanıtılan ve önerilen gösterimlerle birleştirilmek için oldukça yeterlidir. Yeni bir gösterimi birleştirmek, geri getirilen ve o gösterimi kapsayan veri üzerinde çalışmayı, regresyon kullanarak formülün, yeni gösterim tarafından sağlanan kanıtlarına göre ilgiyi hesaplayan katsayıları çıkarmaktadır. Bu formül basit bir biçimde diğer gösterilen formüllerinin doğrusal birleşimlerine de eklenebilmektedir.

En geneli Greiff'in çalışmaları gibi görünen, eğitim verisi ve regresyon kullanan tüm yapılar için ortak karakteristik, normalleştirilmiş benzerlik değerleri yerine ilginin yaklaşık hesaplamalarını kullanan geri getirme algoritmaları veya arama sistemleridir. INQUERY (Callan et al. 1995a) gibi diğer olasılıklı sistemler için sistemin asıl amacı dokümanları sıralamak olduğundan, verilen bir sorgu (öncelikli olasılıklar gibi) için olasılık formülünün parçaları sabit kabul edilip, yok sayılmaktadır. Buna ek olarak, ad-hoc formülleri doküman puanlarının parçalarını hesaplamak için kullanılmaktadır. Bunun anlamı, bu sistemler tarafından üretilen sayılar olasılık değerleri olmamaktadırlar. Bu durumda, sistemin sonucu ile diğer sistemlerin sonuçları birleştirilmek istendiğinde gözle görülür bir problem ortaya çıkmaktadır. Birbiriyle uyumlu çıktılara sahip sistemler ve yanlışsız olasılık hesaplamaları, bağımsız ve en iyi birleşimleri oluşturmaktadır.

2.5 Geri Getirim Algoritmalarının Birleştirilmesi için Yapılar

Sonuç çıkarım ağ yapısı, Turtle ve Croft (Turtle ve Croft, 1991; Turtle ve Croft, 1992) tarafından geliştirilen ve INQUERY sistemi olarak implemente edilen (Callan et al. 1995a) sonuç çıkarım ağı yapısı, kesin olarak, çoklu gösterimleri ve geri getirim algoritmalarını ilgi olasılığını ayrıntılı hesaplaması için birleştirmeye yönelik tasarlanmıştır. Bu yapı, olasılıklı modelde önermeleri ve bağlılıkları göstermek için bir Bayes ağı (Pears, 1998) kullanılmaktadır (Şekil 2.5.2). Bu ağ iki parçaya bölünmüştür: birinci parça doküman ağı, ikinci parça ise sorgu ağıdır. Doküman ağındaki düğümler, dokümanların incelenmesi (D düğümleri), dokümanların içeriği

(T düğümleri), içeriğin gösterimi (K düğümleri) hakkında önermeler ortaya koymaktadır. Sorgu ağındaki düğümler, sorguların gösterimi (K ve Q düğümleri), bilgi ihtiyacının tazmini (I düğümü) hakkında önermeler ortaya koymaktadır. Bu ağ modeli Şekil 2.5.1'de verilen sınıflandırıcıların birleştirilmesine olan genel bakışa yakından bezemektedir. Dokümanlarda ham veriyi modelleyen ağın bölümleri, bu veriden çıkarılan nitelikler, ilgiyi hesaplamak için bu nitelikleri kullanan sınıflandırıcılar ve sınıflandırıcı çıktıları için geniş kapsamlı birleştirici, Şekil 2.5.2'de etiketlenmiştir.



Şekil 2.5.1 – Geri Getirim için Stratejilerin Birleştirilmesi

Bu modelde true ve false değerli ikili değişkenlerden oluşan önermeleri gösteren tüm düğümler ve bu durumların bir düğüm için olasılığı, ebeveyn düğümlerin durumları ile belirlenmektedir. A düğümü için, A düğümünün true olma olasılığı,

$$p(A) = \sum_{S \subseteq \{1,\ldots,n\}} \alpha_S \prod_{i \in S} p_i \prod_{i \notin S} (1 - p_i) \qquad (2.5.1)$$

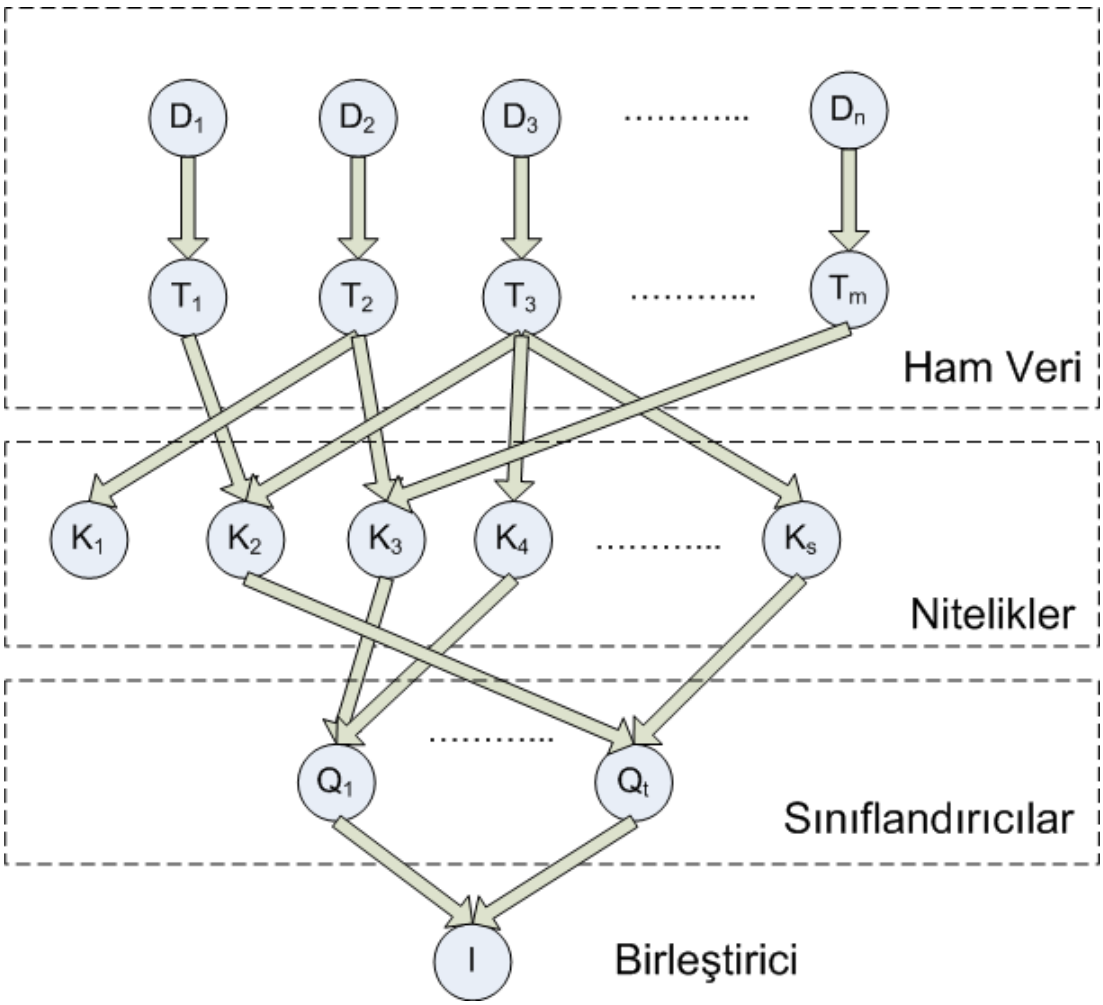
olarak verilmektedir. α_s, durumu true olan n ebeveynlerin özel alt kümesi S ile ilişkilendirilmiş, katsayısı ve p_i, true durumuna sahip ebeveynin olasılığıdır. Bazı katsayı ayarlamaları basit ancak ebeveyn birleştirmeleriyle genel kanıtların etkili birleştirilmeleriyle sonuçlanır. Örneğin, α_s = 0 olsun, tüm ebeveynler true durumda değil ise bu bir Booleanana uygun gelmektedir. Bu durumda,

$$p(A) = \prod_{i=1}^{n} p_i \qquad (2.5.2)$$

olmaktadır.

Bu yapıda en yaygın olarak kullanılan birleştirme formülleri, ebeveyn olasılıklarının ortalamalarını ve ağırlıklı ortalamalarını almaktadır. Bu formüller aynı zamanda sınıflandırıcılar için kullanılan en iyi birleştirme stratejileridir. Ebeveyn olasılıklarının ortalamasını temel alan birleştirme formülü A'nın true olma olasılığının, sadece durumu true olan ebeveyn düğümlerinin sayısına bağlı olduğundaki katsayı düzeninden gelmektedir. Ağırlıklı ortalama, A'nın olasılığının belirli ebeveynlerin durumunun true olduğu düzenden gelmektedir. Büyük ağırlıklara sahip ebeveynlerin A'nın durumu üzerindeki etkileri de daha büyük olmaktadır.

Sonuç çıkarım ağının avantajı farklı gösterimleri, geri getirim algoritmalarını ve bunların çıktılarını birleştirmeyi temel alan yeni sınıflandırıcıların hızla kurulması için olasılıklı bir yapı sağlamaktadır. Bu yapı içerisinde her geri getirim algoritması tanımlanamaz ancak Greiff et al. 1997'de kolayca hesaplanabilen birleşim operatörlerinden oluşan bir sınıfı ve bunların iyi bilinen bir vektör uzayı sıralama algoritmasını modellemek için nasıl kullanıldığını anlatmaktadır.



Şekil 2.5.2 – Bilgi Geri Getirimi için Bayes Net Modeli

Her ne kadar bu çaba kıyaslanabilir ya da daha iyi etki yaratmak için (kıyaslandığında daha iyi bir etki yarattığı için) başarıya ulaşmış olsa da, olasılıklı yaklaşım tabanlı olmayan böylesine basit geri getirim algoritmalarını modellemede ki zorluğu göstermektedir. Örneğin, sinir ağları mimarisi tabanlı daha karmaşık bir geri getirim algoritmasını, sonuç çıkarım ağıyla modellemek mümkün olmamaktadır.

2.6 Arama Sistemlerin Çıktılarını Birleştirmek için Yapılar

Lee'nin 1997'de arama sistemlerinin çıktılarını birleştirmek için tanımladığı stratejiler, basit ve deneyimsel bir yapıda ortaya konulmuştur. Sistem çıktıları mevcut stratejilerden biri kullanılarak normalleştirilmiş ve birleştirilmiş benzerlik değerleri olarak sayılmıştır. Belirli normalleştirme ve birleştirme stratejileri, sadece deneysel kanıtlara dayalı olarak seçilmektedir ancak min ve max gibi bazı birleştirme stratejileri biçimsel görüşlerle ispat edilebilmektedir.

Hull et al. 1996'da hem çeşitli birleştirme stratejileri hem de sınıflandırıcı çıktısının birleştirmesi yapısında ki genel durum üzerinde deneysel çalışmalarda bulunmuşlardır. Koşullu olarak bağımsız (given relevance (R) non relevance) n adet sınıflandırıcının $C_1,\ldots,C_n$ sonuçlarını birleştirmek için bir formül türetmişlerdir.

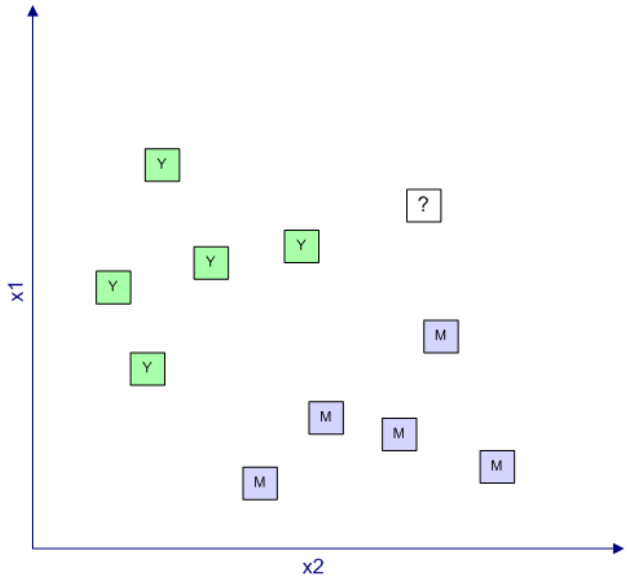
$$\log O(R \mid C_1,...,C_n) = \sum_{i=1}^{n} \log O(R \mid C_i) - (n-1) \cdot \log O(R) \quad (2.6.1)$$

3 – MAKİNE ÖĞRENMESİ

3.1 Makine Öğrenmesi

Günümüzde, bilgi teknolojilerinin hızlı gelişimi sayesinde büyük oranda faydalı ve faydasız bilgi kayıt altında tutulabilmektedir. Bu kayıt işlemleri bilgisayar ve bilgisayarlı sistemlerin girdiği birçok alanda yapılmaktadır. Toplanan tüm veriler toplandıkları alana yönelik amaçlar ile analiz edilir ve veriler üzerinden sonuçlara ya da önemli yeni verilere ulaşılmaya çalışılır. Çok büyük miktarlarda olan bu verilerin analizini yapmak pek de mümkün değildir. Günümüze kadar geliştirilen ve geliştirilmeye devam eden makine öğrenme yöntemleri bu problemi çözmeyi amaçlar. Makine öğrenme yöntemleri geçmiş de elde edilmiş veriyi alır, bu veriyi kullanarak, ona en uygun modeli belirler. Yöntemler gelen yeni verileri alıp belirlenen veriye göre analiz eder ve buradan sonuç, sonuçlar veya yeni bilgiler elde eder. Büyük miktarda veri içerisinden yeni yararlı bilgi elde edilmesi işlemine veri madenciliği de denilmektedir. Makine öğrenme yöntemlerini kullanıldığı alanlara göre sınıflandırmak mümkündür.

1. Sınıflandırma: Geçmiş bilgilerin hangi sınıflara ait olduğu verildiğinde yeni gelen verinin hangi sınıfa dahil olduğunun bulunması işlemidir. Örnek olarak bir kişiye ait iki tahlil sonucuna göre kişinin hasta olup olmadığı belirlenmeye çalışılırsa önceki hasta ve sağlam kişilerin tahlil sonuçları kullanılır. Şekil 3.1.1'de kişilere ait tahlil sonuçları x1 ve x2 eksenleriyle, gösterilmiştir. Şekilde mavi olarak işaretli kişiler hasta, yeşil ile işaretli olan kişiler ise sağlam kişilerdir. Hasta olup olmadığı merak edilen kişinin tahlil sonucu beyaz ile gösterilmiştir.



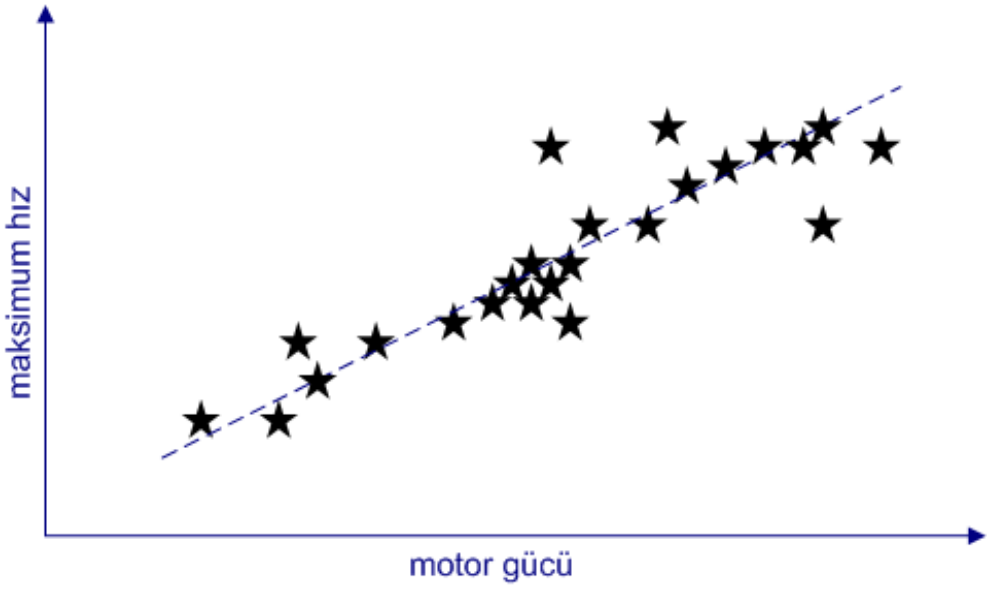
Şekil 3.1.1 – Sınıflandırma Problemi

2. Kümeleme: Geçmiş bilgilerin sınıflarının veya etiketlerinin bilinmediği ya da verilmediği durumlarda verilerden birbirine benzerlerin yer aldığı kümelerin bulunması işlemidir.

3. Regresyon (eğri uydurma): Geçmiş bilgilere karşılık gelen sınıflar yerine sürekli değerlerin yer aldığı problemlerdir. Şekil 3.1.2'deki grafikte motor güçleri verilen araçların maksimum süratleri tahmin edilmek istenmektedir. X ekseni araçların motor güçlerini, Y ekseni ise maksimum hızları göstermektedir. Bu tür problemlerde giriş ile çıkış arasındaki kesikli çizgilerle gösterilen fonksiyon eğrisi bulunmaya çalışılır. Grafikte verilen yıldızlar araçların motor güçlerine karşılık gelen maksimum hızlarını, düz çizgi ise bir regresyon metoduyla bulunan fonksiyonu göstermektedir. Bu fonksiyon sayesinde motor gücü bilenen bir aracın maksimum hızı da tahmin edilebilmektedir.

4. Özellik Seçimi ve Çıkarımı: Veriye ait birçok özellikten verinin kümesini, sınıfını ya da değerini belirleyen özelliklerinin hangileri olduğu bilinmeyebilir. Bu durumlarda tüm özellik kümesinin bir alt kümesi seçilir ya da bu özelliklerin birleşimlerinden yeni özellikler elde edilir.

5. İlişki Belirleme: Bir süpermarkette X ürününü alan müşterilerden %80'i Y ürününü de alıyorsa, X ürününü alıp Y ürününü almayan müşteriler, Y ürünün potansiyel müşterileridir. Müşterilerin alışveriş bilgilerinin bulunduğu bir veritabanında potansiyel Y müşterilerini bulma işlemi türündeki problemler ilişki belirleme yöntemleriyle çözülmektedir

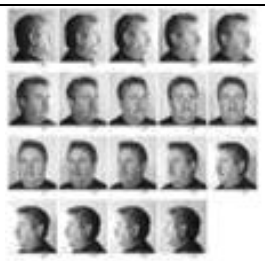
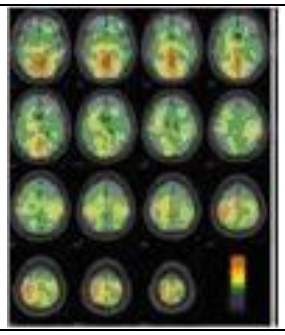
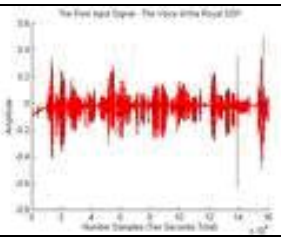


Şekil 3.1.2 – Regresyon

3.1.1 Günlük Hayatımızdaki Uygulamalar

Makine öğrenmesi uygulamaları bilgisayarların gelişmeleriyle birlikte günlük hayata önemli ölçüde katılmışlardır. El yazısı, imza, iris, parmak izi, yüz, ses tanımada yaygın kullanım alanlarına sahiptirler. Örneğin girdi olarak alınan insan sesi makine öğrenme yöntemleri sayesinde metne çevrilmektedir. Tıbbı verilerin yorumlanmasında, metin içeriğinin tespitinde, kamera kayıtlarının incelenip yorumlanmasında da yine önemli yere sahiptirler. Birkaç örnek uygulama Çizelge 3.1.1'de verilmiştir.

Çizelge 3.1.1.1 – Makine Öğrenmesi ve Günlük Hayattaki Birkaç Uygulaması

Girdi	Örnek	Çıktı
Kişinin yüz fotoğrafı.		Resimdeki kişinin kimliği kime ait? Sorusunun cevabı.
Tıbbı veri, tomografi.		Kişi hasta mı? Değil mi? Sorusunun cevabı.
Parmak izi.		Parmak izinin sahibinin kimliği.
Ses.		Ses kimin? Ses hangi nesneye ait? Sorularının cevapları.
Kamera kayıtları.		Olağan dışı bir durum var mı? Sorusunun cevabı.
Metin. Posta.		Metnin konusu nedir? Metnin yazarı kimdir? Posta gönderen kimdir? Sorularının cevapları.

3.2 Verilerin Sayısallaştırılması

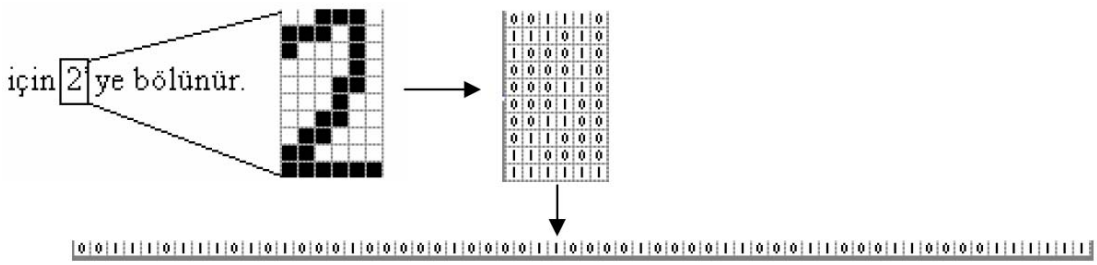
Makine öğrenme yöntemlerinin tamamına yakın kısmı sayılarla çalışmaktadır. Bu nedenle uygulamalarda ilk olarak yapılması gereken verilerin sayısallaştırılması işlemidir. Resim verisi, resmin her bir pikselinin renkli resimlerde kırmızı, yeşil ve mavi renklerine karşılık gelen 1-255 arası R, G, B değerleri, siyah-beyaz resimlerde 1–255 arası gri seviyesi kullanılarak sayısallaştırılır. Renkli resimler 3 adet, siyah-beyaz resimler ise 1 adet en * boy büyüklüğünde matrisle ifade edilmektedir. Metin verisi, metindeki harfler, heceler ve kelimeler genelde frekanslarına göre kodlanarak sayısallaştırılır. Hareketli görüntü diğer adıyla video verisi, resim bilgisine ek olarak resmin hangi resimden sonra geldiğini gösteren zaman bilgisini de içerir. Bu ek bilgi haricinde yapılan sayısallaştırma işlemi resim ile aynıdır. Ses verisi, ses, genlik ve frekansın zaman içinde değişimi kullanılarak sayısallaştırılır.

Örnek uygulama:

Optik karakter tanıma:
Sistemde, bir kitap fotokopisinin içindeki yazıların metne dönüştürülmesi için;
Öncelikle metindeki satırlar bulunur.
Her bir satırdaki harfler bulunur.
Her bir harfin resmi sınıflandırıcıya gönderilerek tanınır.

Sınıflandırıcı eğitimi:
Her bir harfin farklı yazı biçimleriyle yazılmış resimleri toplanır.
Resimler aynı boyuttaki matrislere çevrilir.

10 satır ve 6 sütundan oluşan 2 resmi, 1 ve 0'lardan oluşan 60 uzunluğundaki bir vektöre dönüştürülür, Şekil 3.2.1'de verilmektedir.



Şekil 3.2.1 – Metindeki ‘2’ Karakterinin Sayısallaştırılması

Bu şekilde tanınmak istenen harf için çeşitli yazı biçimleriyle yazılmış birçok örneği gösteren 60 boyutlu vektörler elde edilir. Bu uygulama için özellik sayısı 60’tır. Diğer bir deyişle örnekler 60 boyutlu bir uzayda gösterilmektedir. Eldeki 10 rakımına ait farklı yazı biçimleriyle yazılmış 10’ar resim olduğunda veri kümesi 10 örnek * 60 boyutlu bir matris olmaktadır. Bu matris eğitim ve test kümeleri oluşturmak için 2’ye bölünür.

3.3 Özellik Seçimi ve Çıkarımı

Eğitim bilgilerindeki her bir özellik teker teker ele alınmaktadır. Bunun için seçilen özellikle sınıf ya da sonucun birlikte değişimleri incelenir. Özellik değiştiğinde sınıf ya da sonuç ne kadar değişiyorsa o özelliğin sonuca o kadar etkisi vardır denilmektedir. Örneğin bir A bölgesinde doğan kişilerin hepsi hasta iken B bölgesinde doğan kişilerin hiçbiri hasta değil ise bu özelliğin yani doğum yerinin sonuca etkisinin büyük olduğu söylenebilmektedir.

Var olan özelliklerin doğrusal birleşimlerinden yeni bir özellik uzayının oluşturulması ve verilerin bu uzayda ifade edilmesine ise yeni özelliklerin çıkarılması denilmektedir.

3.4 Yöntemler

Makine öğrenme yöntemleri, birden fazla bağımlı ya da bağımsız değişkene sahip regresyon ve sınıflandırma işlemleriyle uğraşmak için bazı ileri seviye istatistiksel yöntemleri bünyesinde barındırmaktadır.

Bu yöntemler, sınıflandırma için “Acemi Bayes”i, regresyon ve sınıflandırma için “K-en yakın komşu”yu ve yine regresyon ve sınıflandırma için “Destek Vektör Makineleri”ni bünyelerinde barındırmaktadırlar. Bu teknikler üzerindeki detaylı tartışmalar Hastie, Tibshirani, ve Freedman (2001)’ın kitabında verilmiştir, ayrıca destek vektör makinelerine özel kapsamlı bir girişte Cristianini ve Shawe-Taylor (2000)’ın kitabında mevcut bulunmaktadır.

3.4.1 Acemi Bayes

Acemi Bayes, birincil olarak sınıflandırma işlemlerini gerçekleştirmek için iyi oluşturulmuş bir Bayes metodudur. Basit oluşunun yanında örneğin; bağımsız değişkenler, istatistiksel olarak da bağımsızdır, Acemi Bayes modelleri kolay kullanımı ve yorumlanabilirliği ile etkili sınıflandırma araçlarındandır. Acemi Bayes modeli, bağımsız uzayın boyutlarının fazla olduğu (örneğin; girdi değişkenlerinin sayısı) durumlar da (bu problem boyutların laneti olarak da bilinir) özellikle uygundur. Yukarıda verilen nedenlerden dolayı, Acemi Bayes metodu belirli durumlarda, diğer karışık sınıflandırma yöntemlerinden daha iyi performans göstermektedir. Normal, lognormal, gamma ve Poissonu kapsayan girdilerin koşullu dağılım modellemek için çeşitli yöntemler mevcuttur.

3.4.2 K-en yakın komşu

K-en yakın komşu metodu diğer istatistikî yöntemlerin tersine bellek tabanlı bir yöntemdir, eğitime ihtiyacı yoktur. İlk örneklere dayalı yöntemler kategorisinde bulunmaktadır. İşleyişi, 'birbirine yakın olan nesneler muhtemelen aynı kategoriye aittir' diyen sezgisel fikir üzerine kuruludur. Bundan dolayı K-en yakın komşu metodunda ön görüler, yeni veriyi çoğunluğun oyuyla tahmin etmek için kullanmak (sınıflandırma için) ve k-en yakın ilk örnekler kümesi üzerinde ortalama bulmak (bağlanım için), için ilk örnek örnekler kümesine dayanmaktadır.

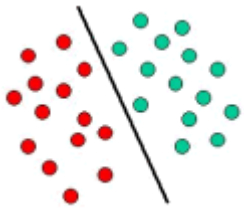
3.4.3 Destek Vektör Makineleri (DVM)

Vapnik tarafından geliştirilen bu yöntem günümüzde performansı sayesinde oldukça yaygın kullanılan bir yöntemdir. Temelde sınıfları birbirinden ayıran özel bir eğrinin bulunmasını amaçlar, eğitim verileriyle bu sınır eğrisi bulunduktan sonra test verileri sınırın hangi tarafında kaldıklarına göre sınıflandırılmaktadır. Destek vektör makineleri sınıflandırma ve eğri uydurma için kullanılan öğreticili öğrenme yöntemlerinin kullanıldığı bir kümeden oluşmaktadır. Genelleştirilmiş doğrusal sınıflandırıcılardan oluşan bir aileye aittir. Tikhonov düzenlemesinin özel bir hali olarak da düşünülmektedir.

Bu yöntem doğrusal olmayan karar sınırları oluşturarak bağlanım ve sınıflandırma yapar. Sınırları bulunan nitelik uzayının doğasından dolayı Destek Vektör Makineleri değişen karmaşıklıkta sınıflandırma ve bağlanım yaparlarken büyük ölçüde esneklik ortaya koymaktadırlar. Destek Vektör Makinelerinin birkaç çeşidi mevcuttur, doğrusal, çokterimli, RBF ve sigmoid.

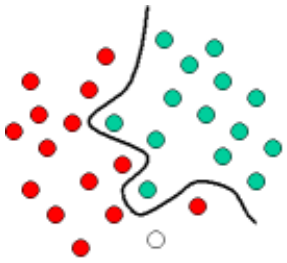
Destek Vektör Makinelerinin dayandığı temel kavram karar sınırlarını belirleyen karar düzlemleri olarak verilmektedir. Karar düzlemi bir küme içerisindeki nesneleri üyeliklerine göre sınıflara ayırır. Karar düzleminin şematik bir gösterimi Şekil 3.4.3.1'de yapılmıştır. Şekilde verilen örnekte nesneler yeşil ya da kırmızı sınıflara aittir. Şekilde gösterilen ayırıcı çizgi bir sınır belirlemektedir, çizginin sağ tarafında

kalan nesneler yeşil, sol tarafında kalan nesneler ise kırmızıdır. Belirlenen bu sınıra göre çizginin sağ tarafında düşen her kırmızı daire, yeşil olarak etiketlenecek böylece, yeni gelen nesnenin sınıfı yeşil olacaktır. Aynı durum kırmızı içinde geçerlidir, yani çizginin sol tarafına düşen nesneler kırmızı sınıfa ait nesneler olacaktır.



Şekil 3.4.3.1 – Karar Düzeninin Şematik Bir Gösterimi

Yukarıda verilen klasik bir doğrusal sınıflandırıcı örneğidir. Bir kümedeki nesneleri kendi gruplarına (bu örnekte yeşil ve kırmızı) bir çizgi ile ayıran sınıflandırıcıya doğrusal sınıflandırıcı denir. Ancak, pek çok sınıflandırma işlemleri böylesine basit değildir, en iyi sınıflamayı yapabilmek için çok daha karmaşık yapılara ihtiyaç duyulur. Örneğin, kullanılabilir örneklerden (eğitim aşamaları), yeni nesneleri doğru bir şekilde sınıflandırmak (test aşamaları).

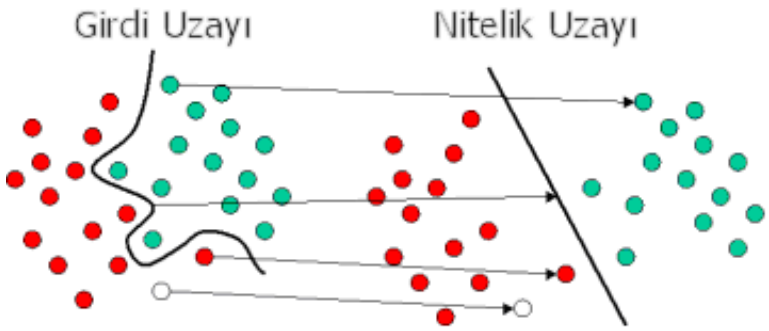


Şekil 3.4.3.2 – Hiper Düzlem Sınıflandırıcısı

Bu durum Şekil 3.4.3.2'de tarif edilmiştir. Doğrusal sınıflandırıcı ile kıyaslandığında görülmektedir ki, yeşil ve kırmızı nesneleri tam bir şekilde ayrılması için bükümlere ihtiyaç duyulmaktadır (bu yapı düz bir çizgiden daha karmaşıktır). Farklı sınıflara üye nesneleri ayırt etmek için çizilen ayırıcı çizgilere dayalı sınıflandırma işlemleri,

hiper düzlem sınıflandırıcıları olarak bilinmektedir. Destek Vektör Makineleri için özellikle bu tip işlemlerin üstesinden gelmek uygundur.

Destek Vektör Makinelerinin arasında yatan temel fikir Şekil 3.4.3.3'de gösterilmiştir. Şekilde orijinal olarak sol tarafta verilen nesneler, eşleştirilmiş olarak gösterilmiştir. Diğer bir değişle nesneler çekirdek olarak bilinen bir küme matematiksel fonksiyon kullanılarak yeniden düzenlenmiştir. Nesneleri yeniden düzenleme işlemi, eşleştirme (mapping) ya da diğer adıyla dönüşüm olarak bilinmektedir. Şekilde de gösterildiği gibi yapılan bu yeni düzenlemede, eşleştirilmiş (şekilde sağ tarafta yer alan) nesneler, doğrusal olarak ayrılabilirdir ve bundan dolayı karmaşık bir eğri (şekilde sol tarafta görüldüğü gibi) oluşturmaktansa, yapılması gereken tek şey, yeşil ve kırmızı nesneleri ayıracak en uygun çizgiyi bulmak olacaktır.



Şekil 3.4.3.3 – Yeniden Düzenleme, Eşleştirme İşlemi

3.4.3.1 DVM için Teknik Notlar

Destek Vektör Makineleri (DVM) her şeyden önce çok boyutlu uzayda hiper düzlemler oluşturarak farklı sınıf etiketlerini birbirinden ayırarak sınıflandırma işlemini gerçekleştiren bir sınıflandırma metodudur. Destek Vektör Makineleri hem bağlanım hem de sınıflandırma işlemlerini destekler ve birçok sürekli ve kategorik değişken kullanabilir. Kategorik değişkenler için durum değerleri ile 0 ya da 1 olacak şekilde bir adet yalancı (dummy) değişken tanımlanır. Dolayısıyla, üç

seviyeden meydana gelen, say(A,B,C), bir kategorik bağımlı değişken, üç yalancı değişkenden oluşan bir küme ile gösterilir:

A: {1 0 0}, B: {0 1 0}, C: {0 0 1}

En iyi hiper düzlemi oluşturabilmek için, DVM tekrarlı bir eğitim algoritmasını işletir, bu algoritmanın kullanım amacı hata fonksiyonunu en aza indirmektir. Bu hata fonksiyonunun biçimine göre, DVM modelleri dört ayrı gruba sınıflandırılır.

- Sınıflandırma DVM Tip 1 (C-DVM sınıflandırıcısı olarak da bilinir)
- Sınıflandırma DVM Tip 2 (nu-DVM sınıflandırıcısı olarak da bilinir)
- Bağlanım DVM Tip 1 (epsilon-DVM bağlanım olarak da bilinir)
- Bağlanım DVM Tip 2 (nu-DVM bağlanım olarak da bilinir)

Sınıflandırma DVM

Sınıflandırma DVM Tip 1

Bu tip DVM için, eğitim hata fonksiyonun en aza indirgenmesini gerektirir.

$$\frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \ , \qquad (3.4.3.1.1)$$

sınırlamalara bağlı olarak,

$$y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \qquad ve \qquad \xi_i \geq 0, \qquad i = 1,..., N \qquad (3.4.3.1.2)$$

C kapasite sabiti, w katsayılar vektörü, b bir sabit ve ξ_i ayrılamayan veriye (girdi) karşılık gelecek parametrelerdir. İndeks i, N eğitim durumlarını etiketler. Dikkat edilmelidir ki, $y \in \pm 1$ sınıf etiketleridir ve x_i bağımsız değişkenlerdir. Çekirdek ϕ girdi olarak alınan (bağımsız) veriyi, nitelik uzayına dönüştürmek için kullanılır. C değeri büyükçe, hata artar. Bundan dolayı, C özenle seçilmelidir.

Sınıflandırma DVM Tip 2

Sınıflandırma DVM Tip 1'in aksine, Sınıflandırma DVM Tip 2 modeli hata fonksiyonunu minimuma indirir:

$$\frac{1}{2} w^T w + v\rho + \frac{1}{N} \sum_{i=1}^{N} \xi_i \qquad (3.4.3.1.3)$$

Sınırlamalara bağlı olarak:

$$y_i (w^T \phi(x_i) + b) \geq \rho - \xi_i, \xi_i = 0, i = 1,..., N \quad ve \quad \rho \geq 0 \qquad (3.4.3.1.4)$$

Regresyon DVM

Regresyon DVM için bağımlı değişken y için, bağımsız değişkenler x'den oluşan küme üzerindeki fonksiyonel bağımlılığı hesaplanmalıdır. Diğer regresyon problemlerindeki gibi, bağımlı ve bağımsız değişkenler arasındaki bağıntının gerekirci f fonksiyonu artı ilave gürültünün toplamı ile verildiği kabul edilir:

$$y = f(x) + gürültü \qquad (3.4.3.1.5)$$

Daha sonra yapılması gereken iş, DVM'nin daha önce tanışmadığı durumları doğru bir şekilde önceden haber vermesi için f için bir fonksiyonel formun bulunmasıdır.

Bunu yapmak için DVM modelini örnek bir küme üzerinde eğitmektir. Örneğin, daha önceden sınıflandırmada değinildiği gibi hata fonksiyonunu ardışık olarak en uygun şekle getirecek bir süreç gerekir. Bu hata fonksiyonunun tanımına göre, iki tip DVM modeli kabul edilmektedir.

Regresyon DVM Tip 1

Bu tipteki bir DVM için

$$\begin{aligned} & w^T \phi(x_i) + b - y_i \le \varepsilon + \xi_i^\bullet \\ & y_i - w^T \phi(x_i) - b_i \le \varepsilon + \xi_i \\ & \xi_i, \xi_i^\bullet \ge 0, \qquad i = 1,..., N \end{aligned} \tag{3.4.3.1.6}$$

tesiri altındaki indirgenen hata fonksiyonu:

$$\frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i + C \sum \xi^\bullet{}_i \tag{3.4.3.1.7}$$

şeklindedir.

Regresyon DVM Tip 2

Bu tipteki bir DVM için

$$\begin{aligned} & (w^T \phi(x_i) + b) - y_i \le \varepsilon + \xi_i \\ & y_i - (w^T \phi(x_i) - b_i) \le \varepsilon + \xi_i^\bullet \\ & \xi_i, \xi_i^\bullet \ge 0, i = 1,..., N, \varepsilon \ge 0 \end{aligned} \quad , \tag{3.4.3.1.8}$$

tesiri altındaki indirgenen hata fonksiyonu

$$\frac{1}{2}w^T w + C\left(v\varepsilon + \frac{1}{N}\sum_{i=1}^{N}(\xi_i + \xi_i^{\bullet}) \right), \qquad (3.4.3.1.9)$$

olacaktır.

Çekirdek Fonksiyonları

Destek Vektör Makineleri modelleri olarak kullanılan birtakım çekirdekler vardır. Doğrusal, polinom, radyan taban fonksiyonu (RTF) ve sigmoid.

$$\phi = \begin{Bmatrix} x_i * x_i & lineer \\ (\gamma x_i x_j + katsayı)^{derece} & polinom \\ \exp(-\gamma \mid x_i - x_j \mid^2) & RTF \\ \tanh(\gamma x_i x_j + katsayı) & sigmoid \end{Bmatrix} \qquad (3.4.3.1.10)$$

RTF, Destek Vektör Makinelerinde çekirdek tiplerinden uzun zamandır yaygın şekilde kullanılmaktadır. Bunun başlıca nedeni reel x-eksenin tüm menziline karşın belirli bir yerle sınırlanan ve sonlu yanıtlarıdır.

4 – PROTEİN DİZİLİMLERİ GÖSTERİMİ

4. Protein Dizilimlerinin Gösterimi

Proteinlerin yapı ve fonksiyon sınıflarına göre otomatik olarak kategorilere ayrıştırılması, hesaplamalı biyolojinin önemli problemlerindendir. Günümüzde proteinleri, önceden bilinen sınıflara atamak için geliştirilmiş çoğu yöntem, yapay sinir ağları, Bayes sınıflandırıcıları ve destek vektör makineleri gibi makine öğrenme teknikleri üzerine kurulmuştur. Sınıflandırma teorisine göre, sınıflandırıcıya verilecek girdi, sabit uzunlukta nitelik vektörlerinin bir koleksiyonu şeklinde olmalıdır. Hem öğretim hem de test için kullanılacak verilerin bu koşulu sağlaması şarttır. Proteinler, farklı sayılarda aminoasidin bir zincir şeklinde sıralanmasından oluşan yapılar olduğu için, dizilim bilgisinin sınıflandırıcıya doğrudan verilmesi mümkün değildir. Bu sebepten, proteinleri sabit uzunlukta nitelik vektörleri şeklinde göstermeye ihtiyaç duyulmaktadır.

4.1. Birleştirme Gösterimi

Protein dizisi gösterimi için birçok yöntem önerilmiştir. Protein dizilerinin gösteriminde, belki de en geniş şekilde kullanılan yöntem aminoasit birleştirme metodudur. Bu yöntemle, tüm proteinler, 20 boyutlu nitelik vektörleri ile gösterilmiştir. Her boyut ilgili aminoasidin dizilimi içerisinde bulunma yüzdesidir. Basit doğasından dolayı aminoasit birleştirme metodu çok problem üzerinde başarı ile uygulanmıştır ve halen hepsi-alfa, hepsi-beta, alfa-beta gibi yapısal sınıfların bulunmasında yeterli olduğu kabul edilmektedir (Bahar et al., 1997). Bu yöntemdeki esas problem, protein dizisi boyunca aminoasitlerin yerel sıralarını görmezden gelmesidir. Yerel aminoasit sıralarını da göz önünde bulundurmak üzere tek gösterimi ya da aminoasit birleşimi ile vektör birleşimi şeklinde, dipeptit birleşimi kullanılmaktadır. Dipeptit birleştirme, 400 boyutlu bir vektördür. Bu vektörün her bir boyutu da 2-uzunluklu aminoasit zincirinin, protein dizisi içerisindeki kesrine karşılık gelmektedir. Aynı zamanda dipeptit birleştirme de birçok problemde kullanılmıştır.

Bu yöntem birçok problemde aminoasit birleştirmeye göre daha iyi sonuçlar vermiştir. Ancak, uzun peptit zincirlerindeki aminoasitlerin dizilimi hakkında hala bilgiden yoksun olmaktadır.

Dizi içeriği üzerine kurulmuş başka bir protein gösterim düzeni ise temelde proteinleri fizikokimyasal özelliklerini kullanmaktadır. Ding ve Bubchak (2001) altı özellik kullanmışlardır, bunlar; aminoasitler, tahmin edilen ikincil yapı, hidrofobisite, normalleşmiş van der Waal hacmi, polarite ve polarize olup olmama durumudur. Birincil dizilimden her nitelik için üç temel tanımlayıcı üzerinden yüzde belirleyici özellikler çıkartmışlardır; üç bileşenin birleşim yüzdesi (örneğin; kutupsal, nötr ve hidrofobisite içinde hidropobik aminoasitler), değişim frekansları (kutupsaldan nötre, nötrden hidrofobik vb.) ve bileşenlerin dağılım desenleri olacak şekildedir.

Bu yaklaşımı çok sınıflı protein katmanlarını tanımak için, destek vektör makineleri ve yapay sinir ağları kullanarak uygulamışlardır. Benzer bir yaklaşımda protein fonksiyonların sınıflandırılmasında, destek vektör makinelerini kullanarak gerçekleştirilmiştir (Chai et al. 2003). Bahsin ve Raghava (2004) 33 farklı fizikokimyasal nitelik kullanmışlardır ve 33 nicel değere sahip bir girdi vektörü oluşturmuşlardır. Vektördeki her bir değer proteinde saklı, ayrı bir niteliğe ait ortalama değeri olmak üzere alt hücrelerin yerlerini tespit edebilmek için kullanılmıştır.

Motif tabanlı protein gösterimi de yine protein zincirinin içeriği ile bağlantılıdır. Bu yöntemlerde mümkün olan tüm aminoasit ve dipeptit kombinasyonlarının aratılmasından ziyade, önceden çıkarılan, yapısal olarak bilinen ya da önemli olduğu ifade edilen zincir desenleri, özel olmayan ve kullanılmaya hazır PROSITE (Falquet et. al. 2002), I-SITES (Bystroff ve Baker, 1998), BLOCKS (Henikoff et al. 1999), eMOTIF (Huang ve Brutlag, 2001) veritabanları kullanılarak, zincir içerisinde aratılmıştır. Bir boolean değeri, aranılan motifin zincir içerisinde var olduğunu göstermekte ve nitelik vektörünün her boyutuyla ilişkilendirildiğini belirtmektedir (Ben-hur ve Brutlag, 2003; Hou et al. 2003). Bu durumda nitelik vektörünün boyutlarının sayısı, veritabanında kullanılan uygun motiflerin sayısına eşit olmaktadır. Motif tabanlı yaklaşımlar, olağanüstü başarılarına rağmen, sonuç nitelik

vektörleri çok sık şekilde, çok fazla seyrek bilgi içermektedirler. Verilen çoğu motifin, hedef zincire eklenmesinin dahi gerek olmamaktadır.

Daha sonraki bir çalışmada (Oğul ve Mumcuoğlu, 2007), proteinleri n-peptit bileşenlerine göre tanımlanmaktadır, ancak bu aminoasit ve dipeptit bileşenleri için bir genellemedir. N= "1,2,3,…" olmak üzere N'nin her değeri için, uygun düşen nitelik vektörü, her olası n uzunluğundaki her altdizinin zincir içinde olup olmadığının kesrini vermektedir. Örneğin, nitelik vektörü n=1 için aminoasit birleştirmesini ve n=2 için dipeptit birleştirmesini ima etsin. Önceden de değinildiği gibi, n-peptit birleştirmesi bir çok problemde n=1 ve n=2 ile beraber yaygın olarak kullanılmıştır. Ancak daha uzun n-peptit birleştirmeleri için zaman ve bellek gereksinimlerinin, n'nin aldığı değer ile üssel olarak artışından, verimli bir şekilde kullanılamamaktadır. N-peptit kombinasyonuna eş gelen nitelik vektörünün boyut sayısı 20^n olmaktadır. Bundan dolayı da eğitim adımının bellek alanı karmaşıklığı $O(k20^n)$ eşit olmaktadır, burada k ile verilen eğitim setindeki proteinlerin toplam sayıdır. Tüm bunlar öngörmektedir ki, n değişkeninin küçük değerleri için çok-boyutlu nitelik vektörünün biçimlenmesini, sıradan bellek kaynaklarını kullanarak gerçekleştirebilmek, sistemi zorlayacaktır. Bu problemin üstesinden gelebilmek için, aminoasit alfabesi n'in artan değerleri için yavaş yavaş azaltılmaktadır. Böylece her n-peptit birleştirmesi için sonuç vektörü, t sabit değerinden daha az sayıda boyuta sahip olmaktadır, uzay karmaşıklığını da üst seviye olarak O(kt) değişmektedir. Bir başka deyişle, n-peptit birleştirmeleri için r sayısında bir alfabe kullanılmaktadır ve $r^n < t$ koşulu sağlanmaktadır. Sadece verimli bir uzay karmaşıklığı sağlamaktan ziyade uzun n-peptit zincirlerinde mümkün olabilecek uyuşmazlıkları değerlendirebilmektedir. Bu durum proteinlerin gelişiminde doğal bir süreçtir. Çizelge 4.1.1'de azaltılmış aminoasit alfabe büyüklüklerini ve n-peptit birleştirmeler inşa edilirken t'nin farklı değerleri için sonuç nitelik vektörünün sonuçları verilmektedir.

Çizelge 4.1.1 – Aminoasit alfabe büyüklükleri ve değişken eşik değerleri için n-peptit birleştirmeye karşılık gelen özellik vektörü boyutları.

n	t=1000		t=5000		t=10000	
	alfabe boyutu	vektör boyutu	alfabe boyutu	vektör boyutu	alfabe boyutu	vektör boyutu
1	20	20	20	20	20	20
2	20	400	20	400	20	400
3	10	1000	15	3375	20	8000
4	5	625	8	4096	10	10000
5	3	243	5	3125	6	7776
6	3	729	5	4096	4	4096

N-peptit birleştirmelerinin kullanımındaki bir başka problem ise proteinlerin sayısallaştırılmasında zaman karmaşıklığının üssel olarak artışıdır. M uzunluğundaki bir protein zincirinde mümkün olan tüm n-peptitleri arayan sade bir algoritmanın zaman karmaşıklığı $O(m20^n)$ olmaktadır.

Alfabeleri azaltmak amacıyla Murphy et al. (2000) tarafından sağlanan aminoasit gruplama metodu kullanılmaktadır. Bu alfabeler, belirli BLOSUM matrisleri (Henikoff ve Henikoff, 1992) üzerindeki bilgi temel alınarak istatistiksel teknikler kullanılarak üretilmiştir ve çok iyi bilinen biyokimyasal aminoasit sınıfları tarafından onaylanmıştır. Aminoasitleri gruplamak için geçerli işlem iki adımdan oluşmaktadır. Birinci adımda, tüm aminoasit çiftleri için, yer değiştirme matrisi elemanları arasındaki bağıntı katsayıları ayrı ayrı hesaplanmaktadır. İkinci adımda ise, en yüksek bağıntı katsayısına sahip iki aminoasit beraber gruplandırılmaktadır, daha sonra, bir sonraki bağıntıya sahip çift, elemanlarından biri grubun içinde ise ilk gruba eklenmektedir ya da yeni bir gruba ayrılmaktadır. Eğer aminoasit her hangi bir gruba üye değil ise, aynı işlem tüm aminoasitler istenilen sayıda gruba ayrılana

dek tekrarlanmaktadır. Çizelge 4.1.2 bu gruplama senaryosuyla azaltılmış alfabelere örnekler vermektedir.

Çizelge 4.1.2 – Azaltılmış Aminoasit Alfabeleri.

Boyut	Alfabe
20	**L V I M C A G S T P F Y W E D N Q K R H**
15	**(L V I M) C A G S T P (F Y) W E D N Q (K R) H**
8	**(L V I M C) (A G) (S T) P (F Y W) (E D N Q) (K R) H**
6	(L V I M) (A G S T) (P H C) (F Y W) (E D N Q) (K R)
5	(L V I M C) (A G S T P) (F Y W) (E D N Q) (K R H)
4	(L V I M C) (A G S T P) (F Y W) (E D N Q K R H)
3	(L V I M C A G S T P) (F Y W) (E D N Q K R H)
2	(L V I M C A G S T P F Y W) (E D N Q K R H)

4.2. İkili Benzerliklere Dayalı Deneysel Gösterim

2003 yılında Liao ve Noble, proteinleri sabit uzunlukta vektörler olarak göstermek için yeni ancak oldukça basit bir yöntem önermişlerdir. Önerilerinde, her protein benzerlik puanı kümeleri ile gösterilir. Bu öneri aminoasit dizilimlerinin incelenen özelliklerini göz önünde bulundurmadan, bilinen protein kümesi üzerinden oluşturulan deneysel nitelik haritalarını kullanır. Veri kümesindeki her P_x proteini $\varphi(P_X)$ olarak aşağıdaki denklem ile sayısallaştırılır.

$$\varphi(P_x) = \left[S(P_x, P_1), S(P_x, P_2), ..., S(P_x, P_n)\right] \qquad (4.2.1)$$

Burada P_i ile gösterilen etiketlenmiş protein kümesinde *i*. protein dizilimidir, *n* etiketlendirilen kümedeki proteinlerin toplam sayısıdır ve $S(P_x, P_i)$ ise P_x ile P_i

arasındaki hizalama puanıdır. Liao ve Noble çalışmalarında, dinamik programlama ile dizi hizalama, uzak homoloji tespiti testlerinde iki dizilim arasında benzerlik sonucu çıkarımının en doğru ve tam yolu olduğunu göstermişlerdir. Dinamik programlama temelli hizalamanın dizilim benzerliğinde en uygun sonucu verdiğini önceki makalelerinde de gösterdikleri için bu sonuç da beklenmedik değildir. Ancak dinamik programlama algoritmasının hesaplamadaki verimsizliği nedeni ile sistem çok yavaş çalışmaktadır. Bu sebepten dolayıdır ki sonraki çalışmalarda bu yöntem uygulama alanı bulamamıştır.

Sonraki çalışmalarda (Ogul ve Mumcuoğlu, 2005, 2006), protein dizilimleri aralarında benzerliği ortaya çıkartmak için iki farklı yöntem önerilmiştir. Önerilen yöntemler, Liao ve Noble tarafından ortaya konan deneysel nitelik gösterim metodunda benimsenmiştir. Denemelerin temel amacı, bu güçlü stratejiyi gerçek hayattaki uygulamalar üzerinde pratik şekilde kullanabilmektir.

4.2.1 Dizi Hizalama

İki protein dizisi arasındaki benzerliğin derecesi, proteinlerin hizalanışları ile belirlenmektedir. Dizi hizalama problemi önceki üç yüzyıl süresince çok geniş bir şekilde çalışılmıştır (Needleman ve Wunch, 1970; Hirschberg, 1977; Smith ve Waterman, 1981; Myers ve Miller, 1998; Myers, 1991; Lipman ve Pearson, 1985; Altschul et al., 1990; Delcher et al., 1999).

4.2.1.1 İkili Hizalama

İkili dizi hizalama, iki diziyi aralarında belli uyumsuzluklara karşın karşılaştırma problemidir. $A = a_1a_2\ldots a_m$ ve $B = b_1b_2\ldots b_n$ dizileri verilmiş olsun. A ve B dizilerinin hizaları, *mutasyon* adıyla A ve B'yi dönüştüren ve toplam işlemlerin toplam masrafını en aza indirgeyen aynı zamanda benzerliği ise en yüksek dereceye çıkartan bir küme evrensel operasyon içermektedir. Bu operasyonlar, *yer değiştirme* operasyonu bir harfin diğerinin yerini alması, *silme* bir harfi ortadan

kaldırma, *ekleme* yeni bir harf katma olabilirler. Ekleme ve silme birbirinin çifti olduğundan dolayı eklemeler/çıkarmalar adı da yaygın bir şekilde kullanımdadır.

A C – – B C D D D B
| | | |
– C A D B – D A D –

Şekil 4.2.1.1.1 – Örnek Hizalama

Kabaca, *A* ve *B* dizilerini hizalarken, ilk olarak seçilmiş boşlukların *A* ve *B*'nin başına ya da içine eklenir böylece diziler birbirine denk olur, daha sonra ortaya çıkan iki diziyi birbiri üzerine eklenir ve her dizi üzerindeki karakter veya boşluk diğer dizideki karakter ya da boşluklar ile eşleşir. Şekil 4.2.1.1.1'de "ACBCDDDB" ve "CADBDAD" dizilerinin olası bir hizalaması verilmiştir. (–) işareti boşluk, (|) işareti ise eşlik olarak sembolize edilmiştir.

Biyolojik olarak dizi hizalamanın iki farklı biçimi mevcuttur, bunlardan biri global hizalama diğeri ise yerel hizalamadır. Global hizalama, benzerlik (ya da uzaklık) için her iki dizinin tüm uzunluğu üzerinden puanı en uygun şekle getirmektedir. Her iki dizide birbirine benzer ve kabaca aynı uzunlukta ise uygundur. Problem ilk olarak Needleman ve Wunsch (1970) tarafından formüle edilmiştir. *A* ve *B* olarak verilen iki dizi için $|A|=m$ ve $|B|=n$, m ve n kabaca aynı değere sahip olsunlar. $\sigma(a_k, b_k)$, a_k karakterinin b_k karakteri ile beraber hizalama puanı, *V(i,j)*, $a_1a_2\ldots a_i$ ve $b_1b_2\ldots b_j$ $(0 \le i \le n, \quad 0 \le j \le m)$ için en uygun hizalama puanı olsun. Buna göre,

$$V(i,j) = \max\begin{Bmatrix} V(i-1, j-1) + \sigma(a_i, b_j) \\ V(i, j+1) + \sigma(a_i, -) \\ V(i-1, j) + \sigma(-, b_j) \end{Bmatrix} \qquad (4.2.1.1.1)$$

– boşluk karakteri anlamına gelmektedir. *V(i,j) n x m* matrisi şekilde, *i* ve *j* indisleriyle gösterilir. En yüksek puana sahip hizalama, matris üzerinde geriden adım adım giderek bulunur.

Hizalamanın zaman ve alan karmaşıklığı *O(mn)* olmaktadır; algoritmalar matrisi doldurmak için *O(mn)* zamana ve *O(m+n)* alana, matris üzerinde geriden adım adım gelmek için ise *O(mn)* Alana ihtiyaç duymaktadır. Dizi hizalama için doğrusal-alan algoritmasını ilk olarak Hirschberg ortaya çıkartmıştır (Hischberg, 1977), daha sonradan Myers ve Miller tarafından düzeltme çizgileri eklenerek genişletilmiştir (Myers ve Miller, 1988). Ancak, alan ihtiyacını azaltmak, zaman karmaşıklığının artmasına neden olmuştur.

Yerel hizalama, benzer alt bölgeleri bularak, puanı bu alanlar için en uygun şekle getirmektedir. Karşılaştırılacak dizilerin birbirlerine benzerlikleri bilinmiyor ise uygundur. Smith ve Waterman algoritması en yaygın şekilde kullanılan yerel hizalama algoritmasıdır (Smith ve Waterman, 1981). Global hizalamaya benzerdir, *V(i,j)* aşağıdaki denklem ile gösterilir;

$$V(i,j) = \max\begin{Bmatrix} V(i-1,j-1)+\sigma(a_i,b_j) \\ V(i,j+1)+\sigma(a_i,-) \\ V(i-1,j)+\sigma(-,b_j) \\ 0 \end{Bmatrix} \qquad (4.2.1.1.2)$$

Yerel hizalamada ki en önemli nokta, karışık hizalama negatifi için beklenen değerin derecelendirilmesidir. Böylece karışık hizalamalar ve diğer uyuşmazlıklardaki düzeltmeler, yol puanını düşürür. En iyi yerel hizalama, bütün olarak saklanan *V(i,j)* matrisinde maksimum elemanı tespit edildikten sonra yol puanı sıfıra düşene kadar geriden başlayarak adım adım matrisi gezerek bulunur.

4.2.1.2 Hizalama Buluşları (Hizalama Sezgileri)

Diziler kısa olduğunda dinamik programlama metodları uygulanabilirdir. Ancak, bir diziyi, veritabanı içerisindeki yüzlerce dizi ile karşılaştırmak için yetersizlerdir. Bu problemi çözmek için birçok deneyimsel yaklaşım kullanılmıştır. FASTA (Lipman ve Pearson, 1985) veya BLAST (Altschul et al., 1990) bunlara örnek teşkil etmektedir.

FASTA, iki dizi arasındaki bire bir eşleşmeleri göz önünde bulundurur. Eğer bu tip belirleyici sayıda eşleşme bulunursa, FASTA en uygun hizalamayı hesaplamak için dinamik programlama algoritması kullanır. BLAST ise temeli benzer bir düşünce üzerine kurulu başka bir deneyimsel yaklaşımdır. Belli ve sabit uzunluktaki *k* için aralıksız hizalama üzerine odaklanır. Bire bir eşleşmelere bakmaktan ziyade, BLAST benzerlik ölçümü için bir puan fonksiyonu kullanır. Proteinlere mahsus olarak, eşleşmiş aralıksız parçalar ve yüksek benzerlik puanı, fonksiyonel benzerlik kısımlarını belirtir. Verilen *t* eşik değeri için, BLAST kullanıcıya puanı *t* değerinden daha yüksek ve sorgu dizisi ile eşleşmiş kısımları olan tüm veritabanı kayıtlarını bildirir.

4.2.1.3 Proteinler için Puan Matrisleri

Proteinler için $\sigma(a_i, b_j)$ puanları basitçe eşlenenler +1, eşlenmeyenler –1 ve eklemeler/çıkarmalar –2 şeklinde tanımlanır. Ancak, içeriğe bağlı olarak bazı değişiklikler diğerlerine göre daha akla yatkındır. Amino asidin benzer özelliklere (büyüklük, yük, vb.) sahip bir aminoasitle değişimi, karşıt özelliklere sahip bir aminoasit ile değişimine oranla daha mümkündür. Farklı değişim imkânlarını tarif etmek için çeşitli puanlama tasarıları önerilmiştir. Bunların en popüleri PAM (point accepted mutations) matrislerdir. 1970'lerde liderliği M. Dayoff tarafından yapılan bir araştırma grubu, aminoasit dizilerinin evrimi üzerinde çalışmış ve yalnız başına aminoasitlerin değişim olasılıklarını göstermek için bazı matrisler oluşturmuşlardır (Dayoff et. al., 1978). PAM veya PAM1 aminoasitin %1'lik kısmının değişime uğraması için gereken zaman uzunluğudur. PAM bir milyar yıl olarak tahmin edilmektedir. PAM1 matrisi satır ve sütunlarında aminoasit başlıklarına sahip bir matristir, matrisin hücreleri ise bir PAM süresinde meydana gelen değişiklik sayısına veya yüz aminoasit için bir değişime tekabül etmektedir. PAM1 matrisi üzerinde bazı genişlemeler mümkündür. Örneğin, bir PAM70 matrisi, 70 PAM sürede meydana gelen değişimler hakkındaki puanlama bilgisini içerir. Herhangi bir PAM<X> matrisi PAM1 matrisinden X'e yükseltilerek bulunabilir. Şekil 4.2.1.3.1'de PAM70 matrisine bir örnek gösterilmektedir. Matriste de görüldüğü gibi E ile D'nin değişimi E ile C'nin değişiminden daha uygundur, E-D puanı 3, E-C puanı -9'dan

daha yüksektir. Uygun PAM matrisini belirlemek, matrisin uygulanacağı veriye bağlıdır. Eğer diziler birbirlerine ileri seviyede benziyorlarsa, küçük indise sahip bir PAM matrisi tercih edilir. Diğer türlü yüksek indisli biri seçilmelidir. Daha sonraları Henikoff ve Henikoff tarafından 1992 yılında yer değiştirme matrislerinin BLOSUM serileri ortaya konmuştur, uzaktan akraba protein dizileri için daha hassas olduğu kabul edilir.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-4	-2	-1	-4	-2	-1	0	-4	-2	-4	-4	-3	-6	0	1	1	-9	-5	-1
R	-4	8	-3	-6	-5	0	-5	-6	0	-3	-6	2	-2	-7	-2	-1	-4	0	-7	-5
N	-2	-3	6	3	-7	-1	0	-1	1	-3	-5	0	-5	-6	-3	1	0	-6	-3	-5
D	-1	-6	3	6	-9	0	3	-1	-1	-5	-8	-2	-7	-10	-4	-1	-2	-10	-7	-5
C	-4	-5	-7	-9	9	-9	-9	-6	-5	-4	-10	-9	-9	-8	-5	-1	-5	-11	-2	-4
Q	-2	0	-1	0	-9	7	2	-4	2	-5	-3	-1	-2	-9	-1	-3	-3	-8	-8	-4
E	-1	-5	0	3	-9	2	6	-2	-2	-4	-6	-2	-4	-9	-3	-2	-3	-11	-6	-4
G	0	-6	-1	-1	-6	-4	-2	6	-6	-6	-7	-5	-6	-7	-3	0	-3	-10	-9	-3
H	-4	0	1	-1	-5	2	-2	-6	8	-6	-4	-3	-6	-4	-2	-3	-4	-5	-1	-4
I	-2	-3	-3	-5	-4	-5	-4	-6	-6	7	1	-4	1	0	-5	-4	-1	-9	-4	3
L	-4	-6	-5	-8	-10	-3	-6	-7	-4	1	6	-5	2	-1	-5	-6	-4	-4	-4	0
K	-4	2	0	-2	-9	-1	-2	-5	-3	-4	-5	6	0	-9	-4	-2	-1	-7	-7	-6
M	-3	-2	-5	-7	-9	-2	-4	-6	-6	1	2	0	10	-2	-5	-3	-2	-8	-7	0
F	-6	-7	-6	-10	-8	-9	-9	-7	-4	0	-1	-9	-2	8	-7	-4	-6	-2	4	-5
P	0	-2	-3	-4	-5	-1	-3	-3	-2	-5	-5	-4	-5	-7	7	0	-2	-9	-9	-3
S	1	-1	1	-1	-1	-3	-2	0	-3	-4	-6	-2	-3	-4	0	5	2	-3	-5	-3
T	1	-4	0	-2	-5	-3	-3	-3	-4	-1	-4	-1	-2	-6	-2	2	6	-8	-4	-1
W	-9	0	-6	-10	-11	-8	-11	-10	-5	-9	-4	-7	-8	-2	-9	-3	-8	13	-3	-10
Y	-5	-7	-3	-7	-2	-8	-6	-9	-1	-4	-4	-7	-7	4	-9	-5	-4	-3	9	-5
V	-1	-5	-5	-5	-4	-4	-4	-3	-4	3	0	-6	0	-5	-3	-3	-1	-10	-5	6

Şekil 4.2.1.3.1 – PAM70 Matrisi

4.2.1.4 Çoklu Dizi Hizalama

Çoklu dizi hizalama (ÇDH), biyobilişimde temel bir araçtır. ÇDH, dizi oluşturma, moleküler modelleme, veritabanı aramaları, filo genetik ağaç oluşturulması gibi konularda başlıca araçtır. İkili dizi hizalamadan (İDH) çok daha fazla bilgi verir. İDH'da dinamik programlama tekniğiyle polinom zamanda en iyi sonucun bulunabilmesine rağmen, var olan ÇDH teknikleri optimum hizalamayı ancak dizi sayısıyla üssel ilişkili bir sürede bulabilir. Bu da pratikte gerekli olan büyük boyutlu verilerde problemin çözümünü, günümüz hesaplama gücüyle imkânsız kılar. Ayrıca bazı sezgisel (heuristic) yöntemlerle ortaya çıkan sonuçların yerel optimuma takılma ihtimalleri çok yüksektir. Buna ek olarak, bulunan hizalamalar uzmanların bakış açılarına göre de farklı değerler alabilmektedirler. Bu yüzden yerel optimala

takılmadan farklı iyi hizalamalar verebilecek yöntemler gittikçe önem kazanmaktadır.

Çoklu dizi hizalama basit anlamda ikili evrensel hizalamanın genişletilmiş halidir, 2 diziden ziyade k sayıda dizi işin içine girmektedir. *k* sayıda diziye sahip $S=\{s_1,s_2,\ldots,s_k\}$ kümesi verildiğinde, bu tür dizilerin birbirlerine olan benzerlikleri bilindiğinde, bir çoklu hizalama M, S kümesini yeni $S'=\{s_1',s_2',\ldots,s_k'\}$, planını çıkartır, öyle ki;

i. *S'* kümesindeki tüm diziler aynı uzunluktadır ve
ii. s_i' boşlukların kaldırılmasıyla s_i elde edilmektedir, $1 \le i \le k$ için.

Bunun gibi bir hizalama için birçok olasılık mevcut olduğundan, bir sonraki problem, hizalamanın kalitesini belirleyen bir puanlama fonksiyonunu tasarlamaktır. Birbirinden farklı karakterler arasındaki uzaklığı (benzerliği) ölçen bir uzaklık (benzerlik) fonksiyonu, $\sigma(x,y)$, tanımlayacak olursak, *s'* ve *t'* arasındaki ikili hizalama puanı,

$$\sum \sigma(s_i',t_i'), \qquad 1 \le i \le L, \qquad L = |s'| = |t'|, \qquad (4.2.1.4.1)$$

en aza indirgenecektir (en yüksek dereceye çıkartılacaktır). $\sigma(x,y)$ puanı genellikle bir yer değiştirme matrisi sayesinde elde edilmektedir, tıpkı daha önceki bölümlerde tanımlanan PAM matrisi gibi.

k dizilerinin hizalanması için, çok hizalama M için çiftlerinin toplam puanı *(SP)* aynı zamanda M'in neden olduğu tüm ikili hizalamanın toplam puanı olarak tanımlanır. Bundan dolayı, çoklu hizalama M'in $\{s_1,s_2,\ldots,s_k\}$ *k* dizileri için olası en düşük puana sahip olarak hizalanmasıdır.

Son 20 yıl boyunca, çoklu dizi hizalama için kullanılabilir metodların sayısı durmadan artmıştır. Bu yöntemler üç grupta toplanabilir; modern algoritmalar, kesin algoritmalar ve yinelemeli algoritmalar.

Modern algoritmalar (Higgins et al., 1994, Löytynoja ve Milinkovitch, 2003) ikili benzerliklerin ağırlıklı toplamını en uygun şekle getirmeye çalışır. Diziler önceden hesaplanmış dendogramın gösterdiği sıraya uygun olarak teker teker çok hizalamaya eklenir. Dizi ekleme dinamik programlamadaki gibi bir ikili dizi hizalama metodu kullanarak yapılır. Bu yöntem basit ve etkindir. Ancak bir dezavantajı, hizaya bir dizi eklendiğinde, daha sonradan eklenecek dizilerle birbirlerini tutmasalar dahi, bir daha asla üzerinde değişiklik yapılamamaktadır.

Kesin algoritmalar en iyi hizaya yakınlaşmanın yerine, en iyi hizayı bulmaya çalışırlar, *k* boyutlu uzayda (*k* dizilerin sayısını belirtir) dinamik programlamanın kullanılması zaman ve alan karmaşıklığı açısından çok verimsizdir. En fazla üç dizi için kabul edilebilirdir. Konulan bu limit büyük aralıkları keşfeden bir yol bulunarak (Carillo ve Lipman, 1988) biraz daha genişletilebilir.

Yinelemeli algoritmalar, genel olarak, bir problemin çözümünün hesaplanması, önceden mevcut bir alt çözüm değiştirilerek yapılır, düşüncesi temeline dayanmaktadır. Çok hizalama da ise, bu değişiklikler dinamik programlama veya bir stokastik yol kullanılarak yapılabilir. Saklı Markov Modeli (Krogh et al., 1994), tavlama benzetimi (Kim et al. 1994), Gibbs örneklemesi (Lawrence et al. 1993) veya genetik algoritmaları (Notredame ve Higgins, 1996) kullanan birkaç yöntemler önerilmiştir. Bunların arasında, SMM yaklaşımı birçok dizi için (>100) uygulamaya kısıtlıdır. Diğer yöntemler ise uygun biçimde seçilen parametrelerle etkin sonuçlar vermektedir.

4.2.2 Benzerliğin Sade Bir Tanımı; Maksimum Biricik Eşleşme

Bir maksimum biricik eşleşme (MBE), her iki dizide de bir defa rastlanılan ve daha büyük bir alt dizinin kapsamadığı alt dizi olarak tanımlanmaktadır. Bu tanımla beraber, maksimum biricik eşleşme ile hizalamada çekirdeği oluşturan kısım olarak kabul edilir ve diziler arasındaki homoloji için önemli bir kanıt sağlar. Tanım aradaki benzerliği değerlendirirken dizi içinde her hangi bir değiştirmeye ve tekrara izin

vermez bundan dolayı hizalamadan daha katıdır. Ancak, söz konusu birbirlerine olan benzerlikleri düşük seviyedeki diziler olduğunda, herhangi bir mutasyona gösterilen tolerans, diziler arasında şans eseri eşliklere neden olur. Bu nedenle, bu kesin tanımın farklı proteinler arasındaki yerel akrabalıkları daha açık bir şekilde göstermesi beklenmektedir. MBE'nin tanımı esas olarak Delcher et al. (1999) tarafından uzun DNA sarmalların hizalanmasını hızlandırmak için ortaya konmuştur.

Bir defa tanımlandığında, üst üste gelmeyen tüm maksimum biricik eşleşmelerin uzunluğu iki protein dizisini karşılaştırmak için kullanılır; denklemdeki $S(P_x, P_i)$ puanı MBE'lerin tümünün toplam uzunluğu ile yer değiştirmiştir, $M(P_x, P_i)$, protein sayısallaştırılması için;

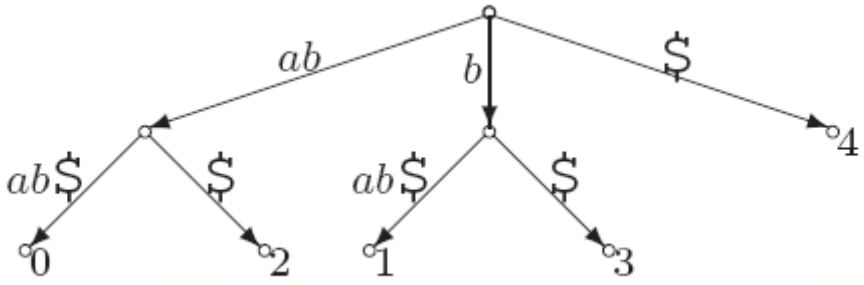
$$\varphi(P_x) = M(P_x, P_l), M(P_x, P_2), ..., M(P_x, P_n) \qquad (4.2.2.1)$$

MBE'lerin tümünün basit toplamı, iki ya da daha fazla AEÇ üst üste bindiğinde, üst üste binen kısımlar birden fazla sayılmasına neden olur. Bu problemi gidermek için, $M(P_x, P_i)$ tanımı P_x ve P_i arasındaki maksimum biricik eşleşmelerin içerdiği artık sayısı olarak değiştirilir.

MBE'leri bulmak üzere özel bir veri yapısı olan sonek ağaçları kullanılır. Verilen bir metin dizisinin tüm soneklerini saklayan sıkıştırılmış ağaç bir sonek ağacıdır. Birçok dizin işleme algoritmasında (Gusfield, 1997) uygulama alanı bulan çok güçlü ve kullanışlı bir veri yapısıdır. Şekil 4.2.2.1'de *"abab$"* dizisine ait sonek ağacı örneği gösterilmiştir.

A, n tane karakterden oluşan bir karakter dizisi olsun, $A=s_1s_2\ldots s_n$, s_n hariç, düzenli alfabe Σ'dır. Varsayım ki *$*, alfabedeki hiç bir karaktere uymayan özel bir karakter olsun. *A* dizisinin sonek ağacı T, n adet yaprağa sahip bir ağaçtır şöyle ki;

— *T* ağacında kökten yapraklara uzanan her yol *A*'nın farklı bir sonekini gösterir.
— *T* ağacının her bir kenarı boş olmayan bir *A* karakter dizinin gösterir.
— *T* ağacının her bir yaprak olmayan düğümü, kök hariç, en az iki çocuğa sahip olmalıdır.
— İki kardeş kenar tarafından gösterilen alt diziler farklı karakterlerle başlamalıdır.



Şekil 4.2.2.1 – "abab$" Sonek Ağacı

Bir diziler kümesinin $\{A_1,A_2,\ldots,A_n\}$ öneklerini göstermek için sonek ağacı tanımı kolayca genişletilebilir. Bu tip sonek ağaçlarına genelleştirilmiş sonek ağaçları denir.

İki dizi arasındaki MBE'leri bulurken ilk olarak diziler için genelleştirilmiş sonek ağaçları (Bieganski et al. 1994) kurulur. Bu basit bir şekilde yapılır, iki dizi ortalarına önemsiz bir karakter (alfabede olmayan) koyarak birleştirilir ve yeni oluşan diziden bir sonek ağacı kurulur. Biçimsel olarak anlatacak olursak, *A* ve *B* karşılaştırılacak dizilerdir, *GT* ise *{A,B}*'nin genelleştirilmiş ağacıdır, *"$a_1a_2\ldots a_n\#b_1b_2\ldots b_n\$$"* dizisinin sonek ağacı ise *Tg* olacaktır.

Bir gösterimde, maksimum biricik eşleşme, ilk kısmı yalancı değişkenden önce, diğer kısmı da sonra gelecek şekilde birleştirilen dizideki *en büyük ikilidir*. En büyük ikilileri bulma algoritması Gusfield tarafından verilmiştir (1997). Daha sonraları bu

algoritma üzerinde, her bir ikilinin farklı dizilerde bulanabileceği düşünülerek değişimler de yapılmıştır.

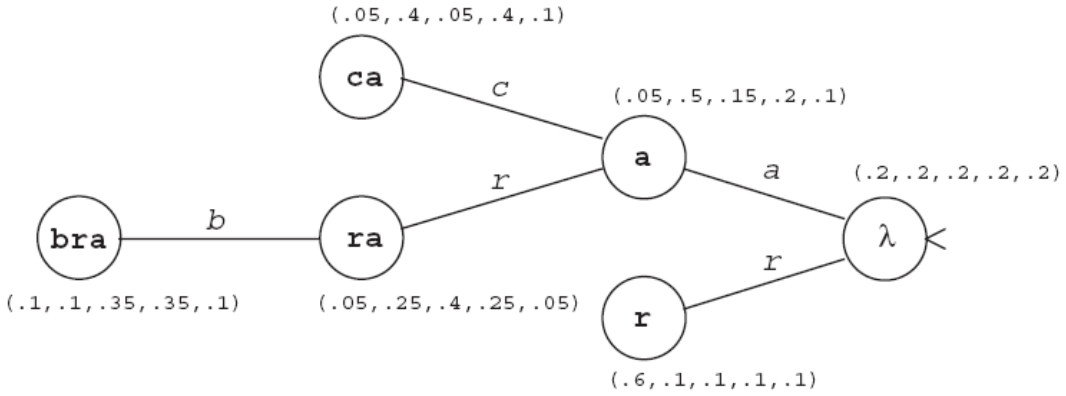
4.2.3 İkili Olasılıklı Sonek Ağaçları

İkili olasılıklı sonek ağaçları (OSA) metodu Bejerano ve Yona (2000) tarafından protein ailelerini modellemek için öne sürülmüştür. Orijinal OSA modelinin temelinde birçok girdi dizileri arasından, farklı proteinlerin içinde parçaların bağıl yerlerine aldırmadan, dikkate değer kısa parçaları bulmak yatmaktadır. Model, önceden belirlenen örneğin *L*, değerinden daha uzun olmayacak bir parçadan hemen sonra gelen sembol üzerinden bir ağırlık dağılıma neden olmaktadır. Bir diziyi belirli bir aile aittir diyerek sınıflamak için, veri kümesindeki her bir aileye ayrı bir OSA kurulmaktadır. Bununla beraber OSA üzerindeki olasılık dağılımına göre, dizinin aileye aitliğinin olasılığı sorgu dizisine atanmaktadır. Bu olasılık puanı, eşik değeri ile karşılaştırıldığında, dizinin aileye ait olup olmadığı kesin bir şekilde tespit edilmiş olmaktadır. Sınırlı bir alfabe üzerinde tanımlandığında, OSA her biri alfabenin tek bir sembolüyle etiketlenmiş kenarlara sahip olmaktadır, hiçbir sembol tek bir düğümden dallanan kenarlar ile gösterilmez. OSA içerisindeki her düğüm, sıfırdan alfabenin boyuna kadar derecelendirilir ve etiketlendirilir, etiketler ağacı düğümden köke kadar inerek oluşturulur. Düğümlerde yine alfabe ile büyüklükte olan koşullu deneysel olasılık vektörüyle beraber tutturulmuştur. Olasılık dağılım vektörü, hali hazırdaki düğümün içinde yer alan alt diziden sonra göze çarpan her sembolle aynı büyüklüktedir.

Şekil 4.2.3.1'de bir OSA örneği verilmiştir. Mesela "*bra*" düğümü ile ilişkilendirilmiş olasılık vektörü (0.1, 0.1, 0.35, 0.35, 0.1) olsun. Diğer bir deyişle, verilen OSA için, *bra* parçasından sonra *a* sembolünün gelmesi olasılığı 0.1'dir, ancak *bra* parçasından sonra *d* sembolünün gelme olasılığı 0.35'dir.

Bir sorgu dizisi S için, belirli bir OSA tarafından olasılık sembol sembol üretilir, her sembole ait olasılık ağacı gezerek, ağaçta mevcut ve o sembolden hemen önce biten en uzun alt diziyi arayarak hesaplanır. Örneğin, OSA üzerindeki *S=abracad*

için tahmini puan 0.2*0.5*0.2*0.6*0.35*0.2*0.4 şeklinde hesaplanır. Proteinler uzunlukları bakımından birbirlerine eşit olmadıklarından, olasılık uzunluklar üzerinden normalleştirilmiştir yani protein sınıflandırma amaçlarıyla her dizi her bir harfin sahip olduğu ortalama olasılık değerleriyle bildirilir.



Şekil 4.2.3.1 – {a,b,c,d,r} alfabesi üzerindeki OSA örneği

Önceki çalışmalarda önerilen iki hizalama karşılaştırmaları için farklı bir OSA metodudur (Oğul ve Mumcuoğlu, 2006). İki dizi arasındaki benzerliği göstermek için, sorgu dizisi kaynak diziden oluşturulan OSA kullanılarak öngörülür ve diziler arasındaki benzerlik puanı, öngörülen olasılık puanı olarak alınır. Dizinin olasılığı, dizideki her harfin olasılığının ortalaması ile hesaplanır:

$$\wp(P_x P_y^T) = (\gamma_{s1}(x_1) + \gamma_{s2}(x_2) + ... + \gamma_{sk}(x_k))/k \qquad (4.2.3.1)$$

$\wp(P_x P_y^T)$, P_x protein dizisinin, P_y protein dizisi üzerinde sahip olduğu olasılık puanıdır, $\gamma_i(x_i)$, P_x dizisi içinde x_i değişkenin *i.* sembolünün olma olasılığıdır, x_i sembolünden hemen önce biten, ağaçta bulunan, en uzun altdizi olan (s_i) bulmak için ağaç taranarak hesaplanmaktadır, k değeri de P_x protein dizisinin uzunluğunu vermektedir.

Yeni benzerlik ölçütünü Liao ve Noble tarafından ortaya konulan nitelik gösterim metoduna adapte etmek için, eğitim kümesindeki her dizi için ayrı bir OSA yapılmış

ve aşağıda denklemde verilen protein sayısallaştırılması üzerinden değişiklikler yapılmıştır.

$$\varphi(P_x) = \left[\wp(P_x P_1^T), \wp(P_x P_2^T), ..., \wp(P_x P_n^T)\right] \qquad (4.2.3.2)$$

Tek bir dizi için kurulan ideal bir OSA, içinde tüm dizi tarafından etiketlenmiş bir yaprak ve dizinin her öneki için bir düğüm bulundurmaktadır. Bu durumun kısa bir dizi girdisi için bile muazzam büyüklükte bellek ihtiyacı doğurması, bir OSA kurulum parametresi, diğer bir adıyla *kısa bellek uzunluğu*'nu Bejenero ve Yeno'nun uygulamasıyla şart koşmuştur. *L* ile gösterilen bu parametre, OSA düğümlerinin mertebelerini sabit bir değerle sınırlar, aynı zamanda soneklerin öngörme adımında kullanılabilmesi için gereken maksimum uzunluğa da tekabül etmektedir. OSA kurulum algoritmasındaki bir diğer parametre P_{min}, sonuçlanan OSA'da düğüm olarak gösterilebilmeleri için OSA girdisindeki dizgilerin sahip olması gereken olasılık değeridir. Bu olasılık dizginin görünüm sayısının aynı uzunluktaki tüm olası dizgilerin sayına bölümüyle hesaplanmaktadır. Kaynak dizide, dizginin sadece bir defa görünmesi bile ikili karşılaştırma için önemlidir, dolayısıyla biz ikili uygulamamızda P_{min} değerini sıfıra eşitlemekteyiz. Çoklu dizi girişi ile OSA kurulum algoritmasındaki alan karmaşıklığını en uygun şekle getirmek için Bejenaro ve Yona tarafından daha başka parametreler verilmiş olmasına karşın, ikili karşılaştırmada sadece tek-dizi girdi kullandığımız için bu parametreler daha fazla iyileştirme sağlamamaktadır. Bundan dolayı, tüm parametreler *L* değerinden daha küçük olan dizgilerin sonuçlanan OSA'da düğümler ile gösterilmesi için ayarlanmıştır.

5 – HOMOLOJİ

5.1 Homoloji

Homolog kelimesi eski Yunancadan gelmektedir ve “anlaşmak” anlamındadır, günümüzde sözlük anlamıyla “benzeşlik, benzeşim” olarak tanımlanır. Genetikte, homoloji protein ve DNA dizilimleri ile ölçülmektedir. Homolog iki gen, yüksek seviyede dizilim benzerliğine ve özdeşliğine sahiptirler, bu paylaşım ile dizilimler aynı soydan geldikleri hipotezini de desteklemektedirler. Dizilim homolojisi aynı zamanda ortak işlev göstergesi de olabilmektedir. Dizilimlerin birbirleriyle homolog olan belli bölgelerine koru adı verilmektedir. Homolog dizilimler, ortolog ve paralog olarak isimlendirilerek iki tipte sınıflandırılırlar. Evrimsel biyolojide, homoloji aynı ortak soydan gelen yapıları tanımlamak için kullanılmaktadır.

5.1.1 Evrim İçerisindeki Yapıların Homolojisi

Biyoloji biliminde paylaşılan bir soydan dolayı birbirine benzer iki ya da daha fazla yapıya homolog denilmektedir. Yapılar ortak soydan evrimleşip başka yapıları oluşturmaktadır, bu duruma evrimsel soy adı verilmektedir (yarasanın kanatlarıyla insanın kolları bu anlamda homologdur). Yapılar, gelişmelerinin ilk aşamalarında aynı dokudan farklı yapılar oluşturmaktadır, bu duruma da gelişimsel soy adı verilmektedir (insanda, dişinin yumurtalıkları ile erkeğin testisleri bu anlamda homologdur).

Homoloji, analojiden farklıdır. Örneğin, böceklerin kanatları, yarasaların kanatları ve kuşların kanatları analogdur ancak homolog değildir, yani birbirlerine benzerdirler ancak aralarında benzeşlik söz konusu değildir. Bu olağanüstü olay homoplasy olarak bilinmektedir. Bu benzer yapılar yakınsak evrim olarak bilinen bir işlem içerisinde farklı gelişimsel yollar izleyerek evrimleşmiştirler.

5.1.2 Genetikte Dizilimlerin Homolojisi

Özellikle biyoinformatikte, proteinler ve DNA arasındaki homoloji çoğunlukla dizilim benzerliği temel alınarak sonuçlandırılmıştır. Genellikle, iki gen birbiriyle yaklaşık olarak aynı DNA dizilimine sahipse, bu genlerin homolog olması olasıdır. Ancak dizilimdeki bu benzerlik bir ortak soy paylaşımından kaynaklanmayabilir; kısa dizilimler şans eseri birbirine benzer olabilmektedir ya da dizilimler özel bir proteine bağlamak üzere benzer seçilmiş olabilmektedir, tıpkı kopya etme için kalıtımsal özellikleri oluşturan genler gibi. Bu dizilimler benzerdirler ancak homolog değildirler.

Bazen, evrimsel biyoloji veya biyoinformatik alanı dışındakilerin kullandığı "yüzdesel homoloji" ifadesi yanlıştır. Biyomolekül dizilimleri arasındaki benzerliği belirtmek için "yüzdesel özdeşlik" veya "yüzdesel benzerlik" ifadeleri kullanılmaktadır. Doğal olarak meydana gelen iki dizilim için, yüzdesel özdeşlik tam bir ölçümdür, nerde ki homoloji kanıtlarla desteklenen bir hipotezdir. Tek bir durum için, aynı aile kökenini paylaştığı farz edilen dizilimlerin küçük parçacıkları karşılaştırılırken kısmı homolojiden bahsetmek mümkün olmaktadır. Protein dizilimlerini, karşılıklı homolog dizilimlerin oluşturduğu dizilim aileleri içine kümeleyecek birçok algoritma mevcuttur.

5.2 Uzak Homoloji Tespiti

İki protein aynı ortak evrimsel soydan geliyorlarsa, bu proteinlere homologdur denir. Homoloji bilgisi, iki protein arasında ortak yapı ve fonksiyon anlamına geldiğinden dolayı önemlidir. Genelde, homolojiyi ifade etmek için sadece dizi bilgisinin kullanılması hesaplamalı biyolojinin temel problemlerindendir. Eğer bir miktar yapısal ve fonksiyonel analizleri tamamlanmış benzer protein bulunabilirse, hedef protein, yapının ana belirteci dizidir varsayımını kullanarak, açıklanabilir.

Ancak, bu görüşle ilgili iki önemli problem mevcuttur. Birinci olarak, hedef protein tamamen yeni ve yapısı veritabanlarında bulunan tüm proteinlerden farklı olabilir. İkinci olarak da, aralarında eksik benzerlik bulunan iki protein yinede aynı evrimsel

ilişkilere sahip olabilirler. Yukarda verilen problemlerden ilki hesaplamalı biyoloji için dar boğazı oluşturmaktadır ve günümüzde iyi çalışan herhangi bir yöntem mevcut değildir. İkinci problem ise bilinen adıyla uzak homoloji tespitidir. Son yıllarda uzak homoloji tespiti için çok çeşitli yöntemler önerilmiştir. Birkaç başarılı çalışma dışında ya hesaplamasal olarak verimsizdirler ya da her durum için yeterli gelememektedirler.

5.3 Önceki Çalışmalar

Homoloji tespiti için kullanılan önceki yöntemler, dinamik programlama ile ifade edilen ikili hizalama benzerliğini temel almaktadır (Smith ve Waterman, 1981). Dinamik programlama yöntemleri önceden tanımlanmış amaç fonksiyonuna uygun olarak bir en uygun puanı bulmasına karşın bunu bulurken nispeten uzun dizilerden dolayı çok uzun hesaplama sürelerine maruz kalmaktadır. Hizalamayı hızlandırmak ve kabul edilebilir süreler içerisinde en yakın ya da en uygun hizalamayı bulabilmek amacıyla BLAST (Altschul et al., 1990) gibi birçok deneye dayalı yöntem geliştirilmiştir. İki protein için dizi kimliği (iki dizi hizalandıktan geriye kalanların özdeşlik yüzdesi) %40'ın üzerinde ise genel sanı bu iki proteinin homolog olduğu yönündedir. İkili dizi hizalamadaki problem değerlendirmedeki biyolojik hatalar ile sadece tek proteinin homolog olmasıdır. Sadece tek bir proteine bakarak yeni dizilmiş bir proteini açıklamak biyolojik olarak kesin olmayan sonuçlara neden olmaktadır. Bu belirsizlik dizi hizalamadaki *alacakaranlık bölgesi* olarak isimlendirilen durumdan kaynaklanır (Rost, 1999), proteinlerin evrimsel ilgilerinin tespitini yapmak için güvenlik seviyelerini sınırlandırmaktadır. Birçok hizalamada alacakaranlık bölgesi, dizi benzerliğinin %20 ila %40'larına kadar düşmektedir. İki proteinin dizi bazında benzerlik göstermemesine rağmen, örneğin dizi benzerliği %40'ın altında olduğu durumlar, yinede proteinler önemli yapısal ve fonksiyonel nitelikleri paylaşabilir. Bu durumda proteinler arasında uzak ya da mesafeli homoloji tespiti söz konusudur.

İkili dizi hizalamadaki alacakaranlık bölgesinden uzaklaşmak için aile tabanlı karşılaştırmalar önerilmiştir (Grundy, 1998). Homoloji tespiti sonuçlarının

hassaslığını iyileştirebilmek için karşılaştırmaya aileden dizileri gösteren bir küme de dâhil edilmektedir. Böylece yeni protein, belirlenmiş ailedeki tüm (ya da birçok) proteinle beraber ve aynı zamanda hizalanmaktadır. Çoklu karşılaştırmanın faydaları, ikili karşılaştırma ile kıyaslandığında homoloji tespitinin hassaslığını yaklaşık üç kat kadar artırmıştır (Park et al., 1998). Çoğu protein, uygun veritabanlarında protein aileleri dahilinde sınıflandırılmaktadır ve asıl açıklamalar bu aileler üzerinden yapılmaktadır. Bu suretle homoloji modellemede tüm aile bilgisini kullanmanın daha uygun olduğu görülmektedir.

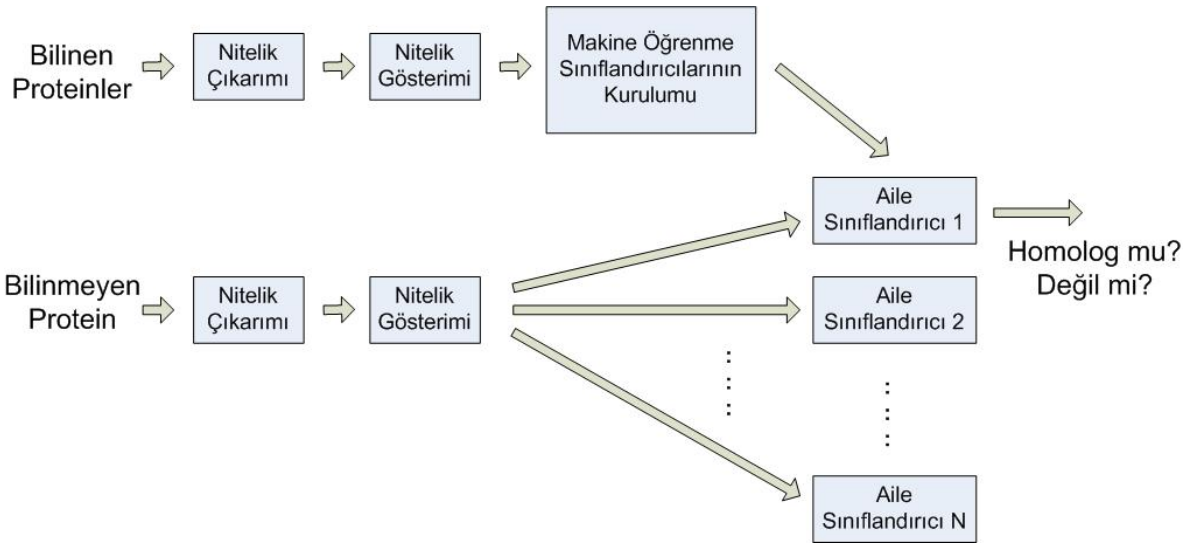
Bir ailenin tüm üyelerinin, yeni bir dizi yaratılırken aynı zamanda hizalanması bazen ailenin esas niteliklerini bozabilmektedir, buna alternatif yeni bir yaklaşım aile *profillerinin* çıkartılması ve yeni diziyi bu profile göre hizalamaktır (Gribskov et al., 1990, Eddy, 1998). Bunun için, dizilerin ortak özellikleri, ailenin profiliyle karşılaştırılarak, aileye ait olup olmadığı kontrol etmek için test edilmektedir. Bu yöntemler birden fazla homolog örnekleri üzerinden türetilmiş benzerlik istatistiğini temel almaktadır, öyle ki, tüm istatistik bilgisi birbiriyle evrimsel olarak ilgili olduğu varsayılan ya da bilinen dizilerin oluşturduğu bir kümeden üretilmiştir. Bu olasılıklı yöntemler çoğunlukla *üretken* olarak adlandırılır, çünkü protein ailesi üzerinden bir olasılık dağılımına sebep olurlar ve bilinmeyen proteinleri bu stokastik modelden ailenin yeni üyeleri olarak oluşturmaya çalışmaktadırlar.

Uygun olan veritabanlarını tekrarlı olarak homologlar için aramak ve merkez modelini her tekrarda iyileştirmek, yaklaşıma daha da kesinlik kazandırmaktadır. SAM-T98 metodu tekrarlı aile iyileştirme modellerine bir örnektir (Karplus, 1998). Bu yöntemde, diziler ilk olarak, artalan dağımlar üzerine kurulu başlangıç modeline göre hizalanmaktadır ve daha sonra model, dizileri yeni istatistiksel sonuçlarla göre birleştirilen geçerli modele göre hizalayarak, tekrarlı bir şekilde iyileştirilmektedir.

Bir başka tekrarlı yöntem PSI-BLAST, BLAST üzerine yerleşmektedir. BLAST arama ve sonuçları iyileştirmeyi tekrarlı olarak gerçekleştirmektedir (Altschul et al., 1997). Tümleşik dizi/yapı hizalamaları homologları aramak için başka bir yöntem de tekrarlı olarak yapılmaktadır (Walqvist et al., 2000).

Üretken yaklaşımlardaki (generative methods) başlıca problem, yanlış pozitiflerin çok fazla üretilmesidir, yani homolog olmamalarına karşın çok sayıda homolog çift bulmaktadırlar. Bu sebepten dolayı, son zamanlarda uzak homoloji tespiti üzerine yapılan çalışmalarda homolog (pozitif) ve homolog olmayan (negatif) sınıflar arasında belirleyiciliği sağlamak için fark gözeten (discriminative) iskelet (framework) kullanılmaya başlanmıştır. Üretken yöntemlerinin tersine, fark gözeten yöntemler, sınıfları farklılaştıran nitelik birleşimlerini öğrenmeye odaklanmaktadır. Bu yöntemler pozitif ve negatif örnekleri ayırt edici modeli kurmaya çalışmaktadırlar. Diğer bir deyişle homolog olmayanlarda hesaba katılmaktadırlar.

Fark gözeten homoloji tespiti modellerinde, başlıca iki aşama vardır: eğitim ve test. Eğitim aşamasında belirlenen aile için bir makine öğrenen sınıflandırıcı (machine learning classifier) kurulmaktadır, test aşamasında ise kurulan bu sınıflandırıcı test edilen proteinin aileyi ait olup olmadığını kararlaştırmak için kullanılmaktadır. Genelde, veritabanındaki her bir aile için bir sınıflandırıcı kurulmaktadır ve proteinlerin, bilinen protein sınıflarından birine ait olup olmadığı kontrol edilir. Her iki aşama da protein dizilimlerinden bir takım bilgilendirici niteliklerin çıkarımına ve bu çıkarımların uygun bir şekilde gösterimlerine ihtiyaç duymaktadır. Şekil 5.3.1'de fark gözeten homoloji tespiti yaklaşımına genel bir bakış açısı vermektedir.



Şekil 5.3.1 – Fark Gözeten Homoloji Tespiti Modeli

Fark gözeten yaklaşımı kullanan geçerli yöntemler, nitelik çıkarım yöntemleri kısmında birbirinden ayrılmaktadırlar, bu ayrım nitelik gösterimlerinde ve kullanılan sınıflandırıcının tipinde yapılmaktadır. Acemi Bayes, K-en Yakın Komşu ve Destek Vektör Makineleri arasından, sonuncusu protein sınıflandırma ile alakalı birçok uygulamada diğerlerine göre daha başarılı bir performans ortaya koymaktadır (Liao ve Noble, 2003; Ding ve Dubchak, 2001).

Doğru pozitifleri (doğru tahmin edilen homologlar) ve yanlış pozitifleri (homolog olarak tahmin edilen gerçekte homolog olmayanlar) birbirinden ayırmada doğru sonuçlar elde etme bakımından fark gözeten yöntemler, üretken yöntemlere kıyasla daha başarılıdır. Ancak, eğitim ve test aşamaları geleneksel işistasyonlarıyla kullanılırken, her bir aşamanın tamamlanması çok zaman almaktadır, bu durum pratik olarak kullanılma uygunluğunu ortadan kaldırmaktadır. Sınıflandırma kesinliğini muhafaza eden daha verimli yöntemlere ihtiyaç duyulmaktadır.

İlk fark gözeten yaklaşımı (DVM-Fisher), her bir proteini Fisher puanı adıyla verilen, protein ailesi için kurulan Saklı Markov Modeli profilinden çıkarılan vektörler ile göstermektedir, Destek Vektör Makineleri proteinleri sınıflandırırken bu vektörlerden faydalanır (Jaakola et al., 2000). Yeni ve daha başarılı bir çalışma olan DVM-Pairwise metodu (Liao ve Noble, 2003), dizi benzerliğini, pozitif ve negatif örnekleri birbirinden ayırt etmek için DVMler ile birleştirmektedir. İkili benzerlik puanı tabanlı nitelik gösterimi hakkında detaylı açıklama Bölüm 3'de verilmiştir. DVM-Pairwise için hem eğitim, hem de test kümeleri, pozitif ve negatif örnekler içermektedir. Bu yöntem daha önceden dinamik programlama tabanlı hizalama ve BLAST puanları için test edilmiştir. Kesinlik açısından DVM-Pairwise yaklaşımı, tüm yaklaşımlar arasındaki en iyi yöntemdir ancak yöntem hesaplamalarda oldukça verimsizdir, başka bir deyişle uzun dizileri hizalamak çok uzun sürmektedir. Bu yaklaşımın başka bir dezavantajı, hizalama yapılırken evrimsel olarak birbiriyle alakalı olmayan eşleşmelerde meydana çıkmaktadır.

5.4 Sistemler ve Yöntemler

Bu çalışmada, bir n-peptit birleşimine ait bütün özellik girdileri sadece bir sınıflandırıcıya toplu olarak verilmek yerine, özellik girdileri belli anlamlı gruplara ayrılıp, farklı sınıflandırıcılara verilerek bir üst sınıflandırma yaklaşımı denenmiştir. Bu yaklaşımda farklı sınıflandırıcıların çıktı değerlerini değerlendirmek üzere aşağıdaki yaklaşımlar kullanılmıştır:

[1] Ortalama alma (ORT): Farklı DVM sınıflandırıcılarının çıktı değerlerinin ortalaması alınarak bu değer tek bir DVM çıktısı olarak kabul edilmiş ve bu değere göre sınıflandırma işlemi gerçekleştirilmiştir.

$$D = \frac{\left(dSVM_1 + dSVM_2 + ... + dSVM_k\right)}{k} \tag{5.4.1}$$

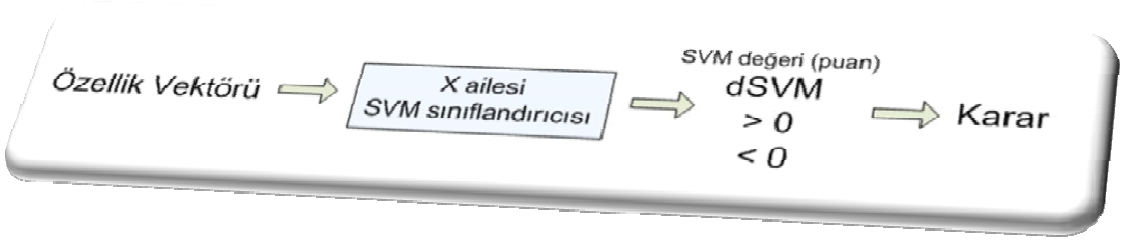
[2] Ağırlıklı ortalama alma (AORT): Her bir DVM sınıflandırıcısına bir ağırlık değeri verilerek bunların çıktı değerlerinin ağırlık ortalaması alınmış ve bu değer tek bir DVM çıktısı olarak kabul edilmiştir ve bu değere göre sınıflandırma işlemi gerçekleştirilmiştir. Bu ağırlık değerleri farklı sınıflandırıcıların eğitim kümesindeki başarı oranlarına göre verilmiştir. Böylece eğitim kümesinde en başarılı olan sınıflandırıcının her hangi bir test örneğinin sınıflandırılmasında diğerlerine göre daha etkili olması sağlanmıştır.

$$D = \frac{\left(w_1 * dSVM_1 + w_2 * dSVM_2 + ... + w_k * dSVM_k\right)}{\left(w_1 + w_2 + ... + w_k\right)} \tag{5.4.2}$$

[3] Öğrenme kümesinde en başarılı olanı seçme (MAX): Her bir DVM sınıflandırıcısını kullanıp, çıktı değerlerini kullanmak yerine sadece o aile için eğitim kümesinde en başarılı olan sınıflandırıcı test edilmiş ve onun sonucu doğru kabul edilerek sınıflandırma işlemi gerçekleştirilmiştir.

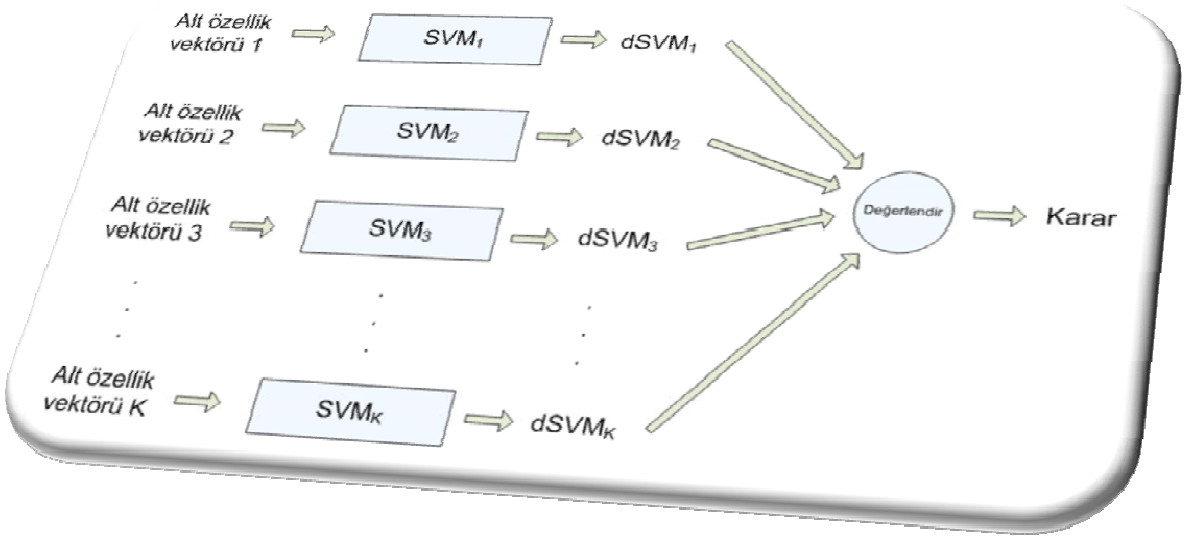
$$D = dSVM_x \text{, } \textit{(x eğitim kümesinde en başarılı olan sınıflandırıcıdır.)} \quad (5.4.3)$$

Kullanılan modeller Şekil 5.4.1 ve Şekil 5.4.2 verilmiştir. Şekil 5.4.1'de gösterilen modelde tek bir sınıflandırıcı mevcuttur.



Şekil 5.4.1 – X Ailesi için Sınıflandırıcı Modeli

Şekil 5.4.2'de ise K sayıda sınıflandırıcı, diğer bir deyişle öngörülen ihtiyaç adedinde sınıflandırıcı kullanılabilmektedir.



Şekil 5.4.2 – Önerilen Sınıflandırıcı Modeli

5.4.1 DVM ile İkili Sınıflandırma

Pozitif ve negatif örnekleri birbirinden ayırt etmek için Destek Vektör Makineleri kullanılmaktadır. Destek Vektör Makinelerini eğitmek için kullanıma açık kaynak koduna sahip DVM-Gist yazılımı (bu yazılım www.cs.columbia.edu/compbio/svm adresinde mevcuttur) kullanılmıştır. DVM-Gist yazılımında, çekirdek fonksiyonu girdi vektörünün ikilileri arasında bir benzerlik puanı olarak rol oynamaktadır. Temel çekirdek, her bir vektör nitelik uzayında 1 uzunluk birimine eşdeğer olacak şekilde normalleştirilmektedir. Bu aşağıda yapılan anlatım ile gerçekleştirilmektedir,

$$K(X,Y) = \frac{X,Y}{\sqrt{(X,X)(Y,Y)}} \qquad (5.4.1.1)$$

X ve Y girdi vektörleridir, K(.,.) çekirdek fonksiyonudur ve "." nokta çarpımını göstermektedir. Çekirdek daha sonra aşağıda ki gibi radyan tabanlı çekirdeğe K'(X,Y) dönüştürülür:

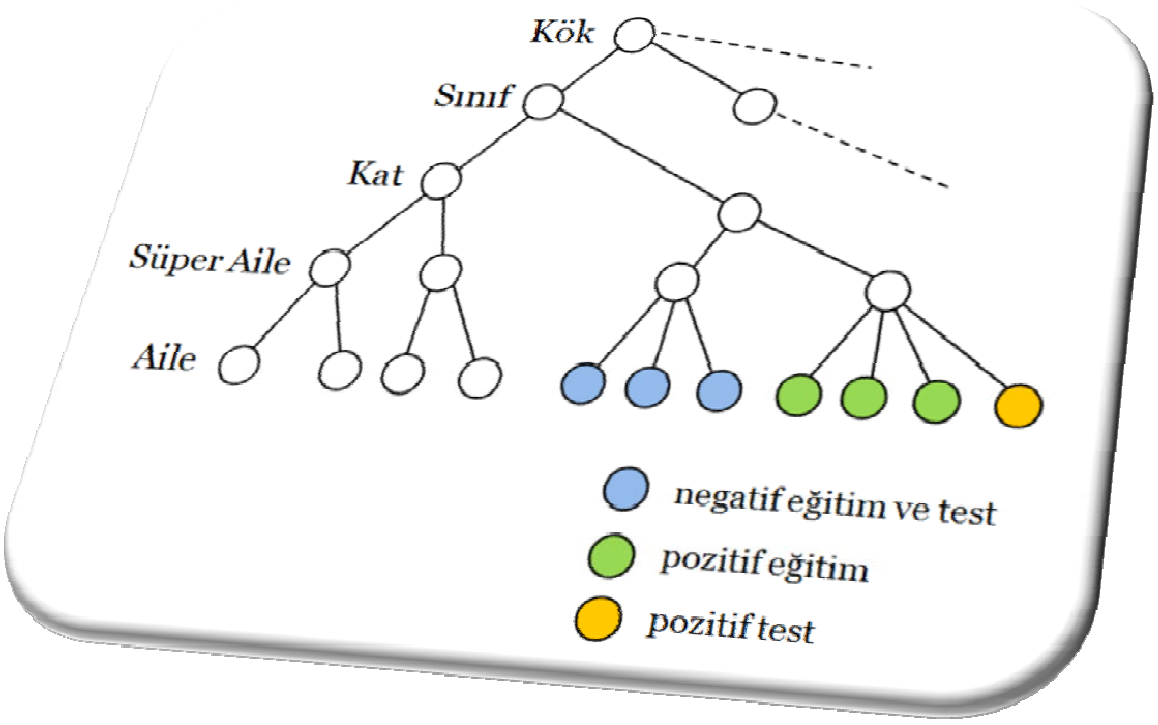
$$K'(X,Y) = e^{-\frac{K(X,X)-2K(X,Y)+k(Y,Y)}{2\sigma^2}} + 1 \qquad (5.4.1.2)$$

Burada σ ile gösterilen genişlik, pozitif eğitim örneğinden en yakın negatif örneğe olan kenarortay öklit uzaklığıdır. DVM'yi ortadan ayıran şeridin orijin noktasından geçmesi gerektiğinden, 1 sabit değeri çekirdeğe eklenmektedir, böylece veri orijin noktasından uzaklaşmaktadır. Çekirdek matrisin diyagonaline 0.02*ρ, değeri eklenir, böylece asimetrik sınır tamamlanır, ρ değeri eğitim kümesi proteinlerinin, geçerli proteinle aynı etikete sahip olanların verdiği kesirdir. Önceki DVM sınıflandırma yöntemlerinde de kullanım aynı şekilde yapılmaktadır (DVM-Pairwise, DVM-BLAST, DVM-Fisher). Sonuç DVM çıktısı, test kümesindeki her proteine karşılık gelen ayırt edici puanların bir listesidir.

5.4.2 Tanımlamalar ve Deneysel Düzenek

N-peptit birleşimleri, Java Programlama dili ve Perl programlama dilinin standart kütüphane fonksiyonları kullanılarak yapılmıştır. Tüm programlar *bash* iş parçacığının denetimindeki Linux sistemde çalıştırılmıştır. Testler, Pentium III 800 MHz çift işlemcili ve 2.00 GB RAM'e sahip bir sunucu üzerinde gerçekleştirilmiştir.

Yöntemler, SCOP protein aile veritabanın bir altkümesinde, proteinlerin ailelere sınıflandırma kabiliyetini anlayabilmek için test edilmiştir (Murzin et al., 1995). Uzak homoloji, aynı ailenin bireylerini eğitim kümesinin dışında ve aynı süperaileden gelen proteinleri pozitif kümede bırakarak taklit edilmiştir. Tarafsız bir karşılaştırma yapabilmek için DVM-Pairwise ve DVM-BLAST tarafından da kullanılmış aynı deneysel düzenek üzerinde çalışılmıştır. Çalışmalar sonucu ortaya çıkan sonuçlar, kesinlik ve verim bakımından değerlendirilmiştir.



Şekil 5.4.2.1 – Uzak Homoloji Tespitinin SCOP Veritabanı Hiyerarşisi Üzerindeki Bir Simülasyonu

Denemeler, sahip olduğu protein çiftleri içinde ikili benzerliği 10-25 BLAST E-değerinden (REF: Basic Alignment Search Tool) yüksek olmayan, SCOP1.53 veritabanının bir alt kümesi üzerinde yapılmıştır. Eğitim ve test kümeleri Liao ve Noble'ın çalışmasında olduğu gibi sonuç olarak 51 aile için test yapılacak şekilde ayrılmıştır. Oluşturulan her aile için, aile içindeki proteikler pozitif test örnekleri olarak, aile dışındaki ancak aynı süperaile içerisindeki proteinler, pozitif eğitim örnekleri olarak, süperaile dışındaki proteinler de negatif eğitim ve test örnekleri olarak kabul edilmiştir. Her bir aile için en az 10 pozitif eğitim örneği seçilmiştir. Negatif örnekler ise pozitif örnekler ile aynı oranda olacak şekilde rasgele test ve eğitim kümelerine ayrılmıştır. Ailelere ait eğitim ve test kümeleri, kümelerdeki pozitif ve negatif örnekleri gösteren Çizelge 5.4.2.1 aşağıda verilmiştir.

Çizelge 5.4.2.1 – SCOP Veri Kümesindeki Örnek Sayıları

	Pozitif Küme		**Negatif Küme**	
Aile Adı	**Eğitim**	**Test**	**Eğitim**	**Test**
1.27.1.2	10	8	2408	1926
1.36.1.2	29	7	3477	839
1.36.1.5	10	26	1199	3117
1.4.1.1	26	23	2256	1994
1.4.1.2	41	8	3557	693
1.4.1.3	40	9	3470	780
1.41.1.2	36	6	3692	615
1.41.1.5	17	25	1744	2563
1.45.1.2	33	6	3650	663
2.1.1.1	90	31	3102	1068
2.1.1.2	99	22	3412	758
2.1.1.4	88	33	3033	1137
2.1.1.5	94	27	3240	930
2.28.1.1	18	44	1246	3044
2.28.1.3	56	6	3875	415
2.38.4.1	30	5	3682	613

Çizelge 5.4.2.1 devam ediyor.

Aile Adı	Pozitif Küme		Negatif Küme	
	Eğitim	Test	Eğitim	Test
2.38.4.3	24	11	2946	1349
2.38.4.5	26	9	3191	1104
2.44.1.2	11	140	307	3894
2.5.1.1	13	11	2345	1983
2.5.1.3	14	10	2525	1803
2.52.1.2	12	5	3060	1275
2.56.1.2	11	8	2509	1824
2.9.1.2	17	14	2370	1951
2.9.1.3	26	5	3625	696
2.9.1.4	21	10	2928	1393
3.1.8.1	19	8	3002	1263
3.1.8.3	17	10	2686	1579
3.2.1.2	37	16	3002	1297
3.2.1.3	44	9	3569	730
3.2.1.4	46	7	3732	567
3.2.1.5	46	7	3732	567
3.2.1.6	48	5	3894	405
3.2.1.7	48	5	3894	405
3.3.1.2	22	7	3280	1043
3.3.1.5	13	16	1938	2385
3.32.1.1	42	9	3542	759
3.32.1.11	46	5	3880	421
3.32.1.13	43	8	3627	674
3.32.1.8	40	11	3374	927
3.42.1.1	29	10	3208	1105
3.42.1.5	26	13	2876	1437
3.42.1.8	34	5	3761	552
7.3.5.2	12	9	2330	1746

Çizelge 5.4.2.1 devam ediyor.

Aile Adı	Pozitif Küme		Negatif Küme	
	Eğitim	Test	Eğitim	Test
7.3.6.1	33	9	3203	873
7.3.6.2	16	26	1553	2523
7.3.6.4	37	5	3591	485
7.39.1.2	20	7	3204	1121
7.39.1.3	13	14	2083	2242
7.41.5.1	10	9	2241	2016
7.41.5.2	10	9	2241	2016

Aile isimlerin yer aldığı Çizelge 5.4.2.2'de yine aşağıda verilmiştir. Çizelge üzerindeki ilk sütun ailenin kimlik bilgisidir, ikinci sütun ise ailenin sahip olduğu biyolojik isimlerdir.

Çizelge 5.4.2.2 – Deneylerde kullanılan ailelerin isimleri

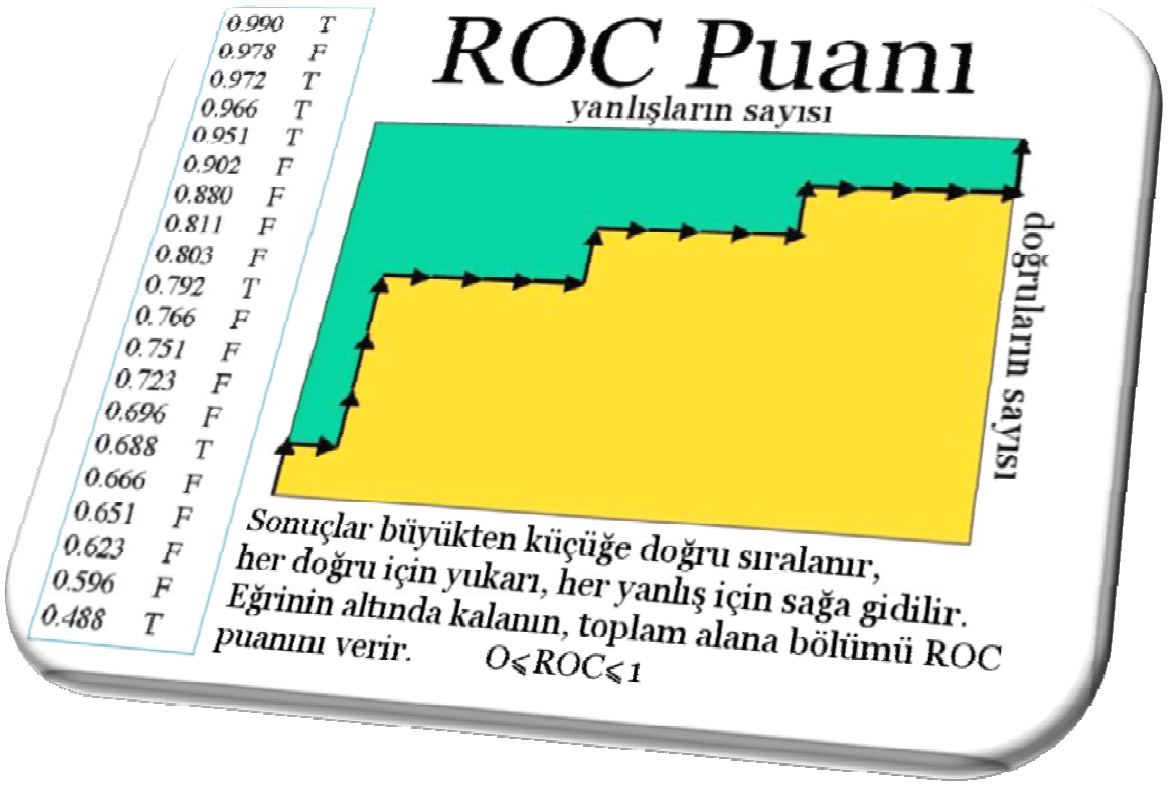
Kimlik	Aile Adı
1.27.1.2	Short-chain cytokines
1.36.1.2	Phage repressors
1.36.1.5	Bacterial repressors
1.4.1.1	Homeodomain
1.4.1.2	Recombinase DNA-binding domain
1.4.1.3	Myb
1.41.1.2	S100 proteins
1.41.1.5	Calmodulin-like
1.45.1.2	Bacterial repressors
2.1.1.1	V set domains (antibody variable domain-like)
2.1.1.2	C1 set domains (antibody constant domain-like)
2.1.1.4	I set domains

Çizelge 5.4.2.2 devam ediyor.

Kimlik	Aile Adı
2.1.1.5	E set domains
2.28.1.1	Legume lectins
2.28.1.3	Galectin (animal S-lectin)
2.38.4.1	Anticodon-binding domain
2.38.4.3	Single strand DNA-binding domain. SSB
2.38.4.5	Cold shock DNA-binding domain-like
2.44.1.2	Eukaryotic proteases
2.5.1.1	Plastocyanin/azurin-like
2.5.1.3	Multidomain cupredoxins
2.52.1.2	Phosphotyrosine-binding domain (PTB)
2.56.1.2	Fatty acid binding protein-like
2.9.1.2	Plant virus proteins
2.9.1.3	Insect virus proteins
3.1.8.1	alpha-Amylases. N-terminal domain
3.1.8.3	beta-glycanases
3.2.1.2	Tyrosine-dependent oxidoreductases
3.2.1.3	Glyceraldehyde-3-phosphate dehydrogenase-like. N-terminal domain
3.2.1.4	Formate/glycerate dehydrogenases. NAD-domain
3.2.1.5	Lactate & malate dehydrogenases. N-terminal domain
3.2.1.6	6-phosphogluconate dehydrogenase-like. N-terminal domain
3.2.1.7	Amino-acid dehydrogenase-like. C-terminal domain
3.3.1.2	FAD-linked reductases. N-terminal domain
3.3.1.5	FAD/NAD-linked reductases. N-terminal and central domains
3.32.1.1	Nucleotide and nucleoside kinases
3.32.1.11	RecA protein-like (ATPase-domain)
3.32.1.13	Extended AAA-ATPase domain
3.32.1.8	G proteins
3.42.1.1	Thioltransferase
3.42.1.5	Glutathione S-transferases. N-terminal domain
3.42.1.8	Glutathione peroxidase-like

Çizelge 5.4.2.2 devam ediyor.

Kimlik	Aile Adı
7.3.5.2	Spider toxins
7.3.6.1	Long-chain scorpion toxins
7.3.6.2	Short-chain scorpion toxins
7.3.6.4	Plant defensins
7.39.1.2	Nuclear receptor
7.39.1.3	LIM domain
7.41.5.1	Rubredoxin
7.41.5.2	Desulforedoxin



Şekil 5.4.2.2 – ROC Puan Hesaplaması

Her sınıflandırma işleminde yapıldığı gibi, protein ailelerini sınıflandırma yöntemleri de kesinlik (yanlış pozitifleri eleme kabiliyeti) ve hassasiyet (doğru pozitifleri tespit etme kabiliyeti) arasındaki dengeyi kurmakla uğraşmaktadır. Veri kümesindeki pozitif ve negatif örnekler, eşit bir şekilde dağılım göstermediğinden dolayı, doğrudan kesinlik ve hassasiyet ölçütlerinden önce, ROC kısaltması ile Receiver Operating Characteristics (Gribskov ve Robinso, 1996) adı verilen eğri, kesinlik ve hasiyet arasındaki dengeyi değerlendirmek için kullanılmıştır. ROC eğrisi altında kalan alan ROC puanı olarak tanımlanabilir. ROC eğrisi değişen sınıflandırma eşik değerlerine göre doğru pozitiflerin sayısının, yanlış pozitiflerin bir fonksiyonu olarak çizilmesiyle oluşmaktadır. ROC puanı 1 (bir) olduğunda anlamı, pozitifler mükemmel bir şekilde negatiflerden ayrılmıştır, olmaktadır. ROC puanı 0 (sıfır) olduğunda ise herhangi bir pozitif bulunamadı anlamına gelir. ROC puanının nasıl hesaplandığı Şekil 5.4.2.2'de gösterilmiştir.

6 – SONUÇLAR ve TARTIŞMA

6.1. Değerlendirmeve Tartışma

SCOP ailesine ait proteinlerin sınıflandırma testleri bitirilmiş ve testler sonucunda uygulanan farklı yöntemlerle ailelere ait ROC puanları birer birer hesaplanmıştır. Çizelge 6.1 ve 6.2'de tüm yöntemler için ailelere ait DVM performansları verilmiştir.

Çizelgelerde aile başlığı ile verilen sütun deneyde kullanılan SCOP veri kümesine ait aileleri belirten kimliklerdir. Proteinler üzerinde yapılan karşılaştırmalar proteinlerin aile isimleri kullanılmadan, kısaca sahip oldukları kimlikler ile yapılmıştır. Çizelge 6.1 üzerinde sırasıyla her sütun başlığı ve sütun içeriği kısaca anlatılmıştır.

Sınıflandırıcı puanlarını, sınıflandırıcılardan çıktı olarak sağlıklı bir biçimde elde edebilmek için sınıflandırıcıya girdi olarak verilecek tüm nitelik vektörlerinin, aynı boyutta olması şarttır. Çizelge 6.1'de DVM performansı 0,7555 olan 1.36.1.2 ailesi içindeki üç proteinin dizilimleri Şekil 6.1'de gösterilmiştir. Şekil 6.1'de görüldüğü üzere aileler içerisindeki proteinlerin birçoğu farklı dizilimlere ve uzunluklara sahiptirler. Sınıflandırıcılara gönderilmek için, bu dizilimlerden, dizilim içindeki n-peptit kompozisyonlarına bakılarak nitelik vektörleri çıkarılır.

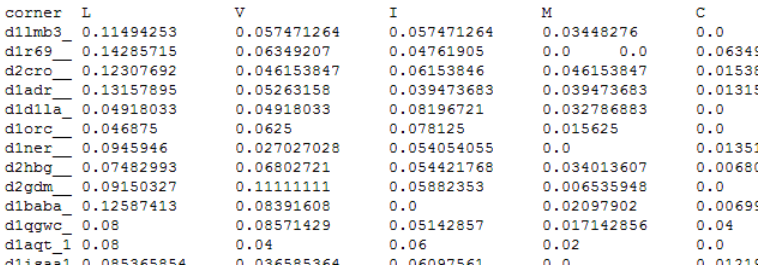
aile 1.36.1.2 için örnek protein dizilimleri

protein adı	dizilim
d1lmb3_	PLTQEQLEDARRLKAIYEKKKNELGLSQESVADKMGMGQSGVGALFNGINALNAYNAALLAKILKVSVEEFSPSIAREIYEMYEAVS
d1r69__	SISSRVKSKRIQLGLNQAELAQKVGTTQQSIEQLENGKTKRPRFLPELASALGVSVDWLLNGT
d2cro__	MQTLSERLKKRRIALKMTQTELATKAGVKQQSIQLIEAGVTKRPRFLFEIAMALNCDPVWLQYGT

Şekil 6.1 – 1.36.1.2 Ailesinden Örnek Protein Dizilimleri

Çizelge 6.1'de ikinci sütunda n=1 ile ifade edilen sütun ailelere ait protein dizilimlerinin aminoasit birleştirme metodu (n=1) ile gösterildiğinde ortaya çıkan DVM performanslarıdır. Bu sütunda yapılan hesaplamalarda tüm proteinler yirmi

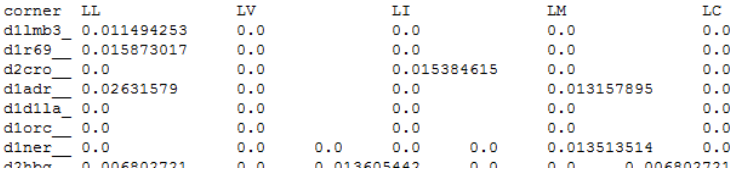
boyutlu nitelik vektörleri ile gösterilmiştir. Nitelik vektörlerinin boyutlarının 20 olmasının nedeni aminoasitler arasındaki bağ sayılarına bakılmasıdır, başka bir deyişle aminoasit alfabesinin 20 elemandan oluşmasıdır. Proteinler sınıflandırıcılara bu şekilde gönderilmektedir. Şekil 6.2'de ilk sütun aile içinde protein kimlikleri verilir, satırlarda da ilk sütunda kimliği verilmiş protein için sınıflandırıcıya gönderilmek üzere çıkarılan nitelik vektörlerini göstermektedir.



corner	L	V	I	M	C
d1lmb3_	0.11494253	0.057471264	0.057471264	0.03448276	0.0
d1r69__	0.14285715	0.06349207	0.04761905	0.0 0.0	0.0634
d2cro__	0.12307692	0.046153847	0.06153846	0.046153847	0.0153
d1adr__	0.13157895	0.05263158	0.039473683	0.039473683	0.0131
d1d1la_	0.04918033	0.04918033	0.08196721	0.032786883	0.0
d1orc__	0.046875	0.0625	0.078125	0.015625	0.0
d1ner__	0.0945946	0.027027028	0.054054055	0.0	0.0135
d2hbg__	0.07482993	0.06802721	0.054421768	0.034013607	0.0068
d2gdm__	0.09150327	0.11111111	0.05882353	0.006535948	0.0
d1baba_	0.12587413	0.08391608	0.0	0.02097902	0.0069
d1qgwc_	0.08	0.08571429	0.05142857	0.017142856	0.04
d1aqt_1	0.08	0.04	0.06	0.02	0.0
d1jgaa1	0.085365854	0.036585364	0.06097561	0.0	0.0121

Şekil 6.2 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 20 Boyutlu

Çizelge 6.1'deki sütun üçte (n=2) proteinler içinde dizilimlere bakılırken dipeptit bağları göz önünde bulundurulur. Protein dizilimi içerisinde birbirine bağlı her aminoasit çifti, aynı zamanda bir dipeptit kompozisyonunun bir elemanıdır. Aminoasit alfabesi yirmi elemandan oluştuğundan dolayı ikili bağlarla kurulabilecek toplam dört yüz adet aminoasit çifti oluşabilir. Bu nedenle her protein 400 boyutlu bir nitelik vektörü ile gösterilmektedir. Sınıflandırıcılara gönderilen bu bilgiler, yine sınıflandırıcılarda işlenir, çıkarılan sonuçlar üzerinden de protein homolojisini veren ROC puanları elde edilir. Şekil 6.3'de ilk sütunda aile içindeki protein kimlikleri verilir, satırlarda da ilk sütunda kimliği verilmiş protein için sınıflandırıcıya gönderilmek üzere çıkarılan nitelik vektörlerini göstermektedir.



corner	LL	LV	LI	LM	LC
d1lmb3_	0.011494253	0.0	0.0	0.0	0.0
d1r69__	0.015873017	0.0	0.0	0.0	0.0
d2cro__	0.0	0.0	0.015384615	0.0	0.0
d1adr__	0.02631579	0.0	0.0	0.013157895	0.0
d1d1la_	0.0	0.0	0.0	0.0	0.0
d1orc__	0.0	0.0	0.0	0.0	0.0
d1ner__	0.0	0.0 0.0	0.0 0.0	0.013513514	0.0
d2hbg__	0.006802721	0.0	0.013605442	0.0	0.006802721

Şekil 6.3 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 400 Boyutlu

Çizelge 6.1'de dördüncü sütundaki (n=3) ailelere ait proteinler içerisindeki aminoasitler tripeptit bağları ile bağlanmaktadır. Proteinin dizilimi içerisindeki her üçlü aminoasit grubu bir tripeptit üçlüsüdür. Proteinin diziliminde bu üçlülerden birbirinden farklı olarak sekiz bin adet birleşim oluşabilmektedir. Kısacası, sütun içindeki ailelere ait her bir protein 8000 boyutlu nitelik vektörleri ile gösterilmektedir. Bu nitelik vektörlerinin sınıflandırıcıya gönderilmesi sonucunda çıktı olarak her proteine özel sınıflandırıcı puanları hesaplanmaktadır. Protein homolojisi son olarak hesaplanan bu ROC puanlarıyla tespit edilmektedir. Son adımda da DVM performansları hesaplanmaktadır. Şekil 6.4'de ilk sütunda aile içindeki protein kimlikleri verilir, satırlarda da ilk sütunda kimliği verilmiş protein için sınıflandırıcıya gönderilmek üzere çıkarılan nitelik vektörlerini göstermektedir.

corner	LLL	LLV	LLI	LLM	LLC	LLA
d1lmb3_	0.0	0.0	0.0	0.0	0.0	0.011494253
d1r69__	0.0	0.0	0.0	0.0	0.0	0.0
d2cro__	0.0	0.0	0.0	0.0	0.0	0.0
d1adr__	0.0	0.0	0.0	0.0	0.0	0.013157895
d1d1la_	0.0	0.0	0.0	0.0	0.0	0.0
d1orc__	0.0	0.0	0.0	0.0	0.0	0.0
d1ner__	0.0	0.0	0.0	0.0	0.0	0.0
d2hbg__	0.0	0.0	0.0	0.0	0.0	0.0
d2gdm__	0.0	0.0	0.0	0.0	0.0	0.0

Şekil 6.4 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 8000 Boyutlu

corner	L		LL		LLL	LLV	LLI
d1lmb3_	0.11494253		0.011494253		0.0	0.0	0.0
d1r69__	0.14285715		0.015873017		0.0	0.0	0.0
d2cro__	0.12307692		0.0		0.0	0.0	0.0
d1adr__	0.13157895	.....	0.02631579	.....	0.0	0.0	0.0
d1d1la_	0.04918033		0.0		0.0	0.0	0.0
d1orc__	0.046875		0.0		0.0	0.0	0.0
d1ner__	0.0945946		0.0		0.0	0.0	0.0

Şekil 6.5 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 8420 Boyutlu

Çizelge 6.1'de beşinci sütunda (n=1-3) protein dizilimlerinden çıkarılan önceki nitelik vektörleri birbiriyle birleştirilmiş ve sonuç olarak ortaya çıkan nitelik vektörünün boyutu ilk üç sütunda çıkan nitelik vektörlerinin boyutları toplamına eşit olmuştur. Nitelik vektörü içerisinde verilen birleşimler hem tripeptit bağlarıyla, hem dipeptit bağlarıyla hem de peptit bağlarıyla oluşan gruplardır. Sırasıyla yirmi, dört

yüz ve sekiz bin boyutlu vektörlerin birleşiminde yeni nitelik vektörü boyutu sekiz bin dört yüz yirmi olmaktadır. Kısacası bir protein için sınıflandırıcıya gönderilen nitelik vektörünün boyutu bu rakama eşittir. Şekil 6.5'de proteinlerin sınıflandırıcıya gönderilmek üzere çıkarılan nitelik vektörleri gösterilmektedir.

Çizelge 6.1 – Aileler için ROC Sonuçları (n-peptit birleşimler)

Aile	N=1	N=2	N=3	N=1–3	**ORT**	**AORT**	**MAX**
1.27.1.2	0,9311	0,9730	0,8523	0,9690	0,9740	0,9740	0,9730
1.36.1.2	0,7555	0,8369	0,5433	0,8677	0,7970	0,7990	0,8369
1.36.1.5	0,6603	0,9344	0,4670	0,8734	0,8390	0,8670	0,9344
1.4.1.1	0,9902	0,9702	0,8180	0,9931	0,9930	0,9940	0,9902
1.4.1.2	0,9165	0,9066	0,9134	0,9493	0,9410	0,9410	0,9165
1.4.1.3	0,9690	0,8383	0,8694	0,9741	0,9410	0,9460	0,9690
1.41.1.2	0,7813	0,9894	0,9594	0,9764	0,9410	0,9500	0,9894
1.41.1.5	0,9242	0,9655	0,9627	0,9881	0,9750	0,9750	0,9655
1.45.1.2	0,8690	0,8969	0,4447	0,9515	0,9040	0,9020	0,8969
2.1.1.1	0,7003	0,8329	0,8877	0,8635	0,8460	0,8560	0,8877
2.1.1.2	0,9044	0,9430	0,9730	0,9676	0,9710	0,9740	0,9730
2.1.1.4	0,7603	0,8349	0,8668	0,8690	0,8550	0,8580	0,8668
2.1.1.5	0,6852	0,6404	0,8321	0,7110	0,7240	0,7350	0,8321
2.28.1.1	0,4958	0,6703	0,3350	0,6847	0,5600	0,5930	0,6703
2.28.1.3	0,8317	0,5631	0,7426	0,7558	0,7880	0,8030	0,8317
2.38.4.1	0,8101	0,5380	0,4963	0,5811	0,6620	0,7250	0,8101
2.38.4.3	0,8234	0,7689	0,5325	0,8203	0,8420	0,8530	0,8234
2.38.4.5	0,8438	0,8306	0,6661	0,8417	0,9040	0,9080	0,8438
2.44.1.2	0,2958	0,3943	0,2957	0,2314	0,2840	0,2870	0,3943
2.5.1.1	0,8694	0,9010	0,6974	0,9212	0,9060	0,9050	0,9010
2.5.1.3	0,7447	0,7546	0,7285	0,7967	0,7740	0,7330	0,7546
2.52.1.2	0,5636	0,7184	0,8519	0,8218	0,6430	0,6620	0,8519
2.56.1.2	0,8492	0,7431	0,6815	0,8884	0,8490	0,8440	0,8492
2.9.1.2	0,9192	0,9503	0,7707	0,9534	0,9420	0,9430	0,9503

Çizelge 6.1 devam ediyor.

Aile	N=1	N=2	N=3	N=1–3	**ORT**	**AORT**	**MAX**
2.9.1.3	0,9928	0,9980	0,9695	0,9989	0,9990	0,9990	0,9980
2.9.1.4	0,9832	0,9851	0,9177	0,9923	0,9930	0,9940	0,9851
3.1.8.1	0,9694	0,9727	0,9415	0,9904	0,9790	0,9790	0,9727
3.1.8.3	0,9668	0,9558	0,9643	0,9780	0,9760	0,9760	0,9668
3.2.1.2	0,7182	0,7897	0,7738	0,8323	0,8160	0,8180	0,7897
3.2.1.3	0,7553	0,7438	0,7204	0,7729	0,7730	0,7730	0,7553
3.2.1.4	0,7120	0,9292	0,9020	0,9410	0,9090	0,9190	0,9292
3.2.1.5	0,8851	0,8791	0,8604	0,8959	0,9210	0,9220	0,8851
3.2.1.6	0,7793	0,7630	0,8331	0,8662	0,8310	0,8320	0,8331
3.2.1.7	0,8524	0,9417	0,9788	0,9659	0,9570	0,9600	0,9788
3.3.1.2	0,7741	0,8182	0,7883	0,8950	0,8550	0,8550	0,8182
3.3.1.5	0,7038	0,8996	0,8110	0,8969	0,8610	0,8720	0,8996
3.32.1.1	0,8879	0,9103	0,9094	0,9322	0,9270	0,9260	0,9103
3.32.1.11	0,8456	0,9444	0,9477	0,9748	0,9620	0,9650	0,9477
3.32.1.13	0,6589	0,8824	0,8288	0,8856	0,8520	0,8620	0,8824
3.32.1.8	0,7636	0,8096	0,8745	0,8768	0,8690	0,8720	0,8745
3.42.1.1	0,7060	0,6856	0,5756	0,7549	0,7350	0,7360	0,7060
3.42.1.5	0,5990	0,7128	0,4801	0,6728	0,7240	0,7300	0,7128
3.42.1.8	0,7583	0,6877	0,6504	0,6899	0,6920	0,6900	0,7583
7.3.5.2	0,9255	0,9872	0,9899	0,9778	0,9720	0,9720	0,9899
7.3.6.1	0,9529	0,9661	0,9878	0,9901	0,9830	0,9820	0,9878
7.3.6.2	0,8077	0,9515	0,9519	0,9618	0,9240	0,9330	0,9519
7.3.6.4	0,9963	0,9794	0,9847	0,9959	0,9960	0,9960	0,9963
7.39.1.2	0,9794	0,9168	0,7938	0,9379	0,9890	0,9890	0,9794
7.39.1.3	0,8203	0,8349	0,9306	0,8292	0,8860	0,8860	0,9306
7.41.5.1	0,8325	0,8468	0,5773	0,8414	0,8620	0,8620	0,8468
7.41.5.2	0,7443	0,9816	0,6834	0,9806	0,8780	0,8990	0,9816
ortalama	**0,8091**	**0,8464**	**0,7807**	**0,8741**	**0,8623**	**0,8672**	**0,8820**
stdsapma	0,1386	0,1328	0,1820	0,1358	0,1318	0,1285	0,1103

Çizelge 6.1 devam ediyor.

Aile	N=1	N=2	N=3	N=1–3	**ORT**	**AORT**	**MAX**
max	0,9963	0,9980	0,9899	0,9989	0,9990	0,9990	0,9980
min	0,2958	0,3943	0,2957	0,2314	0,2840	0,2870	0,3943

Protein homoloji tespiti için, çıkarılan ROC puanlarının hesaplanmasında sınıflandırıcılardan çıktı olarak alınan sınıflandırıcı puanları büyük önem taşımaktadır. Sınıflandırıcılar bu puanları hesaplarken nitelik vektörlerini girdi olarak kabul etmektedirler. Sınıflandırıcılara gönderilmek üzere çıkarılan nitelik vektörlerinin boyutları her protein için eşit olmak zorundadır, aksi taktirde sınıflandırıcı işlevselliğini yitirmektedir. Elimizdeki veri kümesi için, aile bazında her bir proteinin sınıflandırıcıya gönderilmek üzere bir nitelik vektörü mevcuttur. Bu nitelik vektörlerinin her bir boyutu proteinin içerisindeki aminoasitlerin kendi başlarına ya da diğer komşu aminoasitler ile bağlanarak oluşan aminoasit grubunun, tüm protein dizilimi içinde görülme sıklığı olarak nitelendirilmiştir, başka bir deyişle aminoasidin ya da aminoasit grubunun dizilim geneline olan oranının istatistiği olarak tanımlanmaktadır.

Nitelik vektörlerinin çıkarılmasında, protein diziliminde yer alan aminoasitler arasındaki peptit bağlarının sayısını arttırmak mümkündür. Ancak n-peptit kompozisyonlarındaki bu artış, nitelik vektörlerinin boyutlarına üssel olarak yansımaktadır. Başka bire deyişle, basit bir algoritma, uzunluğu *m* olan bir protein dizilimindeki olası n-peptit grupları için $O(m20^n)$ zaman karmaşıklığına sahiptir.

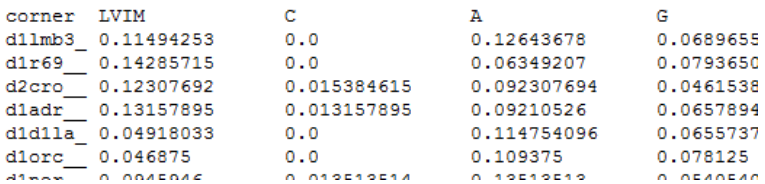
Dizilim içindeki n-peptit kompozisyonlara bakılarak önceden Şekil 6.1'de de görüldüğü üzere aileler içerisindeki proteinlerin birçoğu farklı dizilimlere ve uzunluklara sahiptirler. Sınıflandırıcılara gönderilmek için, bu dizilimlerden, dizilim içindeki n-peptit kompozisyonlarına bakılarak nitelik vektörleri çıkarılır. Nitelik vektörlerinin çıkartılmasında başka bir yöntem olarak protein dizilimi içerisindeki aminoasitler gruplandırılır.

Çizelge 6.2'de işte bu şekilde ailelere ait protein dizilimleri içerisindeki aminoasitler gruplandırılarak, elde edilmiş yeni nitelik vektörlerinin sınıflandırıcılara gönderilmesiyle ortaya çıkan DVM performansları verilmiştir. Nitelik vektörleri ortaya çıkarılırken, benzer aminoasitler önceki bölümlerde anlatılan bu yöntemler bir arada gruplandırılmıştır. Aynı gruba üye olan aminoasitler nitelik vektörünün tek bir boyutu olarak kabul edilmiştir.

Çizelgelerde aile başlığı ile verilen sütun deneyde kullanılan SCOP veri kümesine ait aileleri belirten kimliklerdir. Proteinler üzerinde yapılan karşılaştırmalar proteinlerin aile isimleri kullanılmadan, kısaca sahip oldukları kimlikler ile yapılmıştır. Çizelge 6.1 üzerinde sırasıyla her sütun başlığı ve sütun içeriği kısaca anlatılmıştır.

Çizelge 6.2'de ikinci sütunda Σ 20 ile ifade edilen DVM performans değerleri, Çizelge 6.1'de N=1 başlığına sahip sütundaki performans değerleriyle aynıdır. Çünkü iki gösterimde eş anlamlıdır. Alfabe boyutu 20 olduğundan dolayı yani herhangi bir gruplama yapılmadığından dolayı, nitelik vektörleri de 20 boyutludur. Şekil 6.2'de örnek bir aileye ait proteinlerin nitelik vektörleri gösterilmiştir. Bu vektörler, sınıflandırıcıya gönderilir.

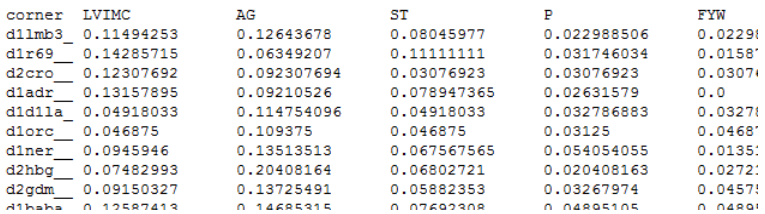
Üçüncü sütunda DVM performansları Σ 15 için verilmiştir. Σ 15 ile gösterilen alfabenin on beş elemandan oluşmasıdır. Bu durumda anlaşılan protein dizilimlerinde amino asitlerin gruplandırılmasıdır. Nitelik vektörleri oluşturulurken benzer aminoasitler aynı boyutun elemanı olarak sayılırlar ve hesaplamalara bu şekilde katılırlar. Şekil 6.6, 1.36.1.2 ailesine ait protein dizilimlerinin 15 boyutlu nitelik vektörlerini göstermektedir. Şekildeki ikinci sütun incelendiğinde de rahatlıkla görülmektedir ki, alfabenin dört elemanı tek bir başlık yani tek bir boyut altında birleştirilmiştir ve tüm proteinler bu şekilde ifade edilmiştir.



corner	LVIM	C	A	G
d1lmb3_	0.11494253	0.0	0.12643678	0.0689655
d1r69__	0.14285715	0.0	0.06349207	0.0793650
d2cro__	0.12307692	0.015384615	0.092307694	0.0461538
d1adr__	0.13157895	0.013157895	0.09210526	0.0657894
d1d1la_	0.04918033	0.0	0.114754096	0.0655737
d1orc__	0.046875	0.0	0.109375	0.078125

Şekil 6.6 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 15 Boyutlu

Dördüncü sütunda alfabedeki eleman sayısı sekize indirilmiştir, yani benzer aminoasitler daha büyük gruplar altında toplanıp, protein dizilimleri için ortaya çıkarılan nitelik vektörleri sekiz boyutlu olarak karşımıza çıkmaktadır. Şekil 6.7, 1.36.1.2 ailesine ait protein dizilimlerinin 8 boyutlu nitelik vektörlerini göstermektedir. 15 boyutlu nitelik vektörlerinde ilk sütun 4 aminoasitle gruplandırılmışken, 8 boyutlu nitelik vektörlerinde bu sütunda 5 aminoasit grubu oluşturur. Benzer aminoasitler daha genel bir çerçevede toplanırlar. Sınıflandırıcıya bu haliyle gönderilir.



corner	LVIMC	AG	ST	P	FYW
d1lmb3_	0.11494253	0.12643678	0.08045977	0.022988506	0.0229
d1r69__	0.14285715	0.06349207	0.11111111	0.031746034	0.0158
d2cro__	0.12307692	0.092307694	0.03076923	0.03076923	0.0307
d1adr__	0.13157895	0.09210526	0.078947365	0.02631579	0.0
d1d1la_	0.04918033	0.114754096	0.04918033	0.032786883	0.0327
d1orc__	0.046875	0.109375	0.046875	0.03125	0.0468
d1ner__	0.0945946	0.13513513	0.067567565	0.054054055	0.0135
d2hbg__	0.07482993	0.20408164	0.06802721	0.020408163	0.0272
d2gdm__	0.09150327	0.13725491	0.05882353	0.03267974	0.0457

Şekil 6.7 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 8 Boyutlu

Son sütundaki yöntem diğerlerinden biraz daha farklıdır, aminoasit alfabesi 20 elemanıyla alınmaktadır. Ancak bu alfabe diğer gruplanmış aminoasitler ile birleştirilmiştir. Böylelikle nitelik vektörünün boyutu arttırılmıştır. Bu birleştirmede alfabe yirmi, on beş ve sekiz elemandan oluşmaktadır ancak nitelik vektörünün boyutu toplam değildir. Gruplanmamış tekil aminoasitlerin bir kısmı yani aynen tekrar edene elemanları yeniden hesaplamaya katılmamıştır. Dolayısıyla elimizde sınıflandırıcılara gönderilmek üzere hazırlanmış 28 boyutlu yeni nitelik vektörleri mevcuttur. Ailelerin sahip oldukların proteinlerin dizilimleri farklı bir yorumla tek bir birime ölçeklendirilir. Şekil 6.8'de sınıflandırıcıya gönderilmek üzere bekleyen nitelik

vektörleri gösterilmiştir. Protein sayısındaki yeni nitelik vektörleri sınıflandırıcılara gönderilir ve sonuçları elde edilmiştir, bakınız Çizelge 6.2.

corner	L	V	I		LVIM	FY	KR
d1lmb3_	0.11494253	0.057471264	0.057471264		0.11494253	0.022988506	0.0804597
d1r69__	0.14285715	0.06349207	0.04761905		0.14285715	0.015873017	0.0793650
d2cro__	0.12307692	0.046153847	0.06153846		0.12307692	0.03076923	0.0923076
d1adr__	0.13157895	0.05263158	0.039473683		0.13157895	0.0	0.0789473
d1d1la_	0.04918033	0.04918033	0.08196721	.....	0.04918033	0.032786883	0.0983606
d1orc__	0.046875	0.0625	0.078125		0.046875	0.046875	0.109375
d1ner__	0.0945946	0.027027028	0.054054055		0.0945946	0.013513514	0.0810810

Şekil 6.8 – 1.36.1.2 Ailesine Ait Proteinlerin Nitelik Vektörleri 28 Boyutlu

Nitelik vektörlerinin çıkarılması ile ilgili verilen son dört yöntemde, aminoasitlerin gruplandırılması, aminoasit alfabelerinin azaltılmasıyla yani alfabe içerisindeki birkaç elemanı bir araya toplayıp tek bir eleman olarak temsil edilerek ortaya konulmaktadır, alfabeler ve elemanlarının gruplandırılma şekilleri, Şekil 4.1.2 detaylı bir şekilde gösterilmiştir.

Çizelge 6.2 – Aileler için ROC Sonuçları (aminoasit gruplama)

Aile	Σ 20	Σ 15	Σ A8	Σ 20–8	**Σ ORT**	**Σ AORT**	**Σ MAX**
1.27.1.2	0,9311	0,8893	0,5355	0,9505	0,8688	0,8806	0,9311
1.36.1.2	0,7555	0,9326	0,3904	0,8103	0,8183	0,8532	0,9326
1.36.1.5	0,6603	0,9206	0,9636	0,8181	0,9058	0,9378	0,9636
1.4.1.1	0,9902	0,9617	0,3752	0,9984	0,9664	0,9859	0,9902
1.4.1.2	0,9165	0,7884	0,5846	0,9215	0,9046	0,9068	0,9165
1.4.1.3	0,9690	0,9805	0,2748	0,9741	0,9832	0,9880	0,9805
1.41.1.2	0,7813	0,5946	0,4024	0,7285	0,6222	0,6439	0,7813
1.41.1.5	0,9242	0,6059	0,6734	0,9373	0,8588	0,8800	0,9242
1.45.1.2	0,8690	0,7230	0,8090	0,8643	0,8673	0,8590	0,8690
2.1.1.1	0,7003	0,6339	0,5892	0,6892	0,6896	0,6829	0,7003
2.1.1.2	0,9044	0,9117	0,6752	0,9226	0,9156	0,9150	0,9117
2.1.1.4	0,7603	0,7931	0,5994	0,7473	0,7821	0,7811	0,7931

Çizelge 6.2 devam ediyor.

Aile	Σ 20	Σ 15	Σ A8	Σ 20–8	**Σ ORT**	**Σ AORT**	**Σ MAX**
2.1.1.5	0,6852	0,5959	0,5766	0,6500	0,6306	0,6328	0,6852
2.28.1.1	0,4958	0,3574	0,6847	0,4983	0,4927	0,5620	0,6847
2.28.1.3	0,8317	0,8241	0,4775	0,8394	0,8602	0,8566	0,8317
2.38.4.1	0,8101	0,7635	0,3706	0,9377	0,7834	0,7984	0,8101
2.38.4.3	0,8234	0,7019	0,4581	0,7656	0,7078	0,7216	0,8234
2.38.4.5	0,8438	0,5812	0,6594	0,8267	0,7759	0,7907	0,8438
2.44.1.2	0,2958	0,2147	0,2460	0,2273	0,2048	0,2069	0,2958
2.5.1.1	0,8694	0,8248	0,5836	0,8335	0,8283	0,8294	0,8694
2.5.1.3	0,7447	0,7249	0,4539	0,7421	0,7004	0,7048	0,7447
2.52.1.2	0,5636	0,6190	0,2442	0,5194	0,5037	0,5318	0,6190
2.56.1.2	0,8492	0,8605	0,6493	0,8676	0,9180	0,9176	0,8605
2.9.1.2	0,9192	0,7309	0,4738	0,9056	0,8366	0,8462	0,9192
2.9.1.3	0,9928	0,9739	0,6178	0,9868	0,9733	0,9762	0,9928
2.9.1.4	0,9832	0,9849	0,8017	0,9723	0,9789	0,9775	0,9849
3.1.8.1	0,9694	0,8972	0,6420	0,9399	0,9558	0,9562	0,9694
3.1.8.3	0,9668	0,9461	0,4673	0,9686	0,9697	0,9699	0,9668
3.2.1.2	0,7182	0,7011	0,5629	0,7149	0,6962	0,6953	0,7182
3.2.1.3	0,7553	0,7180	0,7210	0,6650	0,8104	0,8123	0,7553
3.2.1.4	0,7120	0,7808	0,5770	0,7009	0,7879	0,7896	0,7808
3.2.1.5	0,8851	0,9103	0,6002	0,8806	0,9151	0,9159	0,9103
3.2.1.6	0,7793	0,6642	0,7684	0,7358	0,7595	0,7605	0,7793
3.2.1.7	0,8524	0,7091	0,2785	0,7812	0,7807	0,8198	0,8524
3.3.1.2	0,7741	0,6784	0,2772	0,7721	0,6837	0,7062	0,7741
3.3.1.5	0,7038	0,8247	0,3698	0,7487	0,7473	0,7794	0,8247
3.32.1.1	0,8879	0,7741	0,6968	0,8740	0,8968	0,9010	0,8879
3.32.1.11	0,8456	0,6741	0,2523	0,8252	0,7534	0,7934	0,8456
3.32.1.13	0,6589	0,6671	0,4625	0,6925	0,6792	0,6773	0,6671
3.32.1.8	0,7636	0,7885	0,4602	0,8201	0,7462	0,7512	0,7885
3.42.1.1	0,7060	0,6383	0,3460	0,6905	0,6029	0,6110	0,7060

Çizelge 6.2 devam ediyor.

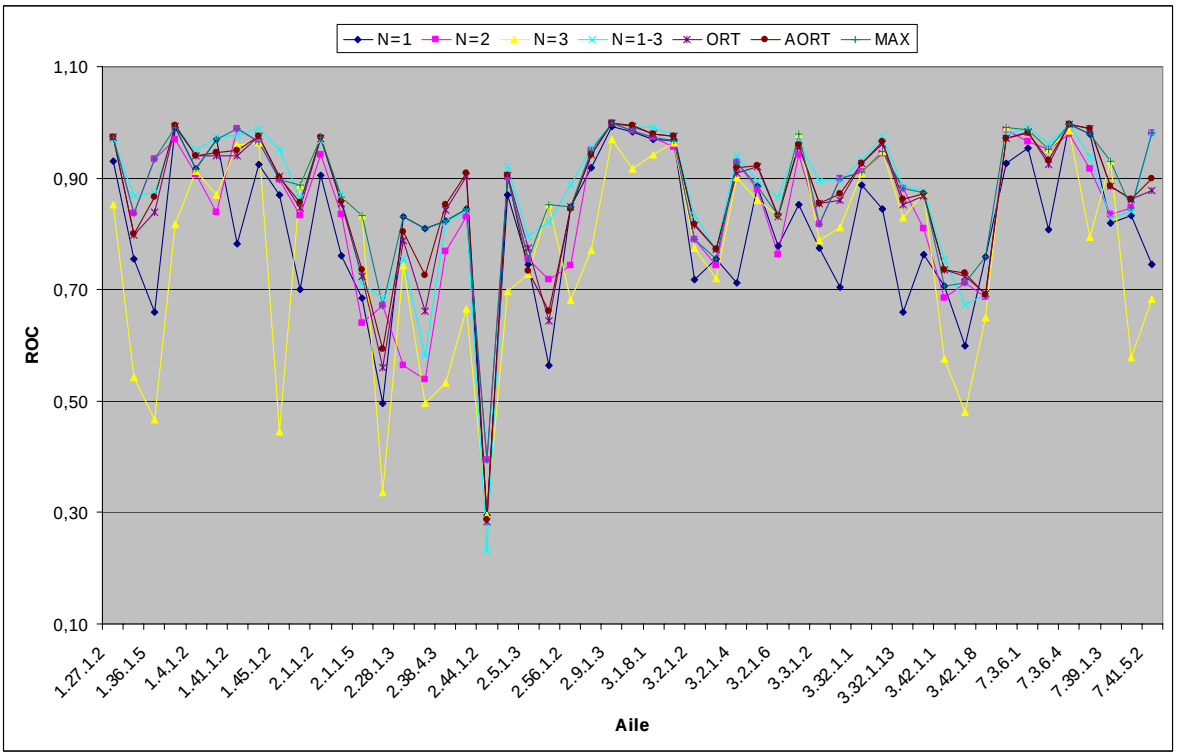
Aile	Σ 20	Σ 15	Σ A8	Σ 20–8	**Σ ORT**	**Σ AORT**	**Σ MAX**
3.42.1.5	0,5990	0,6274	0,2521	0,5816	0,5817	0,6118	0,6274
3.42.1.8	0,7583	0,6297	0,6489	0,7431	0,7500	0,7562	0,7583
7.3.5.2	0,9255	0,6113	0,6560	0,9247	0,8505	0,8639	0,9873
7.3.6.1	0,9529	0,9289	0,8589	0,9621	0,9631	0,9628	0,9529
7.3.6.2	0,8077	0,6857	0,7083	0,8423	0,8015	0,8078	0,8077
7.3.6.4	0,9963	0,9967	0,8961	0,9996	0,9967	0,9963	0,9967
7.39.1.2	0,9794	0,9551	0,8376	0,9783	0,9853	0,9846	0,9794
7.39.1.3	0,8203	0,9286	0,8664	0,8340	0,9388	0,9430	0,9286
7.41.5.1	0,8325	0,6993	0,6523	0,8024	0,7921	0,7955	0,8325
7.41.5.2	0,7443	0,7702	0,7592	0,8245	0,8129	0,8220	0,7702
ortalama	**0,8091**	**0,7588**	**0,5645**	**0,8086**	**0,7968**	**0,8069**	**0,8339**
stdsapma	0,1386	0,1592	0,1900	0,1473	0,1520	0,1476	0,1297
max	0,9963	0,9967	0,9636	0,9996	0,9967	0,9963	0,9967
min	0,2958	0,2147	0,2442	0,2273	0,2048	0,2069	0,2958

Buraya kadar ailelere ait protein dizilimleri için nitelik vektörlerinin nasıl çıkarıldığı ve farklı yöntemlerle çıkarılan bu nitelik vektörlerinin ortaya koydukları sınıflandırıcı performansları verilmiştir. Yukarıda verilen sekiz farklı yöntem içinde yaklaşım aynıdır, nitelik vektörleri çıkarıldıktan sonra tek bir sınıflandırıcıya verilmiştir ve sonuç bu şekilde alınmıştır. Bu sonuçları iyileştirmek için n-peptit birleşimine ait tüm özellik girdileri tek bir sınıflandırıcıya gönderilmek yerine, belirli anlamlı guruplara ayrılmıştır. Bu doğrultuda, önceden çıkarılan çok boyutlu nitelik vektörleri, grup sayılarıyla, belirlenen birden fazla sınıflandırıcıya verilip bir üst sınıflandırma yaklaşımı yapılmıştır. Bu yaklaşım temelde Şekil 5.4.2'de önerilen modeli kullanmaktadır. Nitelik vektörleri grup grup sınıflandırıcılara gönderilmektedir, her sınıflandırıcının çıktı olarak verdiği sınıflandırıcı puanı üzerinden değerlendirici yöntemler ile protein homoloji tespiti için bir ROC sonuçları hesaplanmaktadır.

Bu tezde önerilen model kullanılarak, kompozisyonlar üzerinde üç farklı yöntem uygulanmıştır. Bunlar çizelgelerde de veriliş sıralarıyla, ortalama (ORT) alma, ağırlıklı ortalama (AORT) alma ve maksimum (MAX) almadır. Modele göre, sayısı birden fazla olan n-peptit kompozisyonlarına ait nitelik vektörleri sınıflandırıcılara aynı anda verilmiştir, deneylerde kullanılan 1-peptit, 2-peptit ve 3-peptit kompozisyonları için üç farklı sınıflandırıcı kullanılmaktadır. Sınıflandırma sonunda her sınıflandırıcının verdiği sınıflandırıcı puanları, modelde belirtilen değerlendirme aşamasına girmektedir ve seçilen metoda için ROC performansları hesaplanmaktadır. SCOP veri kümesi üzerinde yapılan deneyler sonucu ailelere ait ROC performansları hesaplanmıştır ve bu değerler Çizelge 6.1'de sırasıyla son üç sütunda ortalama, ağırlıklı ortalama ve maksimum şeklinde verilmiştir. Önerilen model benzer şekilde, küçültülen aminoasit alfabelerinden elde edilen 1-peptit kompozisyonları üzerinde de uygulanmış, ortalama alma, ağırlıklı ortalama alma ve maksimum alma yöntemleri denenmiştir. Sonuç olarak elde edilen ROC performansları Çizelge 6.2'de son üç sütunda aynı sırayla verilmiştir.

Bu tezde homoloji tespiti için önerilen, denenen ve uygulanan, model, yöntemler ve sınıflandırıcılar sonucu ortaya çıkan değerler tablolarda verilmiştir, Çizelge 6.1 ve 6.2'ye bakıldığında ailelere ait performans değerleri ve her sütunun altında kullanılan yöntemlere ait tüm ailelerin verdiği ortalama, standart sapma, minimum ve maksimum değerleri mevcuttur. Ancak sadece sayı değerleri bakıldığında çok da anlam ifade edemeyebilmektedir.

Sonuçlar üzerinde daha ileri düzeyde inceleme yapabilmek için çizelgelerde verilen değerler belirli ölçütler çerçevesinde yeniden değerlendirilmiş bu doğrultuda yeni çizgeler ve grafikler oluşturulmuştur. Yapılan değerlendirmeler, ilk önce Çizelge 6.1 daha sonrada Çizelge 6.2 üzerinde yapılmıştır. İlk önce n-peptit kompozisyonları ile yapılan çalışmaların sonuçları daha sonrada küçültülmüş alfabeler ile 1-peptit kompozisyonları üzerinde yapılan çalışmaların sonuçları verilmiştir.



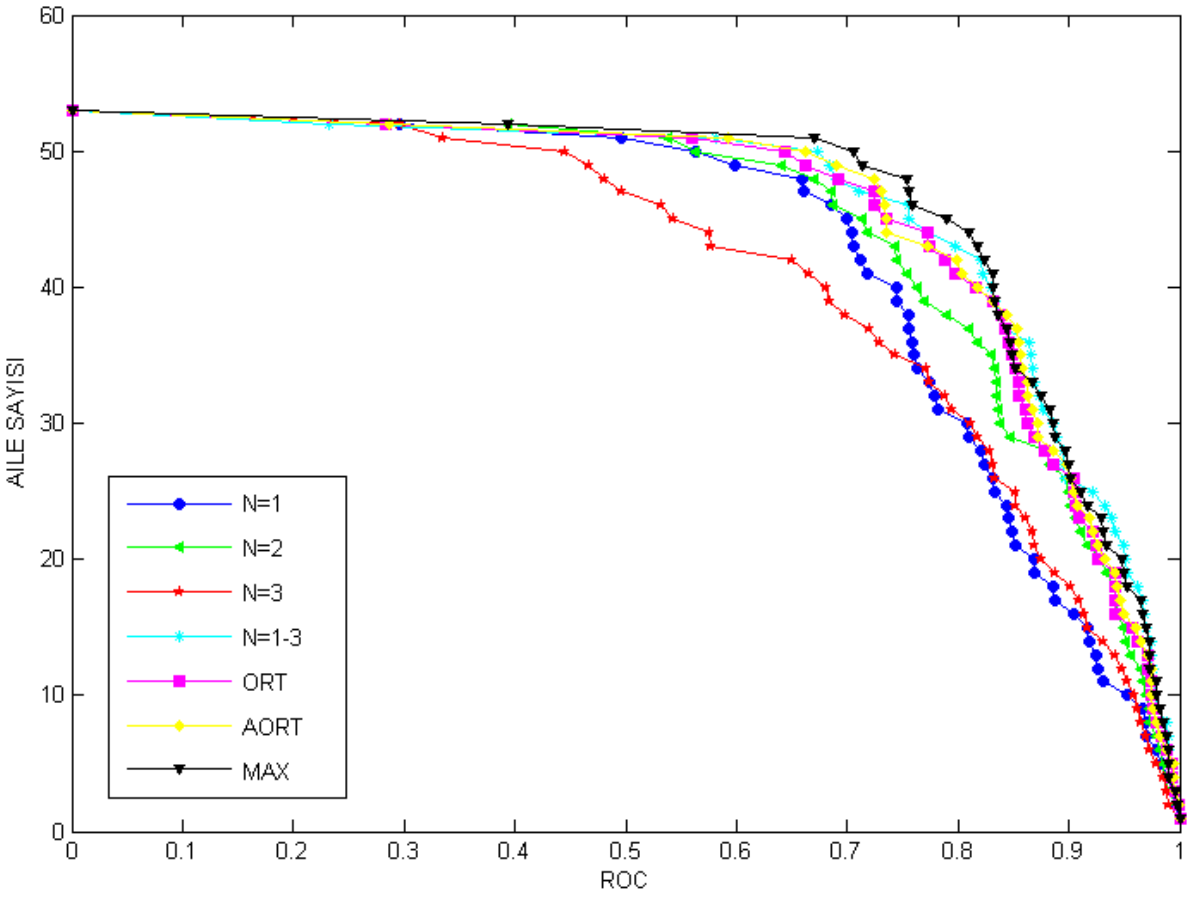
Şekil 6.9 – Ailelerin ROC Performansları (n-peptit birleşimler)

Aile bazında hesaplanan ROC performanslarını daha rahat görebilmek açısından Şekil 6.9'da ailelerin ROC performansları grafik verilmiştir, burada n-peptit kompozisyonlar için kullanılan yöntemlerin sonuçları çizgi grafik üzerinde gösterilmiştir. Aileler üzerinde tüm yöntemlerin sergilediği davranışlar rahat bir grafik üzerinde rahat bir şekilde görünmektedir, genelde kullanılan tüm yöntemlerin aileler üzerindeki davranışı benzerdir ancak belirli ailelerde ya da soylarda bu genelleme bozulabilmektedir. Örneğin genelde n=1 ile n=3 için grafik incelenecek olursa, veri kümesindeki ilk altı aile üzerinde, yöntemlerin davranışları aynıdır. Bir yöntemde bir sonraki aile için performansta bir düşme yaşanırsa aynı ölçüde olmasa dahi diğerinde de bu durum yaşanmaktadır, ama ne var ki 1.41.x.x soyundan gelen ailelerde n=1 metodu için performans düşerken, n=3 metodunda ise çok iyi bir artış gözlemlenmektedir. Yöntemler aileler üzerinde sahip oldukları performansları açısından çok büyük ölçüde nicel farklılıklar göstermektedirler.

Çizelge 6.3 – Homoloji Tespiti için Kullanılan Yöntemlerin T-testi Puanları (n-peptit birleşimleri)

	N=1	N=2	N=3	N=1-3	ORT	AORT	MAX
N=1		1,83E-02	1,77E-01	7,38E-06	5,47E-07	1,23E-07	9,76E-08
N=2			1,60E-03	4,01E-04	6,46E-02	1,76E-02	2,24E-04
N=3				4,35E-06	2,59E-05	1,37E-05	5,14E-07
N=1-3					6,31E-02	2,82E-01	2,98E-01
ORT						9,14E-03	8,56E-03
AORT							2,05E-02
MAX							

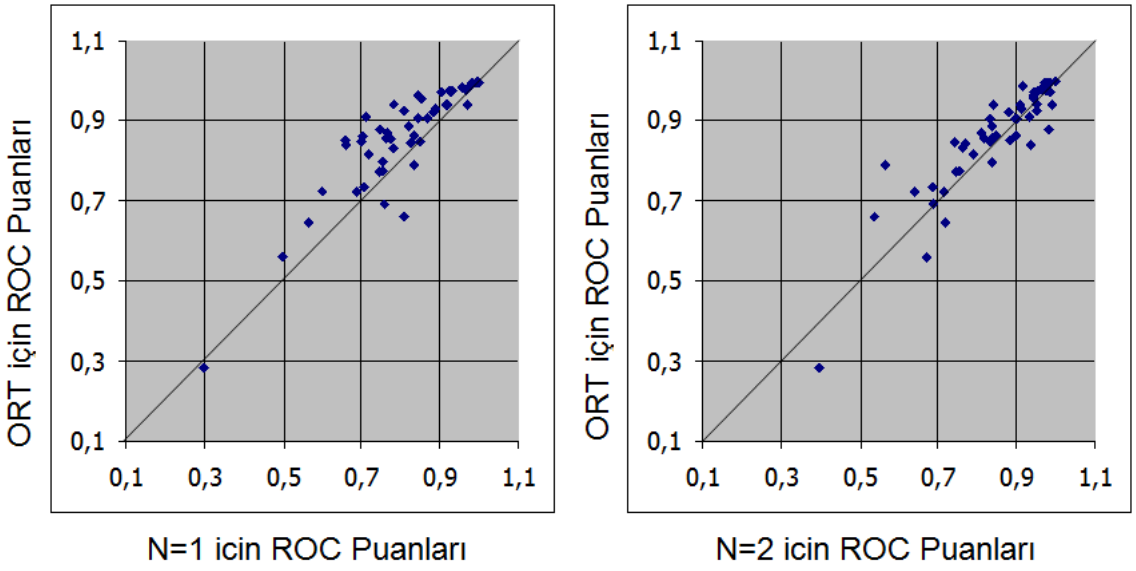
Tez'de kullanılan homoloji tespiti yöntemlerin birbirlerine göre istatistiksel anlamlarına bakabilmek için yöntemler üzerinde ikişerli olarak t-testleri yapılmıştır ve test sonuçları Çizelge 6.3'te verilmiştir. Çizelgede verilen her bir girdi, çift kuyruklu dağılımda eşli t-testi tarafından 51 aile için her iki yöntemden gelen eşli ROC puanlarının kıyaslanması ile verilen p-değerleridir. Çizelgede listelenmiş her bir girdi, ilgili satırdaki metodun ilgili sütundaki metoda göre istatistiksel olarak daha anlamlı olup olmadığını anlatmaktadır. Çizelgede verilmiş olan girdi değeri, 0.05 değerinden daha düşükse yöntem istatistiksel olarak t-teste girdiği eş metoda göre daha anlamlı olmaktadır. Bu durumda çizelge incelendiğinde görülmektedir ki MAX, N=1'e göre istatistiksel olarak anlamlıdır, çünkü çizelgede verilen değer çok küçüktür. Ancak MAX ile N=1–3 arasında akside iddia edilemezken, bu durum da söz konusu değildir. Bunun nedeni, MAX ile N=1–3 yöntemleri biçimsel olarak birbirlerinden farklı olmalarına karşın aslında yöntemler içerisinde yapılan iş matematiksel olarak birbirine benzemektedir.



Şekil 6.9 – Homoloji Tespiti Yöntemlerinin Göreceli Performansları (n-peptit birleşimler için)

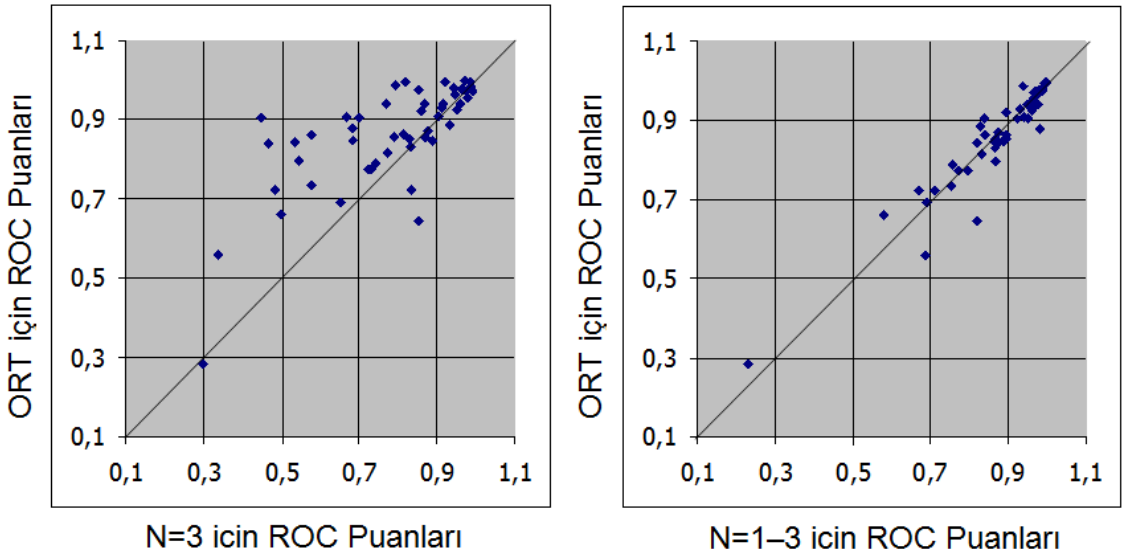
Homoloji tespiti için n-peptit birleşimler esas alınarak, SCOP veri kümesindeki ailelere uygulanan yedi yöntem, Şekil 6.9'da verilen ROC – aile sayısı grafiği aracılığıyla, göreceli performanslarına göre karşılaştırılmıştırlar. Grafikte çizilmiş olan her eğri, kullanılan bir metodu ifade etmektedir. Çizgiler üzerideki her indis metodun ROC eşik değerini aştığı noktaları başka bir değişle veri kümesi içerisindeki aileleri ifade etmektedir. Veri kümesinde toplam 51 aile mevcut olduğu için her eğri üzerinde 51 indis bulunmaktadır. Her çizim için, daha yüksekte bulunan eğri daha kusursuz homoloji tespiti performansa sahiptir anlamına gelmektedir. Şekildeki eğrilere bakıldığında görülmektedir ki, en üstteki eğri MAX eğrisidir, genelde en altta bulunan eğri ise N=3 eğrisidir. Sonuç olarak şekildeki grafikten MAX metodu diğer yöntemlerden daha iyi çalışmaktadır, çıkarımı yapılabilmektedir.

Sonuçlar aynı zamanda ikili karşılaştırma grafikleri kullanılarak yöntemlere göre aileden aileye de karşılaştırılmıştır. Kullanılan bu grafiklerde x ve y eksenleri yöntemlere ait ROC performanslarını göstermektedir. Grafiklerde verilen mavi noktalar SCOP veri kümesi içerisindeki ailelere tekabül etmektedir. Grafiği ortadan ikiye ayıran köşegen çizgisi ise aslında yöntemleri birbirinden ayırmaktadır, bir anlamda sınır olarak kabul edilmektedir.



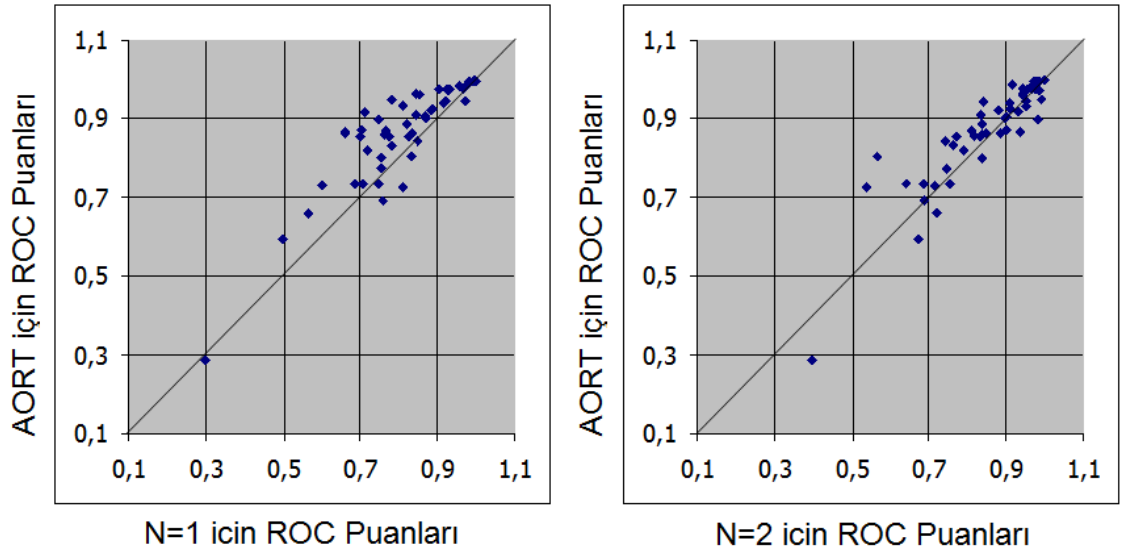
Şekil 6.10 – ORT'a karşı N=1(a) ve N=2(b)

Veri kümesindeki ailelerden örnek verilecek olunursa, 2.44.1.2 ailesi için ORT, N=1 ve N=2 yöntemlerinin ROC puanları sırayla 0,2940, 0,2958 ve 0,3943'dir. Şekil 6.10(a)'da grafiğin alt kısmında sınırın N=1 tarafında kalan mavi nokta 2.44.1.2 ailesini ifade etmektedir. Görüldüğü gibi noktanın eksenlere olan iz düşümünün gösterdiği değerler, Çizelge 6.1'de verilen değerlerdir. Aynı aile için Şekil 6.10(b)'de aile daha belirgin bir şekilde N=2 metodu tarafında yer almaktadır, çünkü ROC puanı belirgin bir şekilde ORT'a göre daha iyidir. Her ne kadar bir ya da birkaç aile için bu böyle de olsa, grafiklere bakıldığında görülmektedir ki ORT metodu, her iki şekilde de, aile bazında grafikteki diğer yöntemlere göre daha iyi sonuçlar vermiştir.



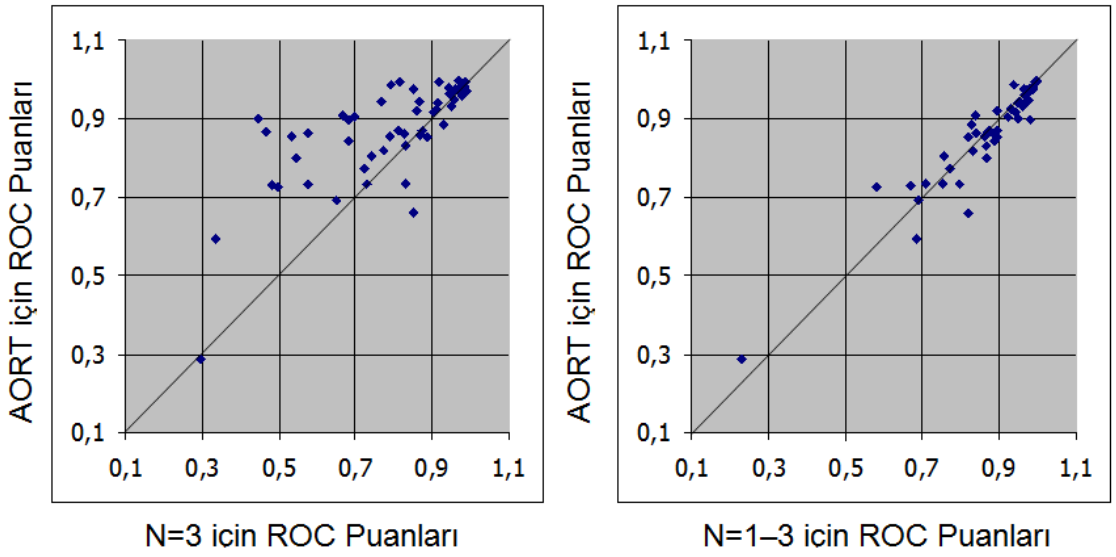
Şekil 6.11 – ORT'a karşı N=3(a), N=1–3(b)

Şekil 6.11'de, ORT ile diğer yöntemlerden N=3 ve N=1–3'ün karşılaştırılması verilmektedir. Şekil 6.11(a) için çok kesin bir biçimde ORT metodu daha başarılıdır, ancak Şekil 6.11(b) için aynısı söylenemez çünkü iki metodun karşılaştırılmasının gösterildiği bu grafikte ailelerin alanlar ve sınır üzerindeki dağımı çok da ayırt edici değildir. Dolayısıyla yöntemlerin verdiği sonuçlar burada detaylı bir karşılaştırma amacıyla daha büyük önem taşımaktadır.



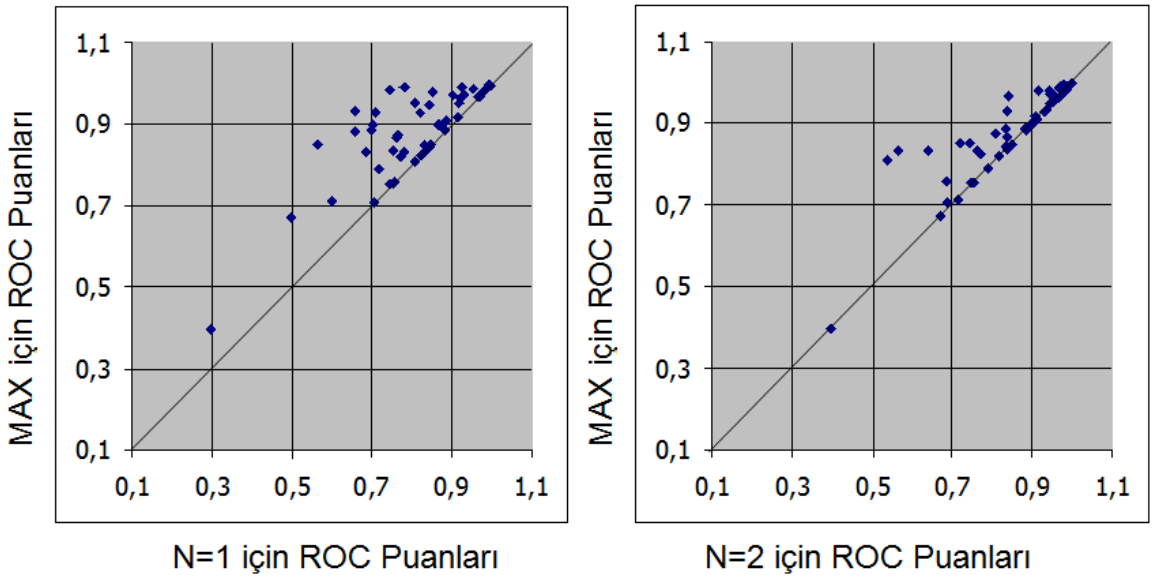
Şekil 6.12 – AORT'a karşı N=1(a) ve N=2(b)

Şekil 6.12'de aileler ağırlıklı ortalama, 1-peptit ve 2-peptit yöntemlerinden aldıkları ROC puanlarına göre karşılaştırılmaktadır. Şekil 6.12(a)'ya baktığımızda AORT'un N=1 üzerinde yüksek performans gösterdiğini gözle belirgin bir şekilde gözükmektedir. Ancak N=2 ile karşılaştırıldığında, sonuçlar hemen hemen aynı görünmektedir. Şekil 6.12(b)'de ailelerin dağılımları genelde sınır üzeri ve hemen civarıdır, grafiğin tam ortasında AORT sınırları içerisinde rahatça görülen üç aile, 2.38.4.x soyundan gelen aileleri göstermektedir. Bu aileler için AORT çok daha başarılı olmuştur.



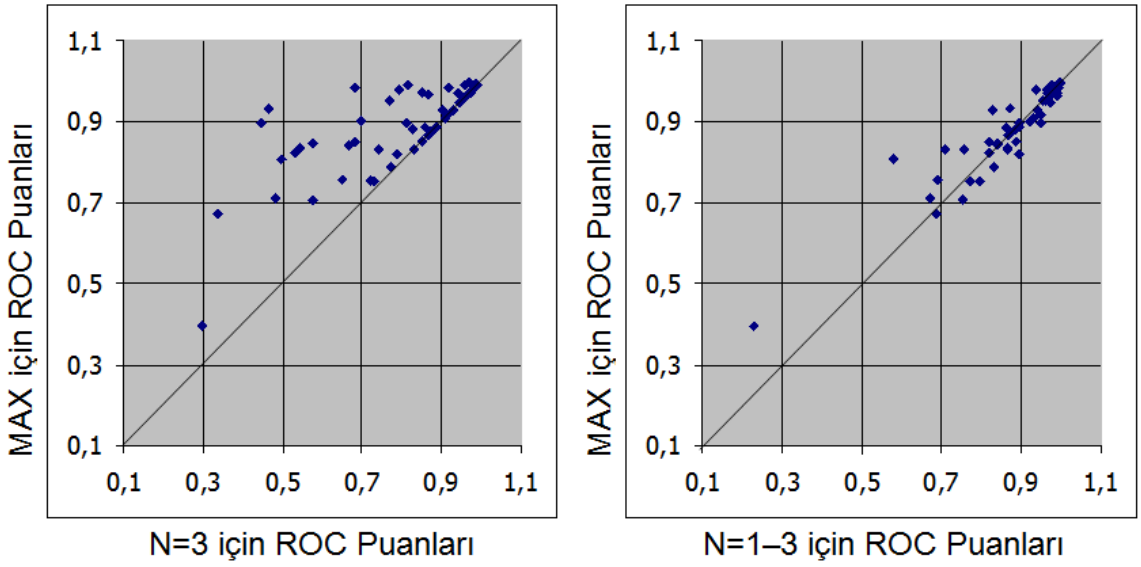
Şekil 6.13 – AORT'a karşı N=3(a), N=1–3(b)

Ağırlıklı ortalama ile 3-peptit ve 1–3-peptit yöntemlerinin aile bazında karşılaştırmaları Şekil 6.13'te verilmiştir. 3-peptit metodu burada da karşılaştırıldığı diğer yöntemlerle olduğu gibi başarısız olmuştur, Şekil 6.13(a)'da verilen grafikte görüldüğü gibi veri kümesindeki 51 aileden sadece 2 aile üzerinde AORT'a göre üstünlük sağlamaktadır. Şekil 6.13(b)'de ise AORT ile N=1–3 metodu karşılaştırılmaktadır, sınırlar içerisindeki aileler genelde eşit dağılmış gibi görünmektedir, ancak N=1–3 performans bakımından daha ileridedir.



Şekil 6.14 – MAX'a karşı N=1(a), N=2(b)

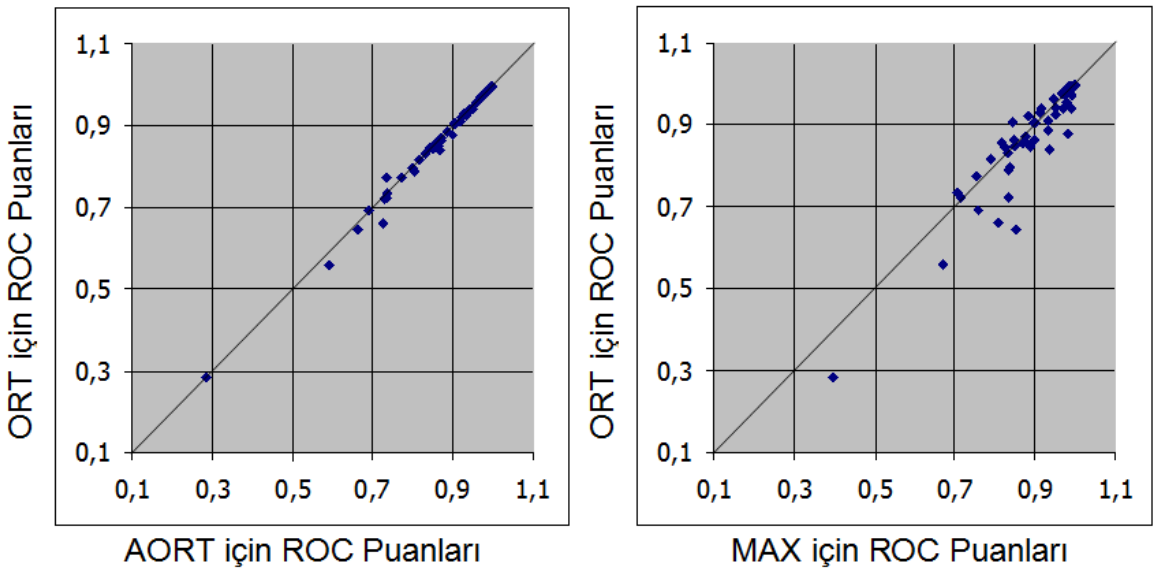
MAX yöntemi, bu tez kapsamında uygulanan yöntemler arasındaki en başarılı ve en güvenilir yöntem olarak kendini ispatlamıştır. Şekil 6.14(a), Şekil 6.14(b) ve Şekil 6.15(a), MAX metodu ile 1-peptit, 2-peptit ve 3-peptit yöntemlerinin aileler üzerindeki karşılaştırmalarıdır. Tüm grafiklerde MAX metodunun performansı en ileridedir.



Şekil 6.15 – MAX'a karşı N=3(a), N=1-3(b)

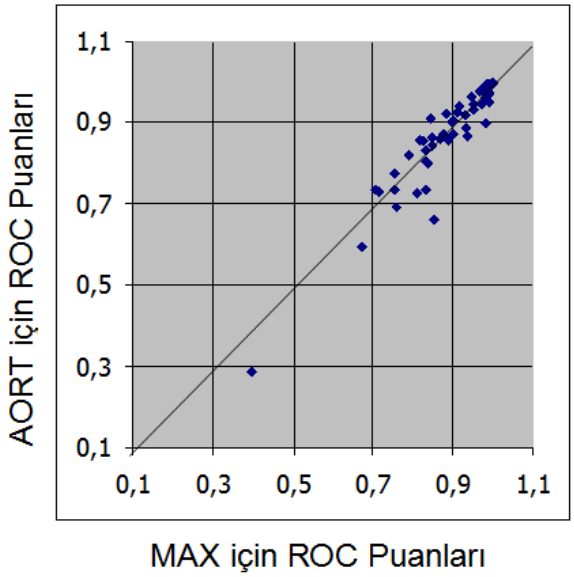
Grafikler yöntemlere ait bölgelere bakılarak incelendiğinde, tüm ailelerin MAX sınırları içerisinde olduğu, sınır üzerinde olmayanların ise sınır çizgisi üzerinde bulunduğu gözlemlenmektedir. Bu gözlemden yola çıkarak denklik durumları dışındaki diğer tüm durumlarda aileler üzerinde kazanan yöntem MAX metodu olmaktadır. Bu durum MAX metodunun N=1–3 metodu ile karşılaştırılmasında geçerliliğini yitirir, Şekil 6.15(b)'de görüldüğü üzere N=1–3 metodunun başarı sağladığı aileler mevcuttur ancak bu başarı sınır çizgisi yakınlarındadır. MAX metodu için grafiğe bakıldığında, 8 aile için metodun daha iyi performans verdiği rahatlıkla görülmektedir.

Aileler bazında yapılan ikili karşılaştırmalarda, bu noktaya kadar farklı modellere sahip yöntemler karşılaştırılmıştır. Bu modeller Şekil 5.4.1 ve 5.4.2'de verilmiştir.



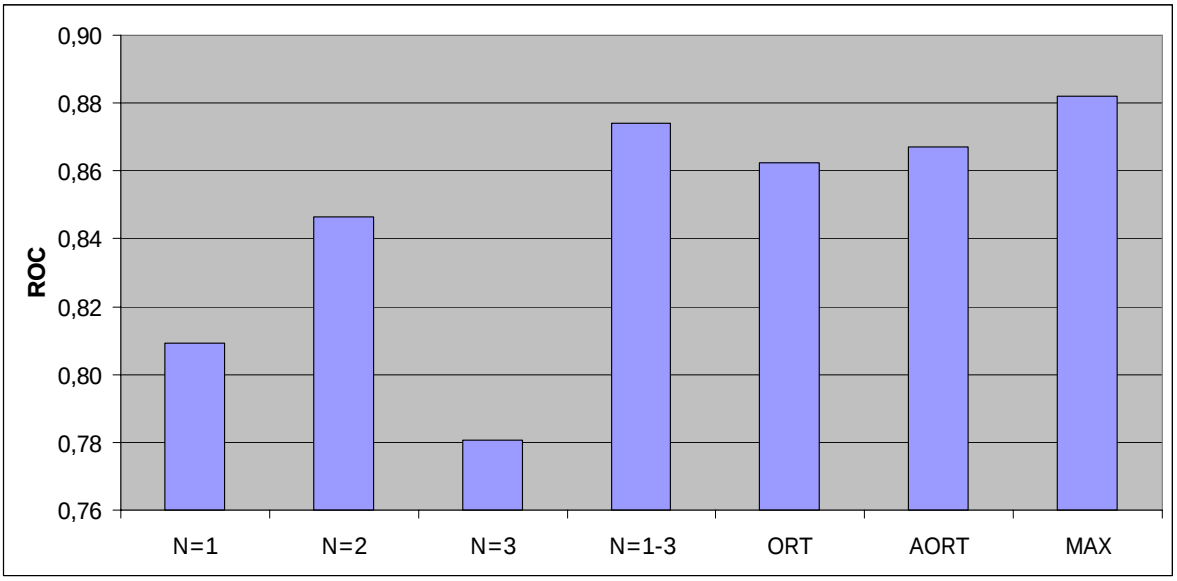
Şekil 6.16 – ORT'a karşı AORT(a), MAX(b)

Şekil 6.16 ve 6.17'de gösterilen ikili karşılaştırmalar ise sadece Şekil 5.4.2'de verilen ve başka bir değişle bu tezde önerilen modelin kullanıldığı yöntemlerin n-peptit birleşimlere göre yapılan karşılaştırmalarıdır. Grafikler incelendiğinde ORT ve AORT'un birine yakın performanslara sahip olduğu görülmektedir. Bakınız, Şekil 6.16(a).



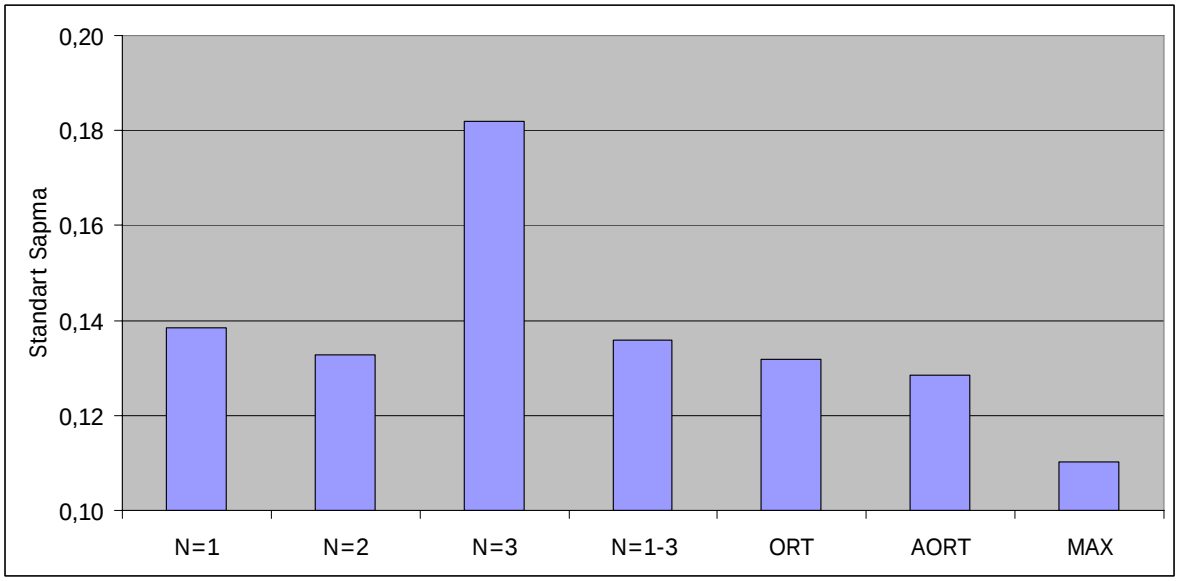
Şekil 6.17 – AORT'a karşı MAX

MAX metodunu aynı modeli kullanan diğer iki yöntemle karşılaştırdığımızda ise görüyoruz ki MAX için belirlenen bölgeye düşen aile sayısı, diğer iki metodun bölgelerine düşen aile sayılarına göre daha fazladır. İkili karşılaştırmalar ile incelenen grafiklerde en iyi sonuçlar her zaman için bu tezde önerilen modeli kullanan MAX metodu ile elde edilmektedir.



Şekil 6.18 – Yöntemlerin ROC Performanslarına Göre Karşılaştırması

Bu tezde önerilen ve uygulanan yöntemlerin, aminoasitlerin n-kompozisyonlarına göre çıkarılan nitelik vektörlerine göre yapılan karşılaştırmalar sonucu ortaya çıkan yöntemlerin genel ROC performansı grafiği, Şekil 6.18'de verilmiştir. Grafiğe göre MAX metodu, en yüksek performansa sahip yöntemdir, 3-peptit metodunun da en düşük performansa sahip yöntem olduğu gösterilmiştir.

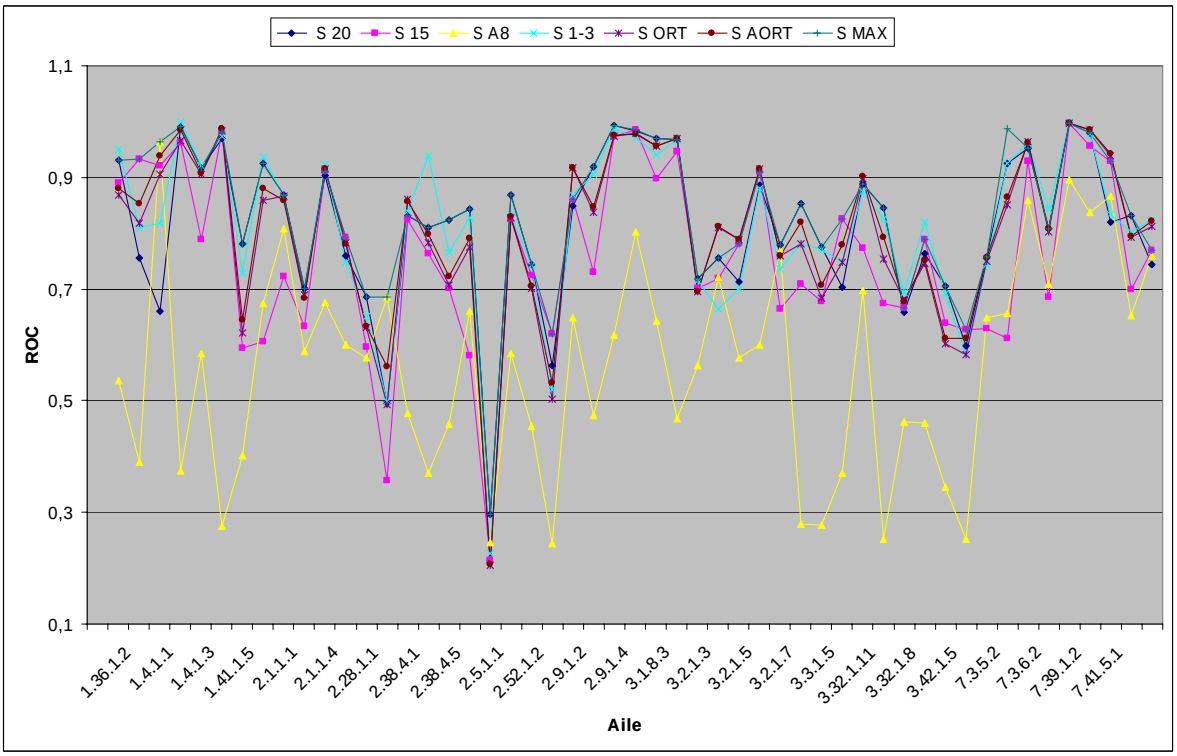


Şekil 6.19 – Yöntemlerin Standart Sapmalarına Göre Karşılaştırılması

Yöntemlerin aileler için verdiği ROC performanslarının standart sapmaları ve en kötü puanları incelendiğinden MAX metodunun, bu tezin kapsamında, n-peptit birleşimlerde, önerilen en iyi yöntem olduğu yinelenir. Şekil 6.19'da gösterilen grafikte standart sapması en düşük olan MAX metodu aynı zamanda en güvenilir yöntem olarak nitelendirilmiştir. Bunun diğer bir anlamı MAX metodunu diğer yöntemlere göre, değişen aile karakteristiklere karşın, oluşabilecek hatalara daha dayanıklıdır.

Küçültülmüş aminoasit alfabeleri üzerinde yapılan deneyler sonucu çıkan değerlerin, analizleri ve yorumları, çıkarılan grafikler ışığında yapılmıştır. Her aileye ait ROC performanslarını daha rahat görebilmek açısından Şekil 6.20'de ailelerin ROC performanslarına ait olan grafik verilmiştir. Burada n-peptit kompozisyonlar için kullanılan yöntemlerin sonuçları çizgi grafik üzerinde gösterilmiştir. Aileler

üzerinde tüm yöntemlerin sergilediği davranışlar rahat bir şekilde görünmektedir. Grafikte de görüldüğü üzere aminoasit alfabesi 8 elemanlı olarak küçültüldüğünde ortaya çıkan performans çok düşüktür. Genelde yöntemler benzer bir desen üzerinde hareket etmektedir. Ancak istisnalar kolayca görülmektedir.



Şekil 6.20 – Ailelerin ROC Performansları (aminoasit gruplama)

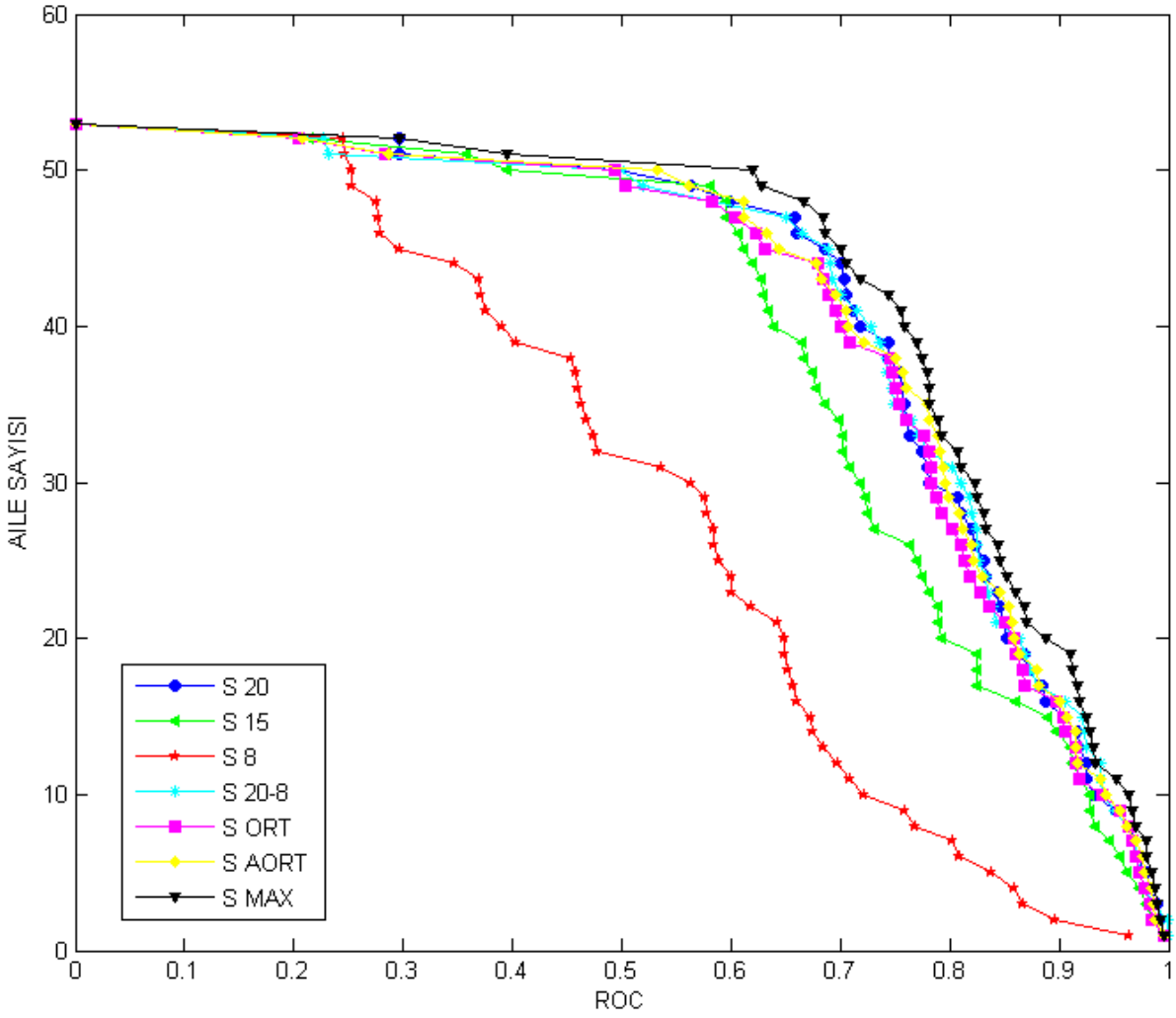
Homoloji tespiti yöntemlerinin birbirlerine göre istatistiksel anlamlarına bakabilmek için yöntemler üzerinde önceki kısımda anlatıldığı gibi ikişerli olarak t-testleri yapılmıştır ve test sonuçları Çizelge 6.4'te verilmiştir. Testlerin yapılış şekilleri ve şartları, öncekiyle aynıdır. Çizelge 6.4 incelendiğinde N=1' göre istatistiksel olarak en anlamlı yöntem N=2'dir daha sonra da MAX metodudur. Ne var ki yöntemlerin performansları incelendiğinde N=2'nin ortalama ROC performansı N=1'den yüksektir, MAX için hesaplanan ROC performansı ise N=1'den de yüksektir. Kullanılan yöntemler ve kullanım şekilleri önceki çalışmaya kıyasla daha farklı olduğu için çıkan sonuçlarında kuş bakışı seyrinin tamamen farklı olması yadsınamaz bir gerçekliktir. Yöntemlerin ikili t-testleri arasında her ne kadar negatif sonuçta çıksa, MAX için tüm yöntemlerin karşılaştırmaları her zaman için anlamlı

çıkmıştır. Çizelge 6.4'te karşılaştırmalar için MAX sütununda görülen değerler hep 0.05 değerinden çok daha düşüktür.

Çizelge 6.4 – Homoloji Tespiti için Kullanılan Yöntemlerin T-testi Puanları (aminoasit gruplama)

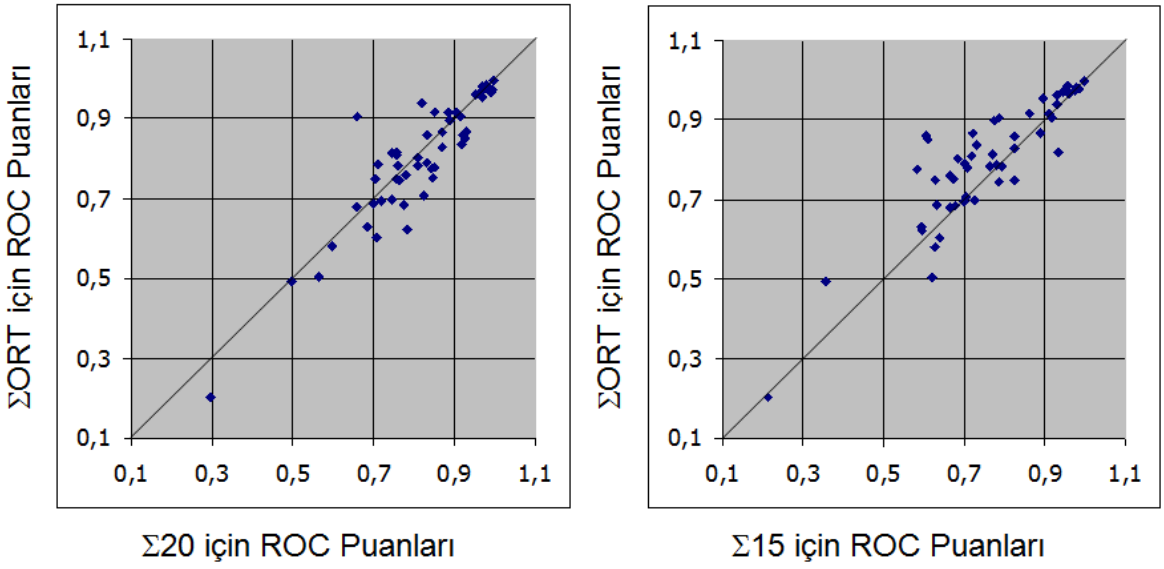
	Σ20	Σ15	Σ8	Σ20–8	ΣORT	ΣAORT	ΣMAX
Σ20		1,79E-03	6,78E-12	9,29E-01	1,83E-01	8,07E-01	3,86E-03
Σ15			1,61E-08	5,96E-04	7,80E-04	4,58E-05	4,77E-07
Σ8				4,68E-12	4,97E-13	1,72E-13	1,24E-14
Σ20–8					1,23E-01	8,17E-01	1,00E-03
ΣORT						1,24E-05	9,78E-06
ΣAORT							6,91E-05
ΣMAX							

Homoloji tespiti için aminoasit alfabelerini küçültme esas alınarak, SCOP veri kümesindeki ailelere uygulanan yedi yöntem, Şekil 6.21'de ROC – aile sayısı grafiği verilmiştir. Eğrilere bakıldığında görülmektedir ki, en üstteki eğri MAX eğrisidir, en altta bulunan eğri ise N=3 eğrisidir. Sonuç olarak şekildeki grafikten MAX metodu diğer yöntemlerden daha iyi çalışmaktadır, çıkarımı kolayca yapılabilmektedir. Bu grafikten varılan sonuç şudur ki, alfabe küçültüldükçe hem aile bazında, hem de genel ROC performanslarında düşüş yaşanmaktadır. Alfabe 20 elemandan 15 elemana indirildiğinde Σ15'e ait çizgi Σ20'ninkine göre biraz daha aşağı inmiştir. Σ15'ten Σ8'e indirildiğinde ise grafiklerdeki düşüş çok bellidir.



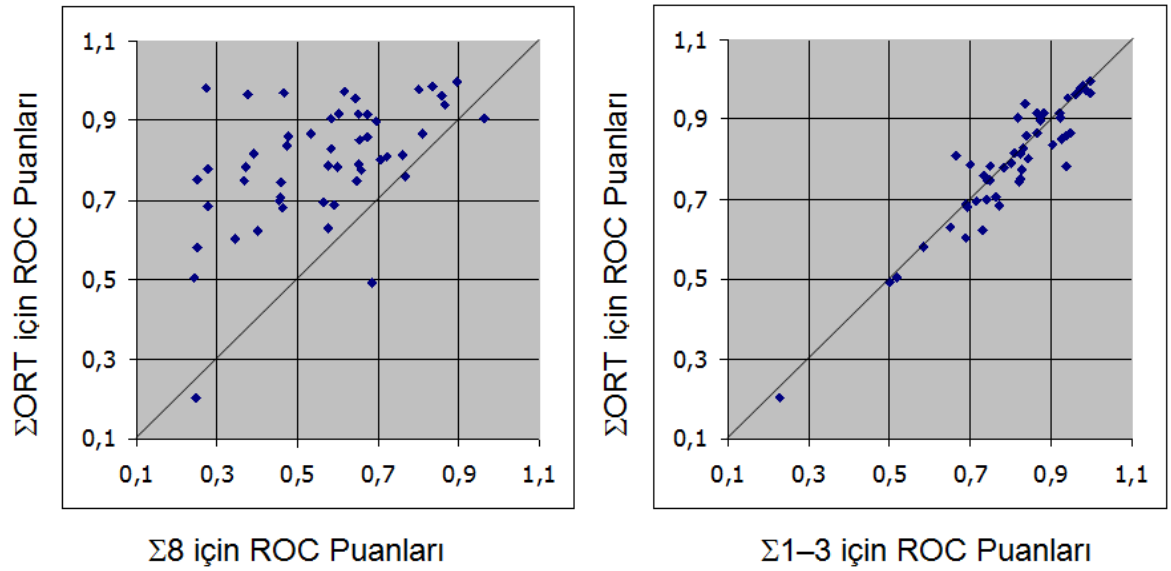
Şekil 6.21 – Homoloji Tespiti Yöntemlerinin Göreceli Performansları (aminoasit gruplama)

Alfabe küçültme üzerinde yapılan ikili karşılaştırma grafikleri kullanılarak yöntemlere göre aileden aileye de karşılaştırılmıştır. Grafiklerde x ve y eksenleri kullanılan yöntemlere ait ROC performanslarını göstermektedir. SCOP veri kümesindeki aileler, grafiklerde mavi noktalar ile ifade edilmektedir. Grafiklerin tam ortasına çizilen çizgi karşılaştırılan iki yöntem arasındaki sınırı vermektedir.



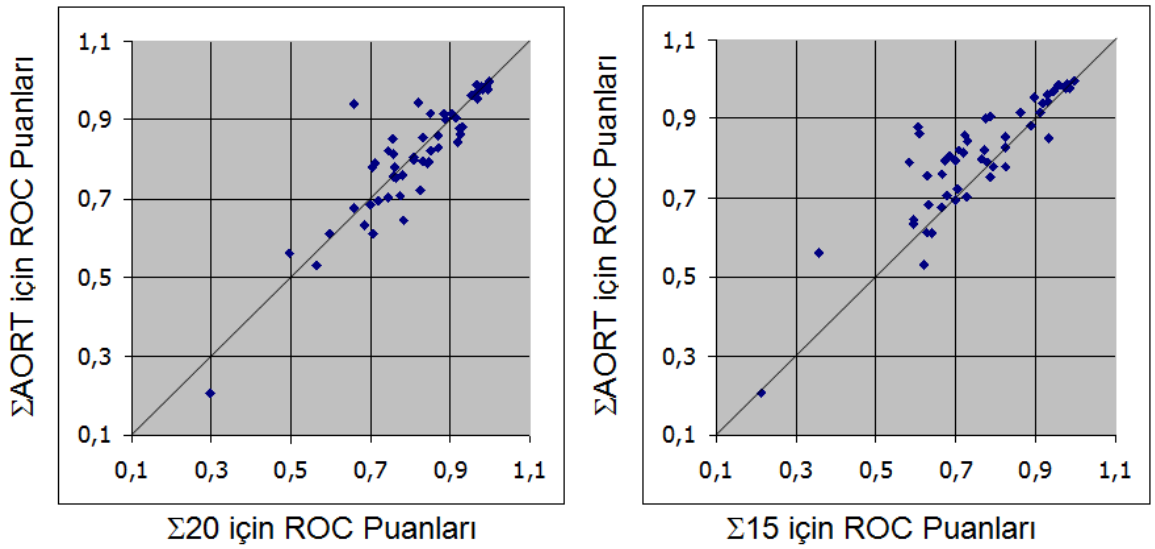
Şekil 6.21 – ΣORT'a karşı Σ20(a), Σ15(b)

Şekil 6.21(b) ve 6.22(a)'a bakacak olursak Şekil 5.4.2'de verilen modeli kullanan ΣORT metodu, Şekil 5.4.1'de verilen modeli kullanan Σ15 ve Σ8 yöntemlerine göre daha yüksek performansa sahiptir. Özellikle metodun Σ8 metodu ile ikili karşılaştırılmasının yapıldığı Şekil 6.22(a) incelenirse, veri kümesindeki 51 aileden 47'si üzerinde ΣORT, 4'ü üzerinde ise Σ8 metodunun performansı yüksektir.

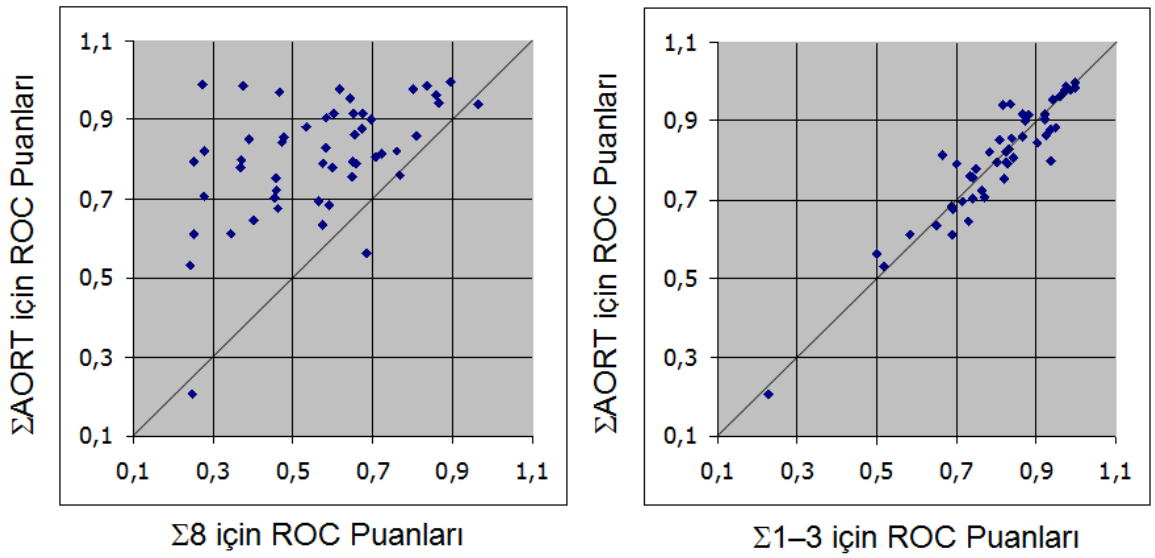


Şekil 6.22 – ΣORT'a karşı Σ8(a), Σ1-3(b)

Şekil 6.21(a) ve Şekil 6.22(b) incelendiğinde ise ΣORT metodunun performansa dayalı başarısı karşılaştırıldığı yöntemlerden yani Σ20 ve Σ1–3'den iyi değildir ancak yakındır.

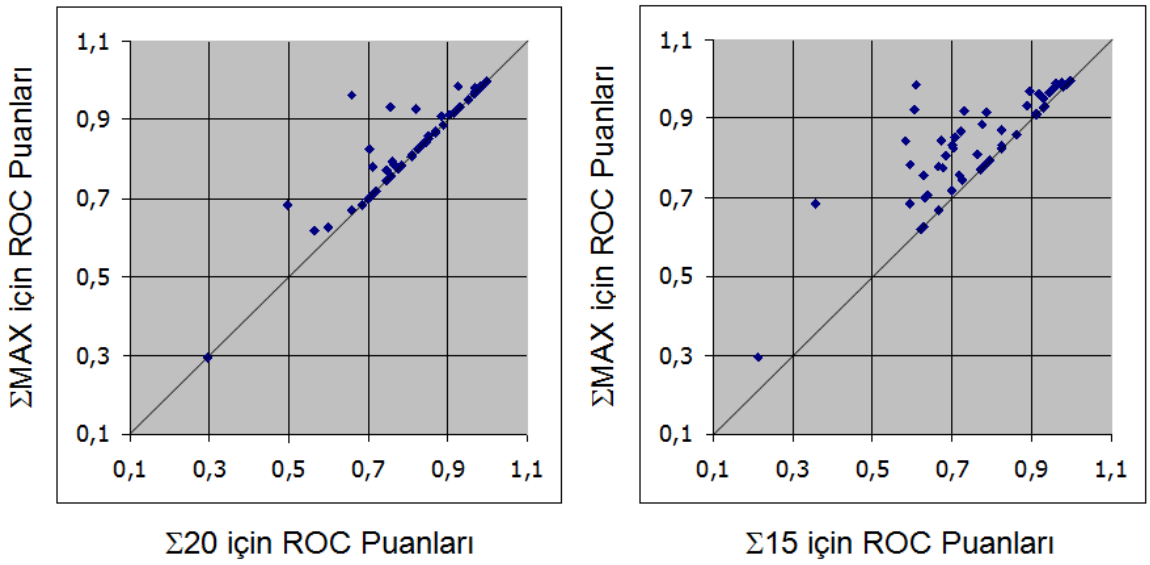


Şekil 6.23 – ΣAORT'a karşı Σ20(a), Σ15(b)



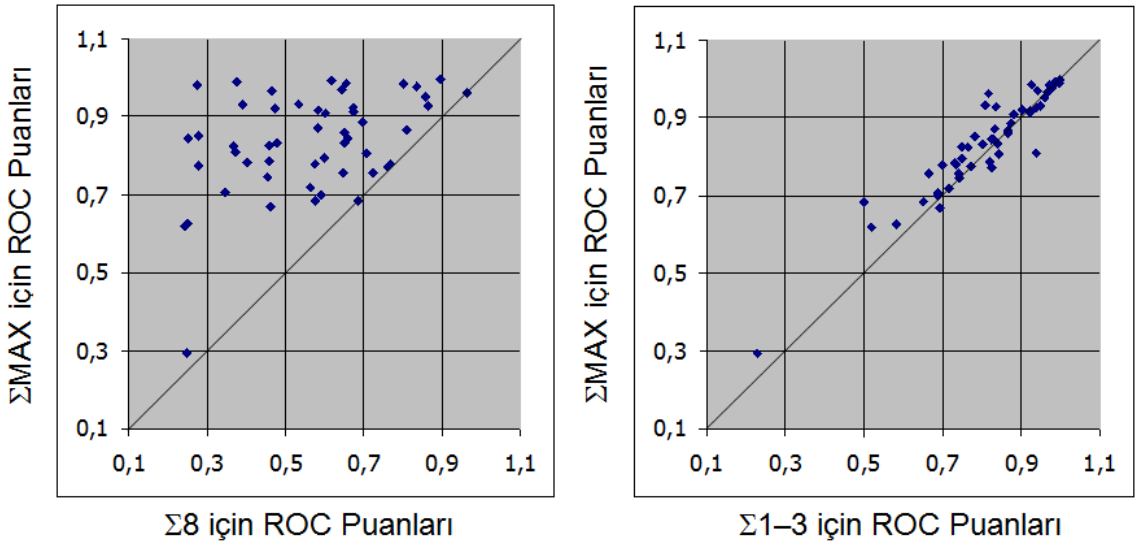
Şekil 6.24 – ΣAORT'a karşı Σ8(a) ve Σ1–3(b)

Temelde sınıflandırıcı puanları üzerinde ağırlıklı ortalama esasına dayanan ΣAORT metodu ile Şekil 6.23(a)'da karşılaştırıldığı Σ20'nin ve Şekil 6.24(b)'de karşılaştırıldığı Σ1–3'ün başarı oranları birbirleriyle aynıdır denilebilir, grafiklerde de bu durum görülmektedir. Yöntemlerin birbirleri üzerinde baskınlığı yoktur, aileler için verilen genel performans ortalamaları aynı seviyelerdedir. Şekil 6.23(b) ve Şekil 6.24(a)'da karşılaştırılan yöntemler arasında üstünlük ΣAORT metodundadır. Veri kümesinde bulunan ailelerden, daha fazlasında, daha iyi sonuçlara sahip yöntemdir.



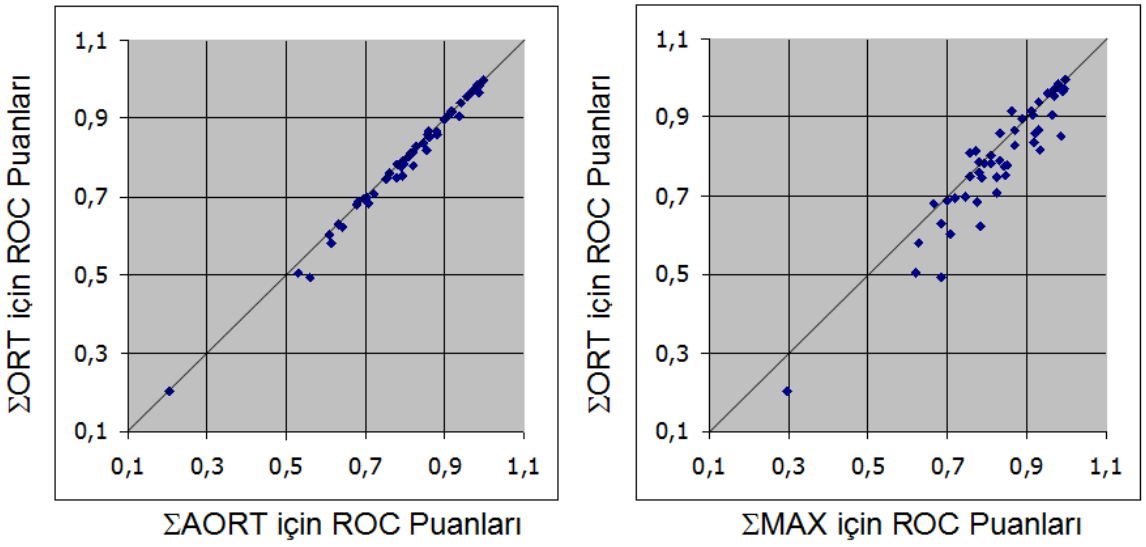
Şekil 6.25 – ΣMAX'a karşı Σ20 ve Σ15

ΣMAX metodu ile yapılan ikili karşılaştırmalar, Şekil 6.25 ve Şekil 6.26 verilmiştir. Grafikler incelendiğinde, verilen grafiklerde uygulanan yöntemler arasındaki en başarılı yöntem olarak kendini ispatlar. Şekil 6.25(a), Şekil 6.25(b) ve Şekil 6.26(a), ΣMAX metodu ile Σ20, Σ15 ve Σ8 yöntemlerinin aileler üzerindeki karşılaştırmalarıdır. Şekil 6.26(b)'de verilen ΣMAX ile Σ1–3'ün ikili karşılaştırmasında, her iki bölgede de ailelerin yer aldığı görülmektedir. Ancak çoğunluk daha yüksek performanslarla beraber yine ΣMAX metodunun sınırları içerisindedir. İki şekilde incelenen toplam dört grafikte ΣMAX metodu her zaman için en iyi sonuçları vermiştir.



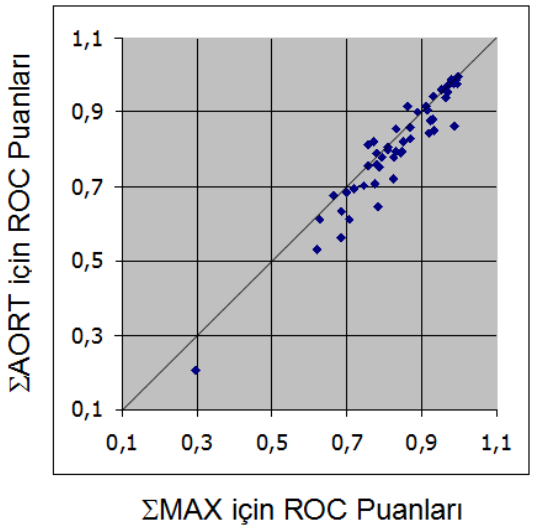
Şekil 6.26 – ΣMAX'a karşı Σ8(a), Σ1–3(b)

Bu tezde önerilen modeli kullanan üç metodun ikili karşılaştırmaları Şekil 6.27 ve Şekil 6.28'de verilmiştir. ΣORT ve ΣAORT birbirleri üzerinde baskın önceliliklere sahip değildir ancak Şekil 6.27(a)'da gösterildiği gibi ΣAORT, ΣORT metoduna göre çok az da olsa daha iyi performansa sahiptir.

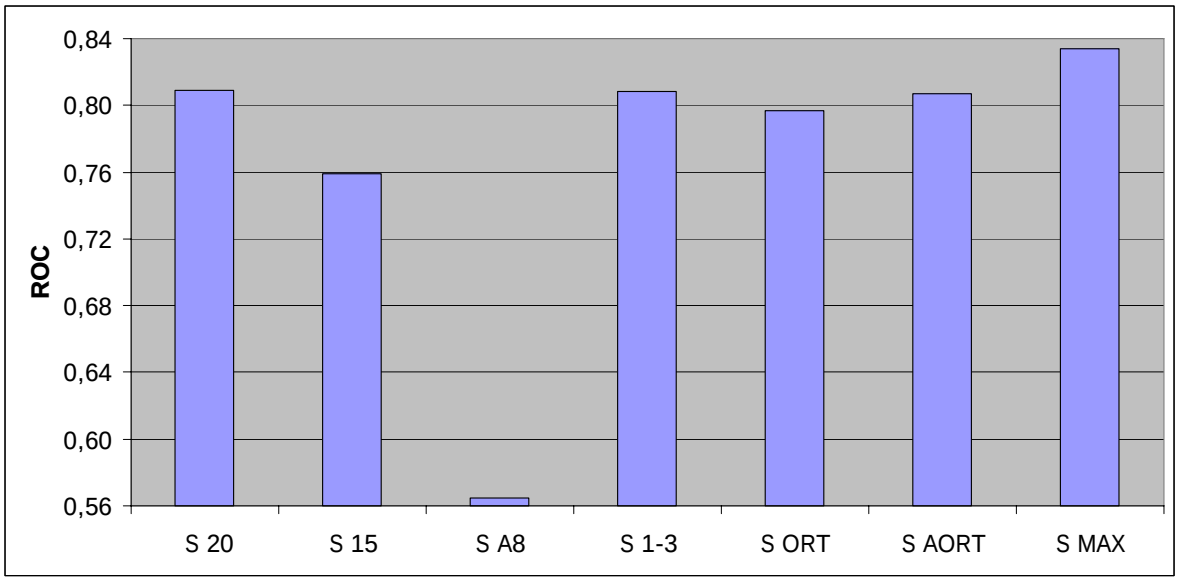


Şekil 6.27 – ΣORT'a karşı ΣAORT(a), ΣMAX(b)

ΣMAX ile diğer iki metodun karşılaştırılmasında, sonuç değişmemektedir. ΣMAX en iyi performansa sahip olan yöntem olarak kendini öne çıkarmaktadır, Şekil 6.26(a) ve Şekil 6.27'de görülmektedir.



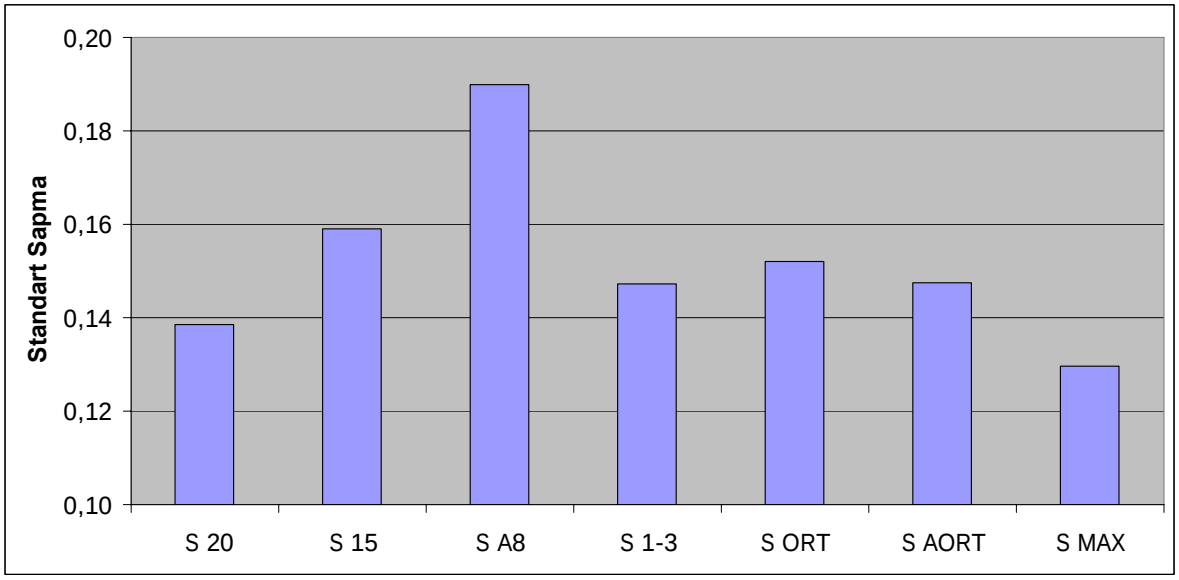
Şekil 6.28 – ΣAORT'a karşı ΣMAX



Şekil 6.29 – Yöntemlerin ROC Performanslarına Göre Karşılaştırması

Tezde önerilen ve uygulanan yöntemlerin, aminoasitlerin alfabelerinin küçültülmesine göre çıkarılan nitelik vektörlerine göre yapılan karşılaştırmalar

sonucu ortaya çıkan yöntemlerin genel ROC performansı grafiği, Şekil 6.29'da verilmiştir. Grafiği yorumlayacak olursak homoloji tespiti için kullanılan yedi yöntemden ΣMAX metodu, en yüksek performansa sahip yöntemdir, Σ3 metodunun da en düşük performansa sahip yöntem olduğu konusunda tereddüt yoktur.



Şekil 6.30 – Yöntemlerin Standart Sapmalarına Göre Karşılaştırması

Yöntemlerin SCOP veri kümesindeki aileler için verdiği ROC performanslarının standart sapmaları ve en kötü puanları incelendiğinden ΣMAX metodunun, bu tezin kapsamında, aminoasit alfabeleri küçültüğünde, önerilen en iyi yöntem olduğu yinelenir. Şekil 6.19'da gösterilen grafikte standart sapması en düşük olan ΣMAX metodu aynı zamanda en güvenilir yöntem olarak nitelendirilmiştir. Bunun diğer bir anlamı ΣMAX metodunu diğer yöntemlere göre, değişen aile karakteristiklere karşın, oluşabilecek hatalara daha dayanıklıdır.

6.2. Sonuçlar

Sonuçları özetleyecek olursak, tez kapsamında, n-peptit birleşimler için 7, aminoasit alfabesindeki küçültmeler için 7, toplam 14 yöntem uygulanmıştır, sonuçları alınmış ve yorumlanmıştır. Bu yöntemler için toplamda 2 model

kullanılmıştır. Deneyler SCOP veri kümesi üzerinde oluşturulan 51 aile üzerinde gerçekleştirilmiştir. Tez kapsamında önerilen 1 üst sınıflandırma yaklaşımı ve bu yaklaşımı kullanan 6 yöntem mevcuttur.

N-peptit birleşimler için sonuçlar incelendiğinde. En iyi puanı, öğrenme kümesinde en başarılı olanı seçme (MAX) vermiştir. N-peptit birleşimler için yöntemlerin standart sapma değerleri incelendiğinde yine en tutarlı yöntem MAX olmuştur. Yöntemlerin istatistiksel olarak anlamlı olup olmadığına bakabilmek için, tüm yöntemler arasında öğrenci t-testleri yapılmıştır. Çıkan sonuçlar 0.05 olarak belirlenen p-değerinden ne kadar küçük çıkarsa, yöntemler birbirinden o kadar farklıdır. Başka bir deyişle, çözüme doğru giderken farklı yollar izlemişlerdir. Sonuçlar göstermektedir ki, bu tezde önerilen ve en iyi sonuçları veren MAX yöntemi ile N=1, N=2 ve N=3 arasındaki ikili karşılaştırmalara bakıldığında istatistiksel olarak anlamlıdır. Ama ne var ki, bu sonuç N=1-3 için geçerli değildir. Çünkü sonuçlardan yola çıkarak söylenebilir ki N=1-3 tek bir sınıflandırıcı kullanan üstsel bir yaklaşımdır. N=1-3 ve önerilen üst sınıflandırma yaklaşımını kullanan diğer yöntemler çözüme aynı yoldan ilerlemektedirler. Ancak eğitim kümesinde en iyi olanı seç yöntemi, bu yolda iyileştirme kaydetmiştir.

Aminoasit alfabesi küçültüldüğünde ortaya çıkan sonuçlar incelendiğinde. Yine en iyi puanı, öğrenme kümesinde en başarılı olanı seçme (ΣMAX) vermiştir. Standart sapma değerleri incelendiğinde yine en tutarlı yöntem ΣMAX olmuştur. Aminoasit alfabesi küçültüldüğünde, en başarılı yöntem ΣMAX olmuştur. İstatistiksel olarak anlamlı mı diye incelendiğinde de çözüm yolunun tüm diğer yöntemlerden tamamen farklı olduğu gözlemlenmiştir.

Nitelik girdileri 1-peptit, 2-peptit, 3-peptit gibi gruplara ayrılıp sınıflandırıcılara parça parça verildikleri için birim zamanda belleğe yüklenen ve bellekte tutulan veride, tek bir sınıflandırıcının kullanıldığı duruma kıyasla azalma olmaktadır. Bu işlem gerek eğitim gerekse test aşamalarında gerçekleştirilir. Önerilen bu üst sınıflandırma yaklaşımının, tüm veriyi değil de sadece bir parça veriyi belleğe yüklemesi, yüklenen parça için sınıflandırıcıdan sonucu aldıktan sonra diğer parçalara geçmesi etkili bellek kullanımını gerçekleştirmiş olur. Parça parça tüm veriler

işlendikten sonra, ortaya çıkarılan sonuçlar birleştirilir ve sistem işlevini yerine getirmiş olur.

Bu çalışma uzak homoloji tespiti ile ilgili sonuçların yanında, DVM tabanlı sınıflandırma problemlerinde hangi üst sınıflandırma yöntemlerinin daha başarılı olacağı hakkında da fikir vermektedir. Başka sınıflandırma problemlerinde de, bu tezde önerilen yöntemlerin denenmesi uygun olacaktır.

7 – KAYNAKLAR

[1] Altschul, S., Gish, W., Miller, W., Myers, E. W., Lipman, D., (1990). A basic local alignment search tool, Journal of Molecular Biology, 251, 403-410.

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

[3] Anfinsen, C. B., (1973). Studies on the principles that govern the folding of protein chains. Science, 181, 223-230.

[4] Bahar, I., Atilgan, A. R., Jernigan, R. J., Erman, B., (1997). Understanding the recognition of protein structural classes by amino asid composition, Proteins 29, 172-185.

[5] Bairoch, A. ve Apweiler, R., (1999). The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999, Nucleic Acids Research, 27, 49-54.

[6] Bartell, B., Cottrell, G. ve Belew, R., (1994). Automatic combination of multiple ranked retrieval systems, In Proceedings of the 17th ACM SIGIR Conference on Research and Development in Information Retrieval, 173-181.

[7] Bejerano, G., Yona, G., (2000). Variations on probabilistic suffix trees: statistical modeling and prediction of protein families, Bioinformatics, 17, 23-43.

[8] Belkin, N., Cool, C., Croft, W. ve Callan, J., (1993). The effect of multiple query representations on information retrieval system performance, In Proceedings of the 16th ACM SIGIR Conference on Research and Development in Informaion Retrieval, 339-346.

[9] Belkin, N., Kantor, P., Fox, E. ve Shaw, J., (1995). Combining the evidence of multiple query representations for information retrieval, Information Processing and Management, 31(3), 431-448.

[10] Ben-hur, A., Brutlag, D., (2003). Remote homology detection: a motif based approach, Bioinformatics, 19, 26-33.

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

[12] Bhasin, M. ve Raghava, G. P. S., (2004). ESLpred: SVM-based method for subcellular localization of eukaryotic proteins using dipeptide composition and PSI-BLAST, Nucleic Acids Research, 32, 415-419.

[13] Bieganski, R., Riedl, J., Carlis, J. V., Retzael, E. R., (1994). Generalized suffix trees for biological sequence data: applications and implementations, Proceeding of the 27th Hawaii International Conference on Systems and Sciences, 35-44.

[14] Bystroff C., Baker, D., (1998). Prediction of local structure in proteins using a library of sequence-structure motifs, Journals of Molecular Biology, 281, 565-577.

[15] Cai, C. Z., Wang, W. L., Sun, L. Z., Chen, Y. Z., (2003). Protein function classification via support vector machines, Mathematical Biosciences, 185, 111-122.

[16] Callan, J., Croft, W. ve Broglio, J., (1995a). TREC and TIPSTER experiments with INQUERY, Information Processing and Management, 31(3), 327-343.

[17] Callan, J., Lu, Z. ve Croft, W., (1995b). Searching distributed collections with inference networks, In Proceedings of the 18th ACM SIGIR Conference on Research and Development in Information Retrieval, 21-28

[18] Carillo, H. ve Lipman, D. J., (1988). The multiple alignment problem in biology, SIAM Journal of Applied Mathematics, 48, 1073-1082.

[19] Christianni, N. ve Shawe-Taylor, J., (2000). An Introduction to Support Vector Machines and other kernel-based learning methods, Cambridge University Press.

[20] Croft, W. ve Harper, D., (1979). Using probabilistic models of document retrieval without relevance information, Journal of Documentation, 35, 285-295.

[21] Croft, W. ve Thompson, R., (1984). The use of adaptive mechanisms for selection of search strategies in document retrieval systems, In Proceedings of the 7th ACM SIGIR Conference on Research and Development in Information Retrieval, 95-110, Cambridge University Press.

[22] Croft, W. ve Thompson, R., (1987). I3R: A new approach to the design of document retrieval systems, Journal of the American Society for Information Science, 38(6), 389-404.

[23] Croft, W., Krovetz, R. ve Turtle, H., (1990). Interactive retrieval of complex documents, Information Processing and Management, 26(5), 593-613.

[24] Croft, W., Lucia, T. J., Cringean, J. ve Willett, P., (1989). Retrieving documents by plausible inference: An experimental study, Information Processing and Management, 25(b), 599-614.

[25] Crounch, C., Crounch, D. ve Nareddy, K. (1990). The automatic generation of extended queries, In Proceedings of the 13th ACM SIGIR Conference on Research and Development in Information Retrieval, 369-383.

[26] Dayhoff, M. O., Schwartz, R. M., Orcutt, B. C., (1978). A model of evolutionary change in proteins, Atlas of Protein Sequence and Structure, 5, 345-352.

[27] Deerwester, S., Dumais, S., Furnas, G., Landauer, T. ve Harsman, R., (1990). Indexing by latent semantic analysis, Journal of the American Soceity for Information Science, 41, 391-407.

[28] Delcher, A., Kasif, S., Fleishmann, R., Peterson, J., White, O., Salzberg, S., (1999). Alignment of whole genomes, Nucleic Acids Research, 27, 2369-2376.

[29] Ding, C. ve Dubchack, I., (2001). Multi-class protein fold recognition using support vector machines and neural networks, Bioinformatics, 17, 349-358.

[30] Eddy, S. R., (1998). Profile hidden markov models, Bioinformatics, 14, 755-763.

[31] Fagin, R., (1996). Combining fuzzy information from multiple systems, In Proceedings of the 15th ACM Conference on Principles of Database Systems (PODS), 216-226.

[32] Fagin, R., (1998). Fuzzy queries in multimedia database systems, In Proceedings of the 17th ACM Conference on Principles of Database Systems (PODS), 1-10.

[33] Falquet, L., Pagni, M., Bucher, P., Hulo, N., Sigrist, C., Hofmann, K., Bairoch, A., (2002). The PROSITE database, its status in 2002, Nucleic Acid Research, 30,235-238.

[34] Fox, E. ve France, R., (1987). Architecture of an expert system for composite document analysis, representation, and retrieval, Journal of Approximate Reasoning, 1, 151-175.

[35] Fox, E. ve Shaw, J., (1994). Combination of multiple searches, In Proceedings of the 2nd Text Retrieval Conference (TREC-2), 243-252, National Institude of Standarts and Technology Special Publication, 500-215.

[36] Fox, E., Nunn, G. ve Lee, W., (1988). Coefficients for combining concept classes in a collection, In Proceedings of the 11th ACM SIGIR Conference on Research and Development in Information Retrieval, 291-308.

[37] Fuhr, N. ve Buckley, C., (1991). A probabilistic learning approach for document indexing, ACM Transactions on Information Systems, 9(3), 223-248.

[38] Fuhr, N., (1992). Probabilistic models in information retrieval, Computer Journal, 35(3), 243-255.

[39] Gey, F., (1994). Inferring probability of relevance using the method of logistic regression, In Proceedings of the 17th ACM SIGIR Conference on Research and Development in Information Retrieval, 222-231.

[40] Greiff, W., (1988). A theory of term weighting based on exploratory data analysis, In Proceedings of the 21th ACM SIGIR Conference on Research and Development in Information Retrieval, 11-19.

[41] Greiff, W., Croft, W. ve Turtle, H., (1997). Computationally tractable probabilistic modeling of Boolean operators, In Proceedings of the 20th ACM SIGIR Conference on Research and Development in Information Retrieval, 119-128.

[42] Gribskov, M. ve Robinson, N. L., (1996). Use of receiver operating characteristic analysis to evaluate sequence matching, Computer and Chemistry, 20, 25-33.

[43] Gribskov, M., Lüthy, R., Eisenberg, D., (1990). Profile Analysis, Methods in Enzymology, 183, 146-159.

[44] Grundy, W. N., (1998). Homology Detection via Family Pairwise Search, Journal of Computational Biology, 5, 479-492.

[45] Gusfield, D., (1997). Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology, Cambridge University Press, New York.

[46] Harman, D., (1992). The DARPA TIPSTER project, ACM SIGIR Forum, 26(2), 26-28.

[47] Harman, D., (1995). Overview of the second text retrieval conference (TREC-2), Information Processing and Management, 31(3), 271-289.

[48] Hastie, T., Tibshirani, R., Friedman, J., (2001). The elements of statistical learning: data mining, inference and prediction, Springer.

[49] Henikoff, S., Henikoff, J. G., (1992). Amino acid substituion matrices from protein blocks, Proc. Natl. Acad. Sci. USA, 89, 10915-10919.

[50] Henikoff, S., Henikoff, J. G., Pietrokovski, S., (1999). BLOCKS: a nonredundant database of protein alignment blocks derived form multiple combinations, Bioinformatics, 15, 471-479.

[51] Higgins, D. G., Thompson, J. D., Gibson, T. J., (1994). CLUSTAL W: Improving the sensitivity of progressive multiple alignment through sequence weighting, position specific gap penalties and weight matrix choice, Nucleic Acids Research, 22, 4673-4680.

[52] Hischberg, D. S., (1997). Algorithms for the longest common subsequence problem, Journal of ACM, 24, 664-675.

[53] Hou, Y., Hsu, W., Lee, M. L., Bystroff, C., (2003). Efficient remote homology detection using local structure, Bioinformatics, 19, 2294-2301.

[54] Huang, J. ve Brutlag, D., (2001). The emotif database, Nucleic Acids Research, 29, 202-204.

[55] Hull, D., Pedersen, J. ve Schutze, H., (1996). Method combination for document filtering, In Proceedings of the 19th ACM SIGIR Conference on Research and Development in Information Retrieval, 279-287.

[56] Internet: Homology, 2006, Wikipedia the free encyclopedia. http://en.wikipedia.org/wiki/Homology_%28biology%29

[57] Internet: Protein, 2006, Wikipedia the free encyclopedia. http://en.wikipedia.org/wiki/Protein

[58] Internet: Support Vector Machine, 2006, Wikipedia the free encyclopedia. http://en.wikipedia.org/wiki/Data_fusion

[59] Internet: Support Vector Machine, 2006, Wikipedia the free encyclopedia. http://en.wikipedia.org/wiki/Support_vector_machine

[60] Jaakola, T., Diekhans, M., Haussler, D., (2000). A discriminative framework for detecting remote homologies, Journal of Computational Biology, 7, 95-114.

[61] Karplus, K., Barrett, C., Hughey, R., (1998). Hidden Markov Models for detecting remote protein homologies, Bioinformatics, 14, 846-856.

[62] Kim, J., Pramanik, S., Chung, M. J., (1994). Multiple Sequence Alignment using Simulated Annealing, Computer Applications in Biosciences, 10, 419-426.

[63] Krogh, A., Brown, M., Mian, I. S., Sjölander, K., Haussler, D., (1994). Hidden markov models in computational biology: application to protein modeling, Journal of Molecular Biology, 235, 1501-1531.

[64] Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., Wootton, J. C., (1993). Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment, Science, 262, 208-214.

[65] Lee, J., (1995). Combining multiple evidence from different properties of weighting schemes, In Proceedings of the 19th ACM SIGIR Conference on Research and Development in Information Retrieval, 180-188.

[66] Lee, J., (1997). Analyses of multiple evidence combination, In Proceedings of the 20th ACM SIGIR Conference on Research and Development in Information Retrieval, 267-276.

[67] Liao, L. ve Noble, W. S., (2003). Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships, Journal of Computational Biology, 10, 857-868.

[68] Lipman, D. J. ve Pearson, W. R., (1985). Rapid and sensitive protein similarity search, Science, 227, 1435-1441.

[69] Löytynoja, A., Milinkovitch, M. C., (2003). A Hidden Markov Model for progressive multiple alignment, Bioinformatics, 19, 1505-1513.

[70] McGill, M., Koll, M. ve Noreault, T., (1979). An evaluation of factors affecting document ranking by information retrieval systems, Final report for grant NSF-IST-78-19454 to the National Science Foundation, Syracuse University.

[71] Mitra, M., Singhal, A. ve Buckley, C., (1998). Improving automatic query expansion, In Proceedings of the 21th ACM SIGIR Conference on Research and Development in Information Retrieval, 206-214.

[72] Murphy, L. R., Wallqvist, A., Levy, R. M., (2000). Simplified amino acid alphabets for protein fold recognition and implications for folding, Protein Engineering, 13, 149-152.

[73] Murzin, A. G., Brenner, S. E., Hubbard, T., Chothia, C., (1995). SCOP: A structural classification of proteins database for the investigation of sequences and structures. Journal of Molecular Biology, 247, 536-40.

[74] Myers, E. W. ve Miller, W., (1988). Optimal alignments in linear space, Computer Applications in Biosciences, 4, 11-17.

[75] Myers, E. W., (1991). An overview of sequence comparison algorithms in molecular biology, Technical Report, TR 91-29, University of Arizona.

[76] Needleman, S. B., Wunch, C. D., (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins, Journal of Molecular Biology, 48, 443-453.

[77] Notredame, C., Higgins, D. G., (1996). SAGA: Sequence Alignment by Genetic Algorithms, Nucleic Acids Research, 24, 1515-1524.

[78] Oğul, H., Mumcuoğlu, E., (2005). Discriminative remote homology detection using maximal unique sequence matches, Lecture Notes in Computer Science, 3546, 283-292, Springer-Verlag.

[79] Oğul, H., Mumcuoğlu, E., (2006). SVM-based Detection of Distant Protein Structural Relationships Using Pairwise Probabilistic Suffix Trees, Computational Biology and Chemistry, 30, 292-299, Elsevier.

[80] Oğul, H., Mumcuoğlu, E., (2007). A discriminative method for remote homology detection based on n-peptide compositions with reduced amino acid alphabets, BioSystems, 87, 75-81, Elsevier.

[81] Park, J., Karplus, K., Barrett, C., Hughey, R., Haussler, D., Hubbard, T., Chothia, C., (1998). Sequence comparisons using multiple sequences detect tree times as many remote homologues as pairwise methods, Journal of Molecular Biology, 284, 1201-1210.

[82] Pears, J., (1988). Probabilistic reasoning in intelligent systems: Networks of plausible inference, Morgan Kaufmann, San Mateo.

[83] Rajashekar, T. ve Croft, W., (1995). Combining automatic and manual index representations in probabilistic retrieval, Journal of the American Society for Information Science, 46(4), 272-283.

[84] Robertson, S. ve Sparck Jones, K., (1976). Relevance weighting of search terms, Journal of the American Society for Information Science, 27, 129-146.

[85] Rost, B., (1999). Twilight zone of protein sequence alignments, Protein Engineering, 12, 85-94.

[86] Salton, G. ve McHill, M., (1983). Introduction to modern information retrieval. McGraw-Hill, New York.

[87] Salton, G., Allan, J. ve Buckley, C., (1993). Approaches to passage retrieval in full text information systems, In Proceedings of the 17th ACM SIGIR Conference on Research and Development in Information Retrieval, 49-56.

[88] Saracevic, T. ve Kantor, P., (1988). A study of information seeking and retrieving. Part III. Searchers, searches, overlap, Journal of the American Society for Information Science, 39(3), 197-216.

[89] Smith T. F. ve Waterman, M. S., (1981). Identification of common molecular subsequences, Journal of Molecular Biology, 147, 195-197.

[90] Sparck Jones, K., (1971). Automatic keyword classification for information retrieval, Butterworths, London.

[91] Tumer, K. ve Ghosh, J., (1999). Linear and order statistics combiners for pattern classification, In Sharkey, A., editor, Combining Artificial Neural Networks, 127-162, Springer-Verlag.

[92] Turtle, H. ve Croft, W., (1991). Evaluation of an inference network-based retrieval model, ACM Transactions on Information Systems, 9(3), 187-222.

[93] Turtle, H. ve Croft, W., (1992). A comparison of text retrieval models, Computer Journal, 35(3), 279-290.

[94] Van Rijsbergen, C., (1979). Information Retrieval, Buttherworths, London.

[95] Van Rijsbergen, C., (1986). A non-classical logic for information retrieval, Computer Journal, 29, 481-485.

[96] Vogt, C. ve Cottrell, G., (1998). Predicting the performance of linearly combined IR systems, In Proceedings of the 21th ACM SIGIR Conference on Research and Development in Information Retrieval, 190-196.

[97] Voorhees, E., Gupta, N. ve Johnson-Laird, B., (1995). Learning collection fusion strategies, In Proceedings of the 18th ACM SIGIR Conference on Research and Development in Information Retrieval, 172-179.

[98] Wallqvist, A., Fukunishi, Y., Murphy, L. R., Fadel, A., Levy, R. M., (2000). Iterative sequence/structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases, Bioinformatics, 16, 988-1002.

[99] Xu, J. ve Croft, W., (1996). Query expansion using local and global document analysis, In Proceedings of the 19th ACM SIGIR Conference on Research and Development in Information Retrieval, 4-11.

8 – EKLER

[A] SCOP

SCOP (Structural classification of Proteins) proteinlerin dört seviyeli yapısal sınıflandırmasına sahip hiyerarşik veritabanıdır (Murzin et al., 1995). En üst seviyede proteinin ikincil yapısını hakkında bilgi veren, sınıf (class) mevcuttur. Veritabanında kat (fold), ikinci seviyeye verilen isimdir, bu seviyede proteinin katlı yapısı verilmiştir. Üçüncü seviye süperaile (super family) olarak adlandırılır, düşük dizilim benzerliğine sahip proteinlerin sınıflandırıldığı seviyedir. Bu seviyede verilen dizilimlerin benzerlikleri %30'un altındadır ancak proteinlerin ortak evrimsel kökenden geldiğine dair yapısal veya fonksiyonel niteliklere sahiptir. En alt seviyede aile olarak verilir, aile içindeki sınıflandırılan proteinler belirgin benzerliklere ve ortal evrimsel kökene sahiptirler. SCOP veritabanının düzeni ağaç yapısındadır, en üstte SCOP kökü bulunmaktadır. SCOP veritabanındaki proteinlerin sınıflandırılması manuel olarak biyoloji uzmanları tarafından yapılmıştır. Veritabanı http://scop.berkeley.edu adresinde erişime açık olarak bulunmaktadır. Kinesin proteini için örnek SCOP çıktısı Şekil 8.1'de verilmiştir.

Protein: Kinesin from Human (*Homo sapiens*)

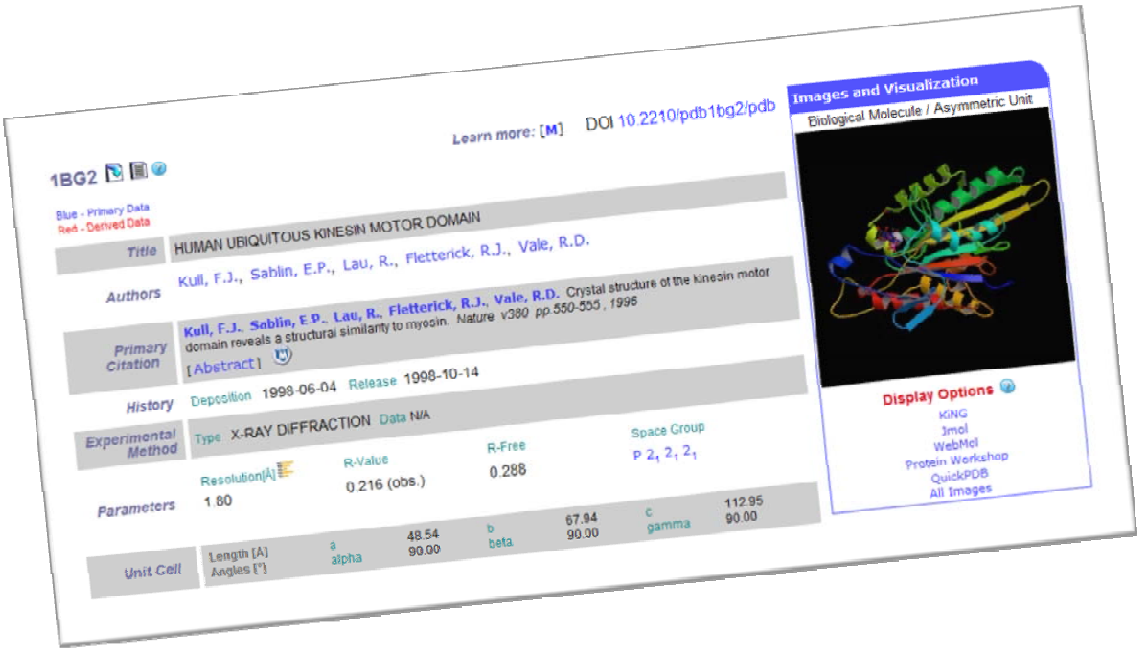
Lineage:

1. Root: scop
2. Class: Alpha and beta proteins (a/b) [51349]
   Mainly parallel beta sheets (beta-alpha-beta units)
3. Fold: P-loop containing nucleoside triphosphate hydrolases [52539]
   3 layers: a/b/a, parallel or mixed beta-sheets of variable sizes
4. Superfamily: P-loop containing nucleoside triphosphate hydrolases [52540]
   division into families based on beta-sheet topologies
5. Family: Motor proteins [52641]
6. Protein: Kinesin [52646]
7. Species: Human (*Homo sapiens*) [52647]

Şekil 8.1 – Örnek SCOP Çıktısı

[B] PDB

Protein veri bankası (Protein Data Bankası) dünyadaki en büyük protein veritabanlarından biridir (Berman et al., 2000). Veritabanı içerisindeki her proteinin PDB ID adında bir birincil anahtarı mevcuttur. Veritabanı porteinler hakkında çok fazla bilgi içermektedir; proteinlerin birincil, üçüncül yapıları, örnek olarak proteinlerin xyz koordinat uzayındaki tüm atomik pozisyonları, organizma kaynağı, yazarların sınıflandırılması, diğer veritabanlarına bağlantılar vb. verilebilir. Veritabanına erişim için web adresi http://www.pdb.org olarak verilmiştir. Veritabanından alınmış örnek bir çıktı Şekil 8.2'de verilmiştir.



Şekil 8.2 – Örnek PDB Çıktısı

9 – ÖZGEÇMİŞ

Curriculum Vitae

Personal Information

First Name	:	AYDIN
Middle Name	:	CAN
Surname	:	POLATKAN
Date of Birth	:	26 August 1982
Place of Birth	:	Ankara
Country of Citizenship	:	Turkish
Marital Status	:	Single
Web	:	http://www.baskent.edu.tr/~polatkan
E-mail	:	aydincan.polatkan@gmail.com
Phone	:	+90 (312) 439 28 69
Mobile Phone	:	+90 (532) 565 59 23

Educational Background

1988 – 1993 Ayranci Primary School
1993 – 1997 TED Ankara College Secondary School
1997 – 2000 TED Ankara College High School
2000 – 2004 Baskent University

Major Field of Study	:	Computer Engineering
Actual Name of Diploma	:	Bachelor of Computer Engineering
Date Received	:	June 2004

2004 – 2006 Baskent University

Major Field of Study	:	Computer Engineering
Actual Name of Diploma	:	Master of Computer Engineering
Date Received	:	February 2007

English Background

Number of years in		
TED Ankara College	:	7
Baskent University	:	7
Total	:	14 years of English education

Occupational Experience

June'02 – September'02 : In Meteksan System and Computer Technologies Inc. Software Department as a Software Developer.
June'03 – September'03 : In Central Bank of the Republic of Turkey System Research and Planning Dep. as a System Developer.
August'04 – February'07 : In Baskent University Department of Computer Engineering as a Research Assistant.

Previous Projects

- A Data Fusion Approach in Protein Homology Detection (MSc Thesis)
- Finding Patterns and Extracting Tactical Plans from Football Sports Data using RFID Technology
- Computational Representation of Protein Sequences for Homology Detection
- Obstacle Avoidance Approach for a Mobile Robot
- Performance analysis comparison of GA, Hopfield Nets and Kohonen SOM on TSP
- Multiple Traveling Salesman Problem with Artificial Neural Network Approach
- Application Development with J2ME, MIDP and Personal Java for Symbian OS, Symbian Programming
- Stock Manager Automation System Design, Implementation and Maintenance
- Web Services App. Development for Central Bank of the Republic Of Turkey Sys. Research & Planning Dep.
- ATS, Airline Ticketing System Software Requirements and Design