ÖNSÖZ

"Yapay Sinir Ağları ve Genetik Algoritmalar Kullanılarak EKG Vurularının Sınıflandırılması" isimli doktora tez çalışmamın her aşamasında maddi ve manevi katkıları bulunan, bilgiyi özümseme, işleme ve değerlendirme konularında bana ışık tutan danışman hocam Sayın Doç. Dr. Tamer ÖLMEZ'e; kıymetli fikirlerini ve önerilerini esirgemeyen, İTÜ Biyomedikal Grubu olarak bilimsel çalışma hevesimizi pekiştiren hocam Sayın Prof. Dr. Ertuğrul YAZGAN'a katkılarından dolayı teşekkür ederim.

Bügüne kadar sevgi, anlayış ve moral desteklerini daima yanımda duyduğum, beni koşulsuz destekleyen sevgili aileme ayrıca teşekkür ediyorum.

Ocak, 2000

Zümray DOKUR

İÇİNDEKİLER

KISALTMALAR

a : Anormal kulakçık erken vurusu
ADD : Ayrık dalgacık dönüşümü
AGF : Alçak geçiren filtre
AND : Lojik 've' işlemi
ANN : Artificial neural network
AV : Atriyo-ventriküler düğüm
ÇKA : Çok katmanlı ağ
DFT : Discrete Fourier transform
DTÖV : Dalgacık temelli öznitelik vektörleri
E : Karıncık kaçak vurusu
EKG : Elektrokardiyogram
f : Yapay ve normal vuru birleşimi
F : Karıncık ve normal vuru birleşimi
FD : Fourier dönüşümü
FIR : Finite impulse response (sonlu darbe cevaplı)
FTÖV : Frekans temelli öznitelik vektörleri
GA : Genetik algoritma
GAL : Grow and learn (büyü ve öğren ağı)
GetÇKA : Genetik algoritmalar ile eğitilen çok katmanlı ağ
GetKoh : Genetik algoritmalar ile eğitilen Kohonen ağı
GetNic : Genetik algoritmalar ile eğitilen nicemleme ağı
GetPrz : Genetik algoritmalar ile eğitilen prizma ağı
GetRCE : Genetik algoritmalar ile eğitilen RCE (kısıtlı Coulomb enerji) ağı
KZFD : Kısa-zaman Fourier dönüşümü
L : Sol dal blok vurusu
LVQ : Öğrenen vektör nicemleyici (learning vector quantization)
MIT-BIH : Massachusetts Institute of Technology-Beth Israel Hospital
MLP : Multi layer perceptron
N : Normal vuru
OR : Lojik 'veya' işlemi
p : İletilmeyen P dalgası
P : Yapay vuru
PS : Parametre sayısı
PVC : Premature ventricular contraction (V, erken karıncık kasılması)
R : Sağ dal blok vurusu (Aynı harf, EKG'deki R dalgası için de kullanıldı)
RCE : Restricted Coulomb energy (kısıtlı Coulomb enerji ağı)
SA : Sino-atrial düğüm
SDD : Sürekli dalgacık dönüşümü
SOM : Self organizing map (öz-düzenleyen harita)
V : Erken karıncık kasılması
YGF : Yüksek geçiren filtre
YSA : Yapay sinir ağı

TABLO LİSTESİ

ŞEKİL LİSTESİ

Yapay Sinir Ağları ve Genetik Algoritmalar Kullanılarak EKG Vurularının Sınıflandırılması

ÖZET

Kalple ilgili hastalıkların teşhisinde, EKG (elektrokardiyogram) kayıtlarının hızlı yorumlanması önemlidir. Ancak, kalbin durumu hakkında yeterli ve anlamlı bilginin elde edilebilmesi, uzun kayıtların analizini gerektirir. Analizde harcanan süreyi kısaltmak, sistemde saklanan veri miktarını azaltmak ve bilgisayar destekli teşhise imkan sağlamak, EKG vurularının çeşitli hastalıklar hakkında ön bilgi içeren kategorilere otomatik olarak ayrılması ile mümkün olur.

Bilgisayar destekli analizi daha da iyileştirmek kategori sayısını artırmakla mümkündür. Ancak, kategori sayısı arttığında, sınıfları biribirlerinden ayırabilme başarımı da azalmaktadır.

Yapılan tez çalışması içerisinde, hem kategori sayısını artırmak hem de sınıfları biribirlerinden başarıyla ayırabilmek için dalgacık temelli *yeni bir öznitelik çıkartma yöntemi* ve karşılaştırma amacıyla *3 yeni yapay sinir ağı yapısı* geliştirilmiştir.

Çalışmada EKG vuruları 1'i normal, 9'u anormal olmak üzere 10 kategoriye ayrılmıştır: Normal vuru (N), sol dal blok vurusu (L), sağ dal blok vurusu (R), erken karıncık kasılması (V), yapay vuru (P), iletilmeyen P dalgası (p), anormal kulakçık erken vurusu (a), karıncık ve normal vuru birleşimi (F), yapay ve normal vuru birleşimi (f), karıncık kaçak vurusu (E)'dur.

Çalışmada karar verme işlemi üç aşamada gerçekleştirilmiştir: Normalizasyon işlemi, öznitelik çıkartma işlemi ve sınıflayıcı olarak yapay sinir ağlarının kullanımı.

Normalizasyon İşlemi

Öznitelik çıkartma işlemleri, işaretin tepeden tepeye genlik değişimlerinden, ofsetten ve EKG işareti içerisinde R tepesinin pozisyonundan etkilenmektedir. Bu etkiler hastanın fizyolojisine, yaşına, cinsiyetine ve ölçüm sisteminin parametrelerine bağlı olarak ortaya çıkmaktadır. Normalizasyon işlemi üç aşamada gerçekleştirilir:

1) Tek bir QRS kompleksini içeren dikdörtgen bir pencere oluşturulur. R tepesi pencere merkezine yerleşecek şekilde EKG işareti ayarlanır.
2) EKG'nin tepeden tepeye genliği 1 mV değerine normalize edilir. Böylece sınıflama performansının, EKG işaretinin maksimum genliğine bağımlılığı yok edilir.
3) Pencere içindeki EKG işaretinin ortalaması sıfır yapılarak ofset etkisi yok edilir.

Öznitelik çıkartma yönteminin, işaretin tepeden tepeye genliğine ve ofset değerine bağımlılığı, normalizasyon işlemi ile azaltılmıştır.

Öznitelik Çıkartma Yöntemleri

Çalışmada her EKG vurusu için R tepesi ortada olacak şekilde 256 ayrık veriden bir pencere oluşturulur. Test ve eğitim kümeleri farklı seçilmiştir ve herbiri 256 boyutlu 10×50 adet vektör içermektedir. Vektörlere önce normalizasyon daha sonra öznitelik çıkartma işlemi uygulanır. Her iki öznitelik çıkartma yönteminde de vektör boyutu 15 olarak seçilmiştir. Boyutu büyütmek avantaj sağlar, ancak hesaplama yükünü artırmaktadır.

Çalışma içersinde diverjans analizi, iki farklı öznitelik çıkartma işleminin karşılaştırılmasında ve en iyi özniteliklerin belirlenmesinde kullanılmaktadır. Diverjans analizi, sınıf vektörlerinin saçılımı hakkında bilgi vermektedir. Sınıf vektörleri saçıldıkça, diverjans değeri de düşmektedir.

EKG vurularını sınıflamada, Fourier ve dalgacık (wavelet) analizlerini kullanan iki öznitelik çıkartma yöntemi karşılaştırmalı olarak incelenmiştir. Çalışmanın başlangıcında, öznitelik vektörleri ayrık Fourier dönüşümü kullanılarak elde edilmiştir. Hesaplanan 40 adet ayrık Fourier katsayısından, işareti en iyi temsil eden 15 katsayıdan oluşan alt küme, dinamik programlama kullanılarak diverjans analizi ile belirlenmiştir.

İşaretin ayrık Fourier dönüşümü ile elde edilen öznitelik vektörlerinin saçıldığı, bu nedenle de dönüşümün düşük diverjans değeri verdiği gözlenmiştir. Elde edilen bulgular, çalışmayı, yeni bir öznitelik çıkartma yöntemi arayışına yöneltmiştir. Literatürde yaygın olarak kullanılan dalgacık analizinin EKG işaretlerini sınıflamada başarı ile kullanıldığı gözlenmiştir.

İkinci öznitelik çıkartma yönteminde, 256 örnekten oluşan pencere içindeki EKG işaretine ayrık dalgacık dönüşümü uygulanır. Daubechies-2 dalgacıkları kullanılarak her bir EKG vurusu için dalgacık katsayıları hesaplanıp, ayrık dalgacık düzlemi oluşturulur.

Araştırmanın ilk aşamalarında, öznitelik vektörü elemanları, 2-4. dalgacık ayrıştırma seviyelerindeki ayrıntı (d2, d3, d4) katsayılarıyla oluşturulmuştur. Ancak, bu şekilde oluşturulan vektörlerin uzayda saçıldığı ve düşük diverjans değeri ürettiği gözlenmiştir. Bu nedenle, 4. seviyedeki dalgacık yaklaşıklık katsayıları (a4) ve bu katsayıların otokorelasyonları da önceki özniteliklere ilave edilmiştir. Yeni oluşturulan öznitelik kümesine diverjans analizi uygulandığında, 4. seviye yaklaşıklık katsayıları ve bunların otokorelasyonlarının önceki katsayılardan daha avantajlı olduğu gözlenmiştir. EKG vurularını sınıflandırmak için öznitelik vektörlerinin oluşturulmasında 4. seviye yaklaşıklık katsayıları ve bunların otokorelasyonları *literatürde ilk kez* kullanılmıştır. Diverjans analizi, dalgacık temelli öznitelik vektörlerinin de bir miktar saçıldığını göstermektedir.

Sınıflayıcı Olarak Yapay Sinir Ağları

Sınıflama işlemlerinde karşımıza çıkan en önemli problem, incelenen fiziksel olayı temsil edebilecek doğru öznitelik uzayının her zaman oluşturulamamasıdır. Fiziksel olayların tümüne uyacak tek bir öznitelik çıkartma yöntemi de mevcut değildir. Bu yüzden sorunun çözümü, sınıflayıcı yapıları içinde aranmaktadır. Diverjans analizi, her iki yöntemle oluşturulan öznitelik vektörlerinin EKG'yi yeterince karakterize edemediğini göstermiştir. Bu nedenle tez çalışmasında, sınıflama başarımını artırmak için sınıflayıcı olarak yapay sinir ağları (YSA) kullanılmıştır. Yapay sinir ağlarının sınıflayıcı olarak kullanılmasına başlıca dört neden gösterilebilir: 1) Çözümleri

temsil eden ağırlıklara adım adım eğitimle yaklaşılır, 2) YSA'ların fiziksel olarak gerçeklenme imkanı bulunmaktadır, 3) YSA'lar karmaşık sınıf dağılımlarını kolaylıkla temsil edebilmektedir, 4) YSA'lar genelleme özellikleri sayesinde önceden karşılaşmadığı girişler için uygun sonuç üretebilmektedir.

Çalışmada, sınıflama başarımını artırmak ve düğüm sayısını azaltmak için *3 yeni ağ yapısı* geliştirilmiştir. Literatürde her ağın kendi özel algoritması ile eğitildiği gözlenmektedir. Tez içerisinde, yeni sinir ağı yapılarını kolayca eğitebilmek için genetik algoritmalar (GA) kullanılmıştır. GA'lar yeni ağ yapılarının eğitiminde başarılı sonuçlar vermiştir. Geliştirilen her yeni yapı için eğitim algoritması fazlaca değişmemiş, böylece yeni yapıların tasarımında ve onların eğitiminde bir zorlukla karşılaşılmamıştır.

Yeni ağ yapıları araştırılırken test ve gözlem amaçlı iki boyutlu örnek bir uzay tanımlanmıştır. Örnek uzayın iki boyutlu olması, sınıf sınırlarının ağ düğümleri tarafından temsil edilişinin görsel incelenmesine imkan vermiştir. Bu uzay, aynı zamanda klasik ağ eğitim algoritmalarının düğümleri yönlendirişi hakkında bilgi de vermektedir. Örnek uzayda beyaz, açık gri, ve koyu gri ile renklendirilmiş 3 tane sınıf vardır (siyah renkli bölgeler herhangi bir sınıfa ait değildir). Her sınıftan 100 adet vektör alınarak eğitim kümesi oluşturulur. Ağların sınıflama yeteneklerini zor koşullarda test edebilmek için özellikle düşük diverjans değerli bir uzay seçilmiştir. Bu analiz, yeni sinir ağı yapılarının geliştirilmesinde kullanılmaktadır.

Tez çalışmasında, genetik algoritmalar ile eğitilen *3 adet yeni YSA yapısı geliştirilmiştir.* Sınıf dağılımına ait bilgi düğümlerde iki şekilde temsil edilir: 1) Düğümlerin sadece yerel bilgiyi temsil etmesi, 2) Düğümlerin global bilgiyi temsil etmesi. İlk kısım için Kohonen, kısıtlı Coulomb enerji ağı (RCE, restricted Coulomb energy), büyü ve öğren ağı (GAL, grow and learn) ve öğrenen vektör nicemleyici ağını (LVQ, learning vector quantization) örnek olarak verebiliriz. Çok katmanlı ağ (ÇKA) ise ikincisine verilebilecek en önemli örnektir.

GetKoh Ağı

Genetik algoritmalar ile eğitilen Kohonen (GetKoh) ağında sınıf dağılımı, yerel bilgi içeren düğümler tarafından temsil edilir. GetKoh iki katmanlı bir ağdır. İlk saklı katmandaki düğümlerin ağırlık değerleri GA'lar ile belirlenir. İkinci katman lojik OR işlemini gerçekler. GetKoh ağı, GAL ve Kohonen ağları ile karşılaştırmalı olarak incelenmiştir. GAL ve Kohonen ağlarına göre GetKoh ağının örnek uzayı ve EKG test kümelerini daha az düğüm ile daha iyi sınıfladığı gözlenmiştir (Tablo 1).

Eğitim sırasında, GAL ağının düğümleri sınıf sınırlarına yerleşir. Öznitelik uzayında vektörler saçılırsa, eğitim tamamlandıktan sonra ağda aşırı sayıda düğüm üretilir. Kohonen'de eğitim sonrasında düğüm dağılımı GAL'dan farklıdır. Kohonen ağında vektörler, sınıfların iç bölgelerine homojen bir şekilde dağılır. Ancak, eğitim öncesinde yeterli sayıda düğüm ataması yapılmamışsa, sınıf sınırları yeterince iyi temsil edilememekte ve ağın sınıflama performansı düşmektedir. GetKoh ağında ise vektörler, ne sınıf iç bölgelerinde ne de sınıf sınırlarında yerleşme eğilimi gösterir. Genetik algoritmalar sayesinde vektörler optimum pozisyonlara yerleşir.

GetÇKA

Genetik algoritmalar ile eğitilen ÇKA'da (GetÇKA) sınıf dağılımı, global bilgi içeren düğümler tarafından temsil edilmektedir. GetÇKA üç katmanlı bir ağdır. İlk

saklı katmandaki düğümlerin ağırlıkları n-boyutlu öznitelik uzayında hiper düzlemleri temsil eder. Bu katmandaki düğümlerin ağırlık değerleri GA'lar ile belirlenir. İkinci katmanda, ilk katman çıkışları ile ikinci katman düğümlerinin ağırlıkları arasındaki minimum mesafe araştırılır. Bu katmanda sadece kazanan düğüm aktif yapılır. Üçüncü katman ise lojik OR işlemini gerçeklemektedir.
GetÇKA, çok katmanlı ağ ile karşılaştırılmıştır. ÇKA'ya göre GetÇKA ağının örnek uzayı ve EKG test kümelerini daha az düğüm ile daha iyi sınıfladığı gözlenmiştir. ÇKA'da üç dezavantaj bulunmaktadır: 1) Geriye-yayılma algoritması eğitim sırasında uzun zaman almaktadır. 2) Katmanlardaki düğüm sayısı eğitimden önce belirlenmelidir. Yapı, eğitim sırasında otomatik olarak belirlenemez. 3)Geriye-yayılma algoritması yerel minimumlara yakalanmakta, bu ise ağın başarımını düşürmektedir. Çalışmada yukarda sıralanan problemlere GA'lar ile çözümler getirilmiştir. Eğitim algoritması ile saklı katmana artımsal bir yapı özelliği kazandırılmış, ağın eğitim sırasında büyümesi sağlanmıştır.

GetPrz ve GetRCE ağları

Genetik algoritmalar ile eğitilen prizma (GetPrz) ve RCE (GetRCE) ağları tez içerisinde *ilk kez* geliştirilmiştir. Bu ağlarda sınıf dağılımı, global bilgi içeren düğümler tarafından temsil edilir. İlk saklı katmandaki düğümlerin ağırlıkları, n-boyutlu öznitelik uzayında hiper prizmaları (GetPrz'de) ve hiper küreleri (GetRCE'de) temsil etmektedir. ÇKA'nın ilk saklı katmanındaki düğümlerin ağırlıkları ise hiper düzlemleri temsil eder. ÇKA'da hiper düzlemler kesiştirilerek bölgeler oluşturulmaktadır. Yeni hibrit yapılarda da hiper prizma ve hiper küreler birbirleriyle kesiştirilerek bölgeler oluşturulmaktadır. Geliştirilen yeni ağlar, öznitelik uzayında düğüm sayısından daha fazla sayıda bölge oluşturmaktadır. Analizler, hiper prizma ve kürelerin ÇKA'da kullanılan düzlemlerden daha avantajlı olduğunu göstermektedir. Hiper kürelerin/prizmaların kesiştirilmesi ile ÇKA'nın oluşturabileceğinden daha fazla sayıda bölge elde edildiği gözlenmiştir.

RCE ağında iki dezavantaj bulunmaktadır: 1) RCE ağında, hiper küreler biribirleriyle kesişim yapmaz. Bir hiper küre tek bir kapalı bölge oluşturmaktadır. Oysa geliştirilen yeni yapılar, öznitelik uzayını daha fazla bölgeye ayırabilme yeteneğindedir. 2) RCE'nin eğitim algoritması hiper küreleri optimum pozisyona yerleştirmez.

GetPrz ve GetRCE ağları üç katmana sahiptir. İlk saklı katmandaki düğümlerin ağırlık değerleri GA'lar ile belirlenir. İkinci katmanda, ilk katman çıkışları ile ikinci katman düğümlerinin ağırlıkları arasındaki minimum mesafe araştırılır. Bu katmanda sadece kazanan düğüm aktif yapılır. Üçüncü katman ise lojik OR işlemini gerçekler. GetPrz ve GetRCE ağları, RCE ağı ile karşılaştırılmıştır. RCE'ye göre GetPrz ve GetRCE ağlarının, örnek uzayı ve EKG test kümelerini daha az düğüm ile daha iyi sınıfladığı gözlenmiştir.

GetNic Ağı

Genetik algoritmalar ile eğitilen nicemleme (GetNic) ağı tez içerisinde *ilk kez* geliştirilmiştir. Bu ağda sınıf dağılımı, global bilgi içeren düğümler tarafından temsil edilmektedir. İlk saklı katmandaki düğümlerin ağırlıkları n-boyutlu öznitelik uzayında hiper düzlemleri temsil eder. Ancak hiper düzlemler için bir kısıt getirilmiştir: Herbir hiper düzlem tek bir boyut eksenine dik, diğerlerine paralel olmalıdır. Böylece birinci katmandaki bir düğüm, tek bir parametre ile temsil edilir. Yeni hibrit yapıda da hiper düzlemler birbirleriyle kesiştirilerek öznitelik uzayı

bölmelenmektedir. Analizler, bu ağın incelenen tüm ağlardan daha az sayıda parametre kullanarak daha fazla bölge oluşturabildiğini göstermiştir.

GetNic üç katmana sahiptir. İlk saklı katmandaki düğümlerin ağırlık değerleri GA'lar ile belirlenir. İkinci katmanda, ilk katman çıkışları ile ikinci katman düğümlerinin ağırlıkları arasındaki minimum mesafe araştırılır. Bu katmanda sadece kazanan düğüm aktif yapılır. Üçüncü katman ise lojik OR işlemini gerçekler. İncelenen tüm ağlara göre GetNic ağının örnek uzayı ve EKG test kümelerini daha az parametre kullanarak daha iyi başarımla sınıfladığı gözlenmiştir.

Bir düğüm tek bir parametre ile temsil edildiğinden, ilk saklı katmandaki düğüm sayısı aşırı derecede büyür. Bu sorunu gidermek için, aynı boyut eksenini bölmeleyen düğümler toplanır ve herbir boyut için tek bir çıkış oluşturulur. Böylece, ilk saklı katmandaki çıkış düğümü sayısı, giriş düğümü sayısına (boyut sayısına) eşitlenir.

Tablo 1'de ağların ilk katmanlarındaki parametre sayıları (PS) ve sınıflama başarımları verilmiştir. Dalgacık öznitelik kümesi ile daha az düğüm kullanılarak daha iyi sınıflama başarımı elde edildiği gözlenmiştir. Sonuçlar, ayrık dalgacık dönüşümü ile elde edilen öznitelik vektörlerinin ayrık Fourier dönüşümü ile elde edilenlerden daha az saçıldığını göstermektedir.

Tablo 1. YSA'ların örnek uzayı ve EKG öznitelik kümelerini sınıflama başarımları

Yapay Sinir Ağları	Örnek uzay		Frekans bileşenleri ile EKG öznitelikleri		Dalgacık bileşenleri ile EKG öznitelikleri	
	PS	Başarım	PS	Başarım	PS	Başarım
GAL	2×24	0.91	15×41	0.94	15×22	0.95
Kohonen	2×49	0.90	15×49	0.80	15×49	0.93
GetKoh	2×14	0.93	15×38	0.93	15×28	0.99
ÇKA	3×40	0.76	16×30	0.90	16×20	0.93
GetÇKA	3×13	0.93	16×18	0.90	16×22	0.96
RCE	3×41	0.92	16×91	0.85	16×40	0.82
GetPrz	4×9	0.95	30×12	0.89	30×10	0.97
GetRCE	3×8	0.93	16×18	0.89	16×16	0.98
GetNic	1×12	0.95	1×18	0.94	1×16	0.98

RCE, ÇKA, Kohonen ve GAL ağlarına kıyasla, genetik algoritmalar ile eğitilen ağların daha az parametre kullanarak daha yüksek başarımlar verdiği gözlenmektedir.

Çalışmada kullanılan EKG verileri MIT-BIH veri tabanından alınmıştır. Algoritmalar, C dili ve MATLAB paket programı kullanılarak PC'de geliştirilmiştir.

Classification of ECG Beats by Using Artificial Neural Networks and Genetic Algorithms

SUMMARY

The electrocardiogram (ECG) has considerable significance in cardiac disease diagnosis. However, long-term ECG records are required to get meaningful information about the heart condition. In order to analyse fast and to reduce the data stored in the system, ECG beats must be automatically separated into categories that contain a-priori information about the diseases. Only the ECG beats of specific types are stored in the system upon request. Since the amount of the saved data is reduced, search and analysis become quite fast.

It is important to increase the number of categories to enhance the analysis. However, as the number of categories increases, performance of the class separation decreases.

In this thesis, a new feature extraction method based on wavelet transform and 3 new neural network structures are developed to increase both the number of categories and the performance of class separation.

ECG beats are classified into 10 categories: Normal beat (N), left bundle branch block beat (L), right bundle branch block beat (R), premature ventricular contraction (V), paced beat (P), nonconducted P wave (p), aberrated atrial premature beat (a), fusion of ventricular and normal beat (F), fusion of paced and normal beat (f), and ventricular escape beat (E).

Decision making is performed in three stages: Normalization process, feature extraction methods, and artificial neural networks as the classifiers.

Normalization Process

Feature extraction processes are affected by the peak-to-peak magnitudes, the offset of the signal, and R-peak position in the windowed ECG. These effects are due to physiology, sex and age of the patient, and parameters of the measurement system. In the normalization, three processes are realized:

1) A rectangular window is formed so that a single QRS complex is contained in this window. ECG signals are adjusted so that the positions of the R-peaks are centered in the window.
2) Peak-to-peak magnitudes of the ECG signal are normalized to 1 mV value. Thus, it is provided that classification decision does not depend on the maximum amplitude of the ECG records.
3) Mean value of the ECG signal in the window is fixed to zero value. Thus, offset is removed from the signal.

The dependence of the feature extraction method to the offset and the peak-to-peak magnitude of the signal is decreased by the normalization process.

Feature Extraction Methods

In this study, the window is formed by 256 discrete data. Test and training sets are separately formed by choosing 10×50 vectors of 15 dimensions. Feature extraction is carried out after the normalization process. Feature vector dimension is selected as 15 in both feature extraction methods. Increasing the dimension is advantageous, but high computational cost is introduced.

In the study, divergence analysis is utilized to compare the performances of the two different feature extraction methods and to determine the best features. Divergence analysis enables us to order the features in the vectors according to their importance in class separation, and gives information about the distribution of the class vectors. As the class vectors scatter, the divergence value decreases.

Two feature extraction methods, Fourier analysis and wavelet analysis, for ECG beat classification are comparatively investigated. At the beginning of the study, the elements of the feature vectors were formed by using the discrete Fourier transform (DFT). Divergence analysis together with dynamic programming was used to find a subset of best 15 features from a set of calculated 40 coefficients of the signal.

It is observed that vectors formed by the DFT of the signal scatter, leading to a low divergence value. Results directed us to find new feature extraction methods. At that moment, wavelet analysis emerged as a promising feature extraction method that is highly used in the literature.

The second feature extraction method involves a dyadic wavelet transform of a window of data of length 256 around the R-peak. Feature vectors are formed by using Daubechies-2 wavelets. For each ECG beat, wavelet coefficients are computed. Using these wavelet coefficients, wavelet plane is formed.

At the beginning of the study, feature vectors were formed by using the wavelet detail coefficients (d2, d3, d4) at the levels from 2 to 4. However, it is observed that feature vectors formed by these coefficients scatter, leading to a low divergence value. For that reason, 4th level wavelet approximation coefficients (a4) and autocorrelation coefficients of these approximation values were added to the previous features. Divergence analysis on this new feature set showed the 4th level wavelet approximation coefficients and their autocorrelations to be more advantageous than the previous coefficients. 4th level wavelet approximation coefficients and their autocorrelations are *firstly used in the literature* in ECG beat classification. Divergence value shows that wavelet based feature vectors also scatter in the feature space.

Artificial Neural Networks as Classifiers

Constitution of the right feature space for the physical process under study is a common problem in connection with classification. There is not a unique feature extraction method that can suit all kinds of data items. In this case, the solution of the problem is searched in the classifier structures. Hence, divergence analysis showed that the required representation of the ECG signal could not be achieved by the feature vectors obtained in the preceding methods. Therefore, in this thesis, artificial neural networks (ANNs) are used as classifiers to increase the classification performance. There are four reasons to use an artificial neural network as a classifier: 1) Weights representing the solutions are found by iteratively training, 2) ANN has a simple structure for physical implementation, 3) ANN can easily map complex class

distributions, 4) Generalization property of the ANN produces appropriate results to the input vectors that are not present in the training set.

In order to increase the classification performance and to decrease the number of nodes, *3 new neural network structures* are developed. In the literature, it is observed that each neural network was trained by its own special algorithm. In this thesis, genetic algorithms (GAs) are used to train the new neural network structures easily. GAs gave satisfactory solutions in the training of the new neural network structures. The training algorithms of the developed networks are similar. Thus, no difficulty is encountered in the design and training of the new networks.

During the search of new network structures, a two dimensional sample space is formed for test and observation purposes. Analysis in two dimensions gives a visual information about the representation of the class boundaries by the nodes. The sample space also gives information about how the training algorithms of the classical networks direct the nodes in the feature space. In the sample space, there are three classes: white, light gray, and dark gray (black coloured region does not belong to any class). Training set is formed by taking 100 vectors from each class in the space. A sample space with a low divergence value is specially selected to force the networks and to test their classification abilities. This analysis will be used in the development of new neural network structures.

In this thesis, *3 new ANN structures* trained by genetic algorithms are developed. Information about the class distribution is represented by the nodes in two ways: 1) By using nodes which represent only local information. 2) By using nodes which represent global information. Kohonen, RCE (restricted Coulomb energy), GAL (grow and learn), and LVQ (learning vector quantization) networks are examples for the first method. MLP (multi layer perceptron) is the best example for the second method.

GetKoh Network

In the first developed neural network (GetKoh, Kohonen network trained by GAs), class distribution is represented by using nodes that contain local information. GetKoh has two layers. The weights of the nodes in the first hidden layer are adjusted by the GAs. Second layer is used to perform logical OR operation. GetKoh is comparatively examined with the GAL and Kohonen networks. It is observed that, sample space and ECG test sets are successfully classified by the GetKoh with less number of nodes compared to the GAL and Kohonen networks (Table 1).

The nodes of the GAL network locate at the boundaries of the classes during the training. If the vectors scatter in the feature space, excessive number of nodes is generated after the training. The nodes of the Kohonen network homogenously move to the inner regions of the classes. If enough number of nodes is not assigned before the training, class boundaries can not be sufficiently represented, and the classification performance of the network decreases. The nodes of the GetKoh neither move to the inner regions, nor locate at the boundaries. They are directed to optimum vector positions by the GAs.

GetÇKA Network

In GetÇKA (MLP trained by GAs), class distribution is represented by using nodes that contain global information. Weights of the nodes in the first hidden layer form

hyper planes in n-dimensional feature space. GetÇKA has three layers. The weights of the nodes in the first hidden layer are adjusted by the GAs. Second layer is used to search the minimum distance between the weights of the second layer nodes and the output of the first layer. Only the output of the winner node is activated. The third layer is used to perform logical OR operation.

GetÇKA is compared with MLP. It is observed that, sample space and ECG test sets are successfully classified by the GetÇKA with less number of nodes compared to the MLP. MLP has three disadvantages: 1) Back-propagation algorithm takes too long time during learning, 2) The number of nodes in the hidden layers must be defined before the training. The structure is not automatically determined by the training algorithm, 3) Back-propagation algorithm may be caught by local_minima, which decreases network performance. It is observed that GAs have produced solutions to the problems mentioned above. By the use of GAs, learning takes less time to reach a satisfactory solution, and the structure of the hidden layer can be made incremental allowing the network structure to evolve during the learning.

GetPrz and GetRCE Networks

GetPrz (prism network trained by GAs) and GetRCE (RCE network trained by GAs) neural networks are *new structures* that are developed in this thesis. In these networks, class distribution is represented by using nodes that contain global information. Weights of the nodes in GetPrz and GetRCE represent hyper prisms and hyper spheres in n-dimensional feature space, respectively. The nodes in the first hidden layer of the MLP represent hyper planes. Class regions are formed by intersecting the hyper planes with each other. In the new hybrid structures, class regions are formed by intersecting the hyper prisms and hyper spheres with each other. There are more closed regions than the number of nodes in the feature space. Analyses show that in forming class regions, hyper prisms and hyper spheres are more advantages than the hyper planes used in MLP. More regions are formed by intersecting the hyper spheres/prisms compared to the MLP.

RCE network has two disadvantages: 1) Hyper spheres do not make intersections with each other. Each hyper sphere forms a single closed region. However, the new networks partition the feature space more than the RCE network does. 2) Training algorithm of the RCE does not direct the hyper spheres to the optimum positions.

GetPrz and GetRCE networks have three layers. The weights of the nodes in the first hidden layer are adjusted by the GAs. Second layer is used to search the minimum distance between the weights of the second layer nodes and the output of the first layer. Only the output of the winner node is activated. The third layer is used to perform logical OR operation. GetPrz and GetRCE are comparatively examined with RCE network. It is observed that, sample space and ECG test sets are successfully classified by the GetPrz and GetRCE with less number of nodes compared to the RCE.

GetNic Network

GetNic (restricted MLP trained by GAs) network is another *new structure* that is developed in this thesis. In the network, class distribution is represented by using nodes that contain global information. Weights of the nodes represent hyper planes in n-dimensional space. But, there is a constraint on the hyper planes. Each hyper plane should be perpendicular to one dimension axis, and should be parallel to the others.

Therefore, one node of the new network is formed by a single parameter. In the new hybrid structure, class regions are formed by intersecting the hyper planes with each other. Analyses show that the new network generates more regions than the other networks by using less number of parameters.

GetNic has three layers. The weights of the nodes in the first hidden layer are adjusted by the GAs. Second layer is used to search the minimum distance between the weights of the second layer nodes and the output of the first layer. Only the output of the winner node is activated. The third layer is used to perform logical OR operation. GetNic is comparatively examined with MLP. It is observed that, sample space and ECG test sets are successfully classified by the GetNic with less number of parameters compared to all other networks examined.

Since a node is represented by a single parameter, the number of nodes in the first hidden layer increases excessively. In order to solve this problem, the nodes, which partition the same dimension axis, are summed and a single output is formed for each dimension. Thus, the number of output nodes at the first hidden layer is made equal to the number of input nodes and the number of dimension.

Table 1 shows the number of parameters (NoP) in the first layer and classification performances (CP) of the networks. Wavelet feature set is found to result in better classification accuracy with less number of nodes. These results show that feature vectors obtained by the wavelet transform scatter less than the ones obtained by DFT.

Table 1. Classification results of the sample space and ECG feature sets by neural networks

Neural Networks	2-D Sample space		DFT features of ECG		Wavelet features of ECG	
	NoP	CP	NoP	CP	NoP	CP
GAL	2x24	0.91	15x41	0.94	15x22	0.95
Kohonen	2x49	0.90	15x49	0.80	15x49	0.93
GetKoh	2x14	0.93	15x38	0.93	15x28	0.99
MLP	3x40	0.76	16x30	0.90	16x20	0.93
GetÇKA	3x13	0.93	16x18	0.90	16x22	0.96
RCE	3x41	0.92	16x91	0.85	16x40	0.82
GetPrz	4x9	0.95	30x12	0.89	30x10	0.97
GetRCE	3x8	0.93	16x18	0.89	16x16	0.98
GetNic	1x12	0.95	1x18	0.94	1x16	0.98

It is observed that neural networks trained by genetic algorithms give satisfactory results using less number of parameters compared to RCE, MLP, GAL and Kohonen networks.

All the evaluations in this thesis are performed on the MIT-BIH arrhythmia database. The results presented in this thesis were produced by software implementation of the algorithms. The programs were developed using C language and MATLAB tools on PC.

1. GİRİŞ

1.1 Çalışmanın Amacı

İnsan vücudu üzerinden algılanan ve kalbin aktivitesi sonucu olarak ortaya çıkan belli tipteki elektriksel kökenli biyolojik işaretlere elektrokardiyogram veya kısaca EKG adı verilir. İnsan ölümlerinin büyük bir yüzdesini kalp rahatsızlıkları oluşturmaktadır. Bu yüzden kalbin çalışması sırasındaki bozuklukların iyi bir göstergesi olan ve hastaya zarar vermeden, vücut üzerinden kolaylıkla elde edilebilen EKG işaretleri, işlenme ve yorumlama açısından büyük önem taşımaktadır.

Teşhis için kullanılacak EKG işaretleri, genellikle birkaç saat süren uzun kayıt dosyalarında saklanmaktadır. Kalbin durumu hakkında yeterli ve anlamlı bilginin elde edilebilmesi uzun kayıtların analizini gerektirir. Bu nedenle, EKG işaretinin hızlı analizi için bilgisayar desteğine ihtiyaç duyulmaktadır. EKG kayıtlarının bilgisayar destekli analizi iki şekilde gerçeklenebilir. 1) Gerçek zamanda alınan veriler incelenir ve önemli vurularda kayıt işlemi başlatılır. Böylece sadece belirli anlardaki vurular saklandığı için kayıt boyları kısaltılmıştır. 2) Tüm EKG işaretleri gerçek zamanda analiz yapılmadan bir dosyaya kaydedilir. Kayıt daha sonra geliştirilen algoritma ile incelenerek vurular analiz edilmek üzere işaretlenir. Her iki yaklaşım, EKG işaretlerinin hızlı bir şekilde yorumlanmasına imkan vermektedir.

EKG işaretlerinin işlenmesinde amaç, bu işaretleri yüksek başarım ile mümkün olduğu kadar çok farklı sınıflara ayırabilmektir. Kategori sayısını artırmak, öznitelik çıkartma işleminin gücüne ve sınıflayıcıların genelleme yeteneklerine bağlıdır. Genellikle kategori sayısının artırılması, sınıflama başarımının düşmesine yol açmaktadır. Çalışmanın amacı, kategori sayısını ve bu kategorideki işaretlerin sınıflanma başarımını artıran öznitelik çıkartma yöntemlerini incelemek ve mümkün olduğunca az parametre ile fiziksel olarak gerçeklenebilen başarımı yüksek ağ yapıları geliştirmektir.

1.2 Kapsam

EKG vurularını belirlemede kullanılan işlemleri üç kısıma ayırmak mümkündür: 1) Normalizasyon işlemi, 2) Öznitelik çıkartma işlemi, 3) Sınıflama işlemi.

Öznitelik çıkartma işleminin sınıflama başarımına katkısını artırmak için iki farklı öznitelik çıkartma işlemi karşılaştırmalı olarak analiz edilmiştir. Her iki yöntemde de diverjans hesabı ve dinamik programlama birlikte kullanılarak başarımı artıracak öznitelikler araştırılmıştır [1]. Sınıflayıcı olarak yapay sinir ağlarının kullanılması düşünülmüştür. Fiziksel olarak gerçeklenmesi basit, az sayıda düğümden oluşan ve sınıflama başarımları yüksek yeni yapılar araştırılmıştır.

1.2.1 Normalizasyon İşlemi

EKG işaretlerinin genlik değişimleri (kazanç, ofset), öznitelik vektörünün elemanlarını olumsuz yönde etkilemektedir. Farklı hastalardan alınan aynı tip EKG vurularında bile dikkate değer bir değişim olduğu gözlenmektedir.

Normalizasyon işlemi ile EKG işaretlerinin genlikleri tepeden-tepeye 1 mV'a sabitlenir ve işaretin ofseti yok edilir. Hesap yükünü düşük tutmak için işaretin taban-hattını ayarlamak yerine ortalamasını sıfırlamak tercih edilmiştir. Böylece öznitelik vektörlerinin hastanın yaşına, cinsiyetine, fizyolojisine ve ölçüm sisteminin parametrelerine bağımlılığı bir ölçüde ortadan kaldırılmıştır. Öznitelik vektörlerinin sadece EKG şekil bilgisinden etkilenmesi sağlanmıştır.

1.2.2 Öznitelik Çıkartma İşlemi

Çalışmada iki farklı öznitelik çıkartma işleminin sınıflama performansı üzerindeki etkisi karşılaştırılarak incelenmiştir: 1) Frekans analizi 2) Dalgacık temelli analiz.

Her iki analizde de öznitelik uzayında sınıfların dağılımı hakkında bir bilgi elde etmek için diverjans hesabı kullanılmıştır. Diverjans değeri, incelenen yöntem için sınıf içi saçılımlar ve sınıflar arası uzaklıklar hakkında bir bilgi vermektedir. Diverjans değerini büyük yapacak çözümü seçmek sınıflayıcının başarımını artıracaktır. Küçük diverjans değerlerinde eğitim kümesindeki vektör sayısı, saçılımı daha iyi temsil etmesi için artırılmalıdır. Aksi taktirde ağlar eksik bilgi ile

eğitileceğinden genelleme özellikleri sönük kalacak ve sınıflama başarımları düşecektir. Ayrıca küçük diverjans değerlerinde eğitim sonrası ağların düğüm sayısının aşırı büyüdüğü gözlenmektedir. Her iki öznitelik çıkartma yönteminde de dinamik programlama ve diverjans hesabı birlikte kullanılarak sınıfları en iyi temsil eden bileşenler aranmıştır. Birinci öznitelik çıkartma yönteminde, frekans spektrumunda 40 bileşenden sınıfları en iyi temsil eden 15 bileşene inilmiştir.

İkinci öznitelik çıkartma yönteminde, Daubechies-2 dalgacığı kullanılarak 2., 3. ve 4. seviyelerdeki dalgacık ayrıntı katsayıları (66+34+18), 4. seviyedeki dalgacık yaklaşıklık katsayıları (18) ve 4. seviyedeki dalgacık yaklaşıklık katsayılarının 11 adet otokorelasyonlarıyla oluşturulan 147 boyutlu öznitelik vektörleri kullanılmıştır. Dalgacık düzlemi ve 4. seviyedeki dalgacık yaklaşıklık katsayılarının otokorelasyonlarından dinamik programlama ve diverjans hesabı ile en iyi özniteliklerin bulunması işlemi *literatürde ilk kez* gerçeklenmiştir. Her iki öznitelik çıkartma yönteminin karşılaştırmalı analizi, dalgacık temelli yöntem ile elde edilen özniteliklerin sınıfları daha iyi temsil ettiğini göstermiştir.

1.2.3 Sınıflayıcı Olarak Yapay Sinir Ağları

Veri analizi, her iki yöntemde de sınıflara ait öznitelik vektörlerinin diverjansının yeteri kadar büyük olmadığını, vektörlerin saçıldığını göstermektedir. Bu nedenle çalışmada sınıf sınırlarını daha iyi temsil edip başarımı artırmak için yapay sinir ağları kullanılmıştır. Sınıflayıcıların tasarımında, klasik ağ yapılarına göre daha az düğüm ile daha iyi sınıflama başarımı veren ve fiziksel olarak gerçeklenebilen basit ağ yapıları araştırılmıştır.

Geliştirilen her yeni ağ yapısı için farklı bir eğitim algoritması oluşturulmak zorundadır. Çalışmada incelenen ağ yapılarının basit olarak eğitimi genetik algoritmalar kullanılarak gerçeklenmiştir. Genetik algoritmaların kullanımı, çalışmayı eğitim algoritması geliştirmek yerine yapısal analize yönlendirmiştir.

Yapay sinir ağlarının eğitim sonrası düğüm pozisyonları, sınıf sınırlarını temsil edebilme yenenekleri ve eğitim algoritmasından kaynaklanan diğer problemleri, suni olarak oluşturulan 2 boyutlu örnek öznitelik uzayında test edilmiştir. Örnek öznitelik uzayının iki boyutlu seçilmesi bu incelemelerin görsel olarak analizine imkan vermektedir. Tüm ağlar aynı örnek öznitelik uzayında incelenmiştir. Öznitelik

uzayındaki incelemeler bölüm 4 içinde *3 yeni yapının* geliştirilmesine olanak sağlamıştır.

Yapay sinir ağlarını bilgiyi temsil etme bakımından iki kısıma ayırmak mümkündür:1) Herbir düğümün sadece bir yerel bilgiyi temsil etmesi (sınıf sınırları ile ilgili sadece bir ayrıntıyı saklaması). 2) Herbir düğümün tüm bilgiyi temsil etmesi (bütün ile ilgili bilgi taşıması). Birinci tip ağlara, öğrenen vektör nicemleyici (LVQ, learning vector quantization), Kohonen, büyü ve öğren ağı (GAL, grow and learn) ve kısıtlı Coulomb enerji ağını (RCE, restricted Coulomb energy) örnek verebiliriz. Herbir düğüm, bütüne ait bilgiyi değil sadece bütünün parçalarına ait bilgiyi saklamaktadır. Diğer düğümlerde bilginin izi bulunmadığı için düğümlerden birinin zarar görmesi durumunda saklanan parçaya ait bilgi yok olmaktadır. İkinci tip ağlara en bilinen örnek çok katmanlı ağdır (ÇKA). Herbir düğüm bütün hakkında bir bilgi taşımaktadır, düğümlerde özel bir bilgi tutulmamaktadır.

Çalışma içerisinde genetik algoritmalar kullanılarak 5 yapı geliştirilmiştir: GetKoh (genetik algoritmalar ile eğitilen Kohonen ağı), GetÇKA (genetik algoritmalar ile eğitilen ÇKA), GetPrz (genetik algoritmalar ile eğitilen prizma ağı), GetRCE (genetik algoritmalar ile eğitilen RCE ağı) ve GetNic (genetik algoritmalar ile eğitilen nicemleme ağı). GetKoh ağı birinci grup ağların sınıflama yeteneğinin analizinde kullanılmıştır. Diğer dört ağ ise ÇKA'nın sınıflama yeteneğini artırmak için geliştirilmiştir.

1.2.3.1 GetKoh Ağı

Çalışma içersinde GetKoh ağı kullanılarak sınıf sınırlarına ait bilgi, düğümlerde ayrı ayrı tutulmaktadır. Kohonen ve GAL ağına göre daha az düğüm ile daha iyi sınıflama başarımı elde edilmiştir. GetKoh ağı iki katmana sahiptir. Birinci katmandaki düğümlerin öznitelik uzayındaki pozisyonu genetik eğitimle bulunmaktadır. Düğüm sayısı, eğitim sırasında ihtiyaca göre otomatik olarak belirlenir. İkinci katman, birinci katmandaki düğümlerin tek bir çıkış düğümüne bağlanmasında kullanılmaktadır. Çalışma içersinde bu ağ için *yeni bir uyumluluk fonksiyonu* geliştirilmiştir.

1.2.3.2 GetÇKA Ağı

Bilginin tüm düğümlere dağıtılarak saklanmasına, çok katmanlı ağı örnek verebiliriz. Literatürde ÇKA'nın sınıflama amacıyla sıkça kullanıldığı gözlenmektedir. Bu ağın birinci katmanındaki düğümleri üzerinde yapılan analizler, çalışma içersinde 3 yeni yapının geliştirilmesine imkan sağlamıştır.

Genetik algoritmalar, GetÇKA'nın sadece birinci katmanındaki düğümlerin ağırlıklarının bulunmasında kullanılmıştır. Birinci katmandaki herbir düğüm, öznitelik uzayında bir hiper düzleme karşılık gelir. Hiper düzlemler kesiştirilerek öznitelik uzayında bölgeler meydana getirilir. Hiper düzlemlerin kesiştirilmesi işlemi ile az sayıda düğüm kullanılarak çok sayıda bölge oluşturabileceği gözlenmiştir. İkinci katman, giriş vektörüne önceden tanımlı en yakın hacimsel bölgeyi bulmak için kullanılır. Üçüncü katman, bulunan hacimsel bölgeyi çıkıştaki sınıf düğümüne bağlamaktadır.

Çalışma içersinde bu ağa ait yerel optimuma yakanlanmayan *yeni bir uyumluluk fonksiyonu* geliştirilmiştir.

1.2.3.3 GetPrz Ağı

Sınıflama performansını yükseltmek ve düğüm sayısını azaltmak için üç katmanlı *yeni bir yapı* geliştirilmiştir. GetPrz ağının birinci katmanındaki düğümler, öznitelik uzayında hiper prizmaları (iki boyutta dikdörtgenleri) temsil etmektedir. Prizmaların merkez koordinatları ve hacimlerinin büyüklüğü genetik algoritmalar kullanılarak bulunmaktadır. ÇKA'da olduğu gibi hiper prizmaların kesiştirilmesi işlemi ile az sayıda düğüm kullanılarak çok sayıda bölge oluşturabileceği gözlenmiştir. Hiper prizmaların sayısına ve uzay boyutuna bağlı olarak yapılan analizler, aynı sayıda düğüm kullanıldığında ÇKA'ya göre daha fazla bölge oluşturulabileceğini göstermektedir. İkinci ve üçüncü katman bir önce anlatılan yapı ile aynıdır.

1.2.3.4 GetRCE Ağı

Bir önceki yapıya benzer olarak öznitelik uzayındaki bölgeler, ikinci *yeni yapıda* hiper kürelerle temsil edilmektedir. Uyumluluk fonksiyonu aynıdır, n boyutlu öznitelik uzayında genetik havuzdaki dizilerde n+1 parametre bulunmaktadır. Hiper

prizmalar 2×n parametreden oluşturulurken, hiper küreler n+1 parametreden oluşturulmaktadır. Hiper prizmalar daha fazla parametre ile oluşturulmasına rağmen, karmaşık uzay dağılımlarını daha iyi temsil edebilme yeteneğine sahiptir. Hiper kürelerde hacim, sadece bir r yarıçap değeri ile kontrol edilmektedir. Prizma ağında ise hacim, n adet parametre ile kontrol edilmektedir.

1.2.3.5 GetNic Ağı

Çalışmada, düğüm sayısını daha aza indirgeyecek ve aynı zamanda sınıflama başarımını artıracak, fiziksel olarak gerçeklenmesi kolay üçüncü *yeni bir yapı* gerçeklenmiştir. GetNic ağı sayesinde daha az düğüm ile aynı sınıflama başarımı elde edilebilmiştir. Birinci katmandaki düğümler öznitelik uzayında hiper düzlemleri temsil etmektedir. Ancak hiper düzlemler, bir boyut hariç tüm koordinat eksenlerine paraleldir. Dolayısıyla bir düğüm sadece bir parametreden meydana gelmektedir. Çalışma içersinde GetNic ağı için hiper düzlemlerin sayısına ve uzay boyutuna bağlı olarak oluşan bölge sayısının *matematiksel ifadesi bulunmuştur.* Geliştirilen yeni yapı üç katmandan meydana gelmektedir. Birinci katmandaki hiper düzlemlere ait parametreler, genetik algoritmalar kullanılarak bulunmaktadır. Bir düğüm bir parametre ile temsil edildiği için birinci katman düğüm sayısı aşırı büyümektedir. Bu durumu yok etmek için aynı eksene ait çıkışlar toplanarak tek bir çıkışa bağlanmıştır. Böylece birinci katmanın çıkış düğüm sayısı, giriş düğüm sayısına eşitlenir. İkinci ve üçüncü katman bir önce anlatılan yapı ile aynıdır.

2. ELEKTROKARDİYOGRAM İŞARETLERİ İÇİN ÖZNİTELİKLERİN BELİRLENMESİ

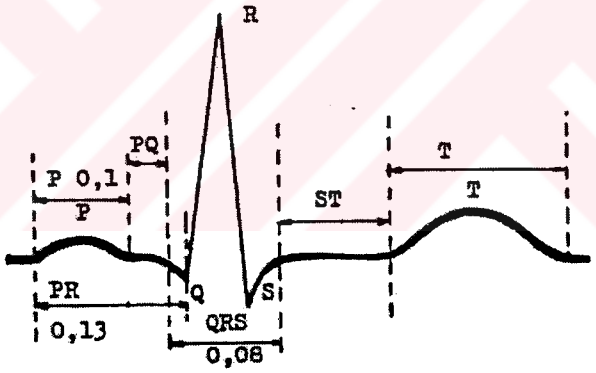
2.1 Elektrokardiyogram

Kalbin iletim sistemi; normal ritmik uyarıların doğduğu SA düğümü (sino-atrial düğüm), uyarının SA düğümünden AV düğümüne (atriyo-ventriküler düğüm) iletildiği internodal yollar, uyarının kulakçıklardan (atriyumlardan) karıncıklara (ventriküllere) geçerken gecikmeye uğradığı AV düğümü, uyarıyı kulakçıklardan karıncıklara ileten AV demeti ve uyarıyı karıncıkların bütün bölümlerine ileten purkinje liflerinin sol ve sağ dallarından oluşur [2].

Normal EKG işareti, kalbin dinlenme durumundaki izoelektrik seviyesi (taban seviyesi) üzerinde sıralanan belli başlı P, Q, R, S ve T adları verilen dalgalardan oluşur (şekil 2.1). SA düğmünde meydana gelen depolarizasyon dalgası, kulakçıklar içindeki iletim yolu üzerinde 30cm/s hızla ilerlerken kulakçık kaslarını uyarır ve bunun sonucunda kulakçıklar bir bütün olarak kasılarak kendi içinde kalmış kanın tümünü karıncıklara pompalar. Kulakçıkların kasılması sırasında EKG işaretinde P dalgası adı verilen değişim oluşur. 0,1s kadar olan P dalgasının genliği, kulakçık kasının aktivitesi hakkında bilgi verir. Depolarizasyon dalgası kulakçıklarla karıncıklar arasındaki AV düğümüne gelince hızı 5cm/s'ye kadar düşer. Bu sayede AV düğümü, bir gecikme elemanı gibi görev yaparak aksiyon potansiyel dalgasının karıncıklara, kulakçıklar kasıldıktan 0,1s sonra ulaşmasına neden olur. Aksiyon potansiyeli dalgası, his demeti ve purkinje lifleri yardımıyla tüm karıncık kaslarına 3m/s gibi bir hızla yayılınca karıncık bir bütün olarak kasılır. Karıncıkların kasılması sırasında EKG işaretinde 0,1s kadar süren ve QRS kompleksi adı verilen değişim gözlenir. Q, R ve S dalgalarından oluşan QRS kompleksinin 0,1s'den daha uzun sürmesi karıncıkların hemzaman olarak uyarılmadıklarını gösterir. Karıncık kaslarının kulakçık kaslarından daha çok ve güçlü olması nedeniyle QRS kompleksindeki R dalgası, P dalgasından genlik olarak oldukça büyüktür. S

dalgasından sonra karıncık kasları aynı potansiyeldedir ve bu nedenle T dalgasına kadar EKG işareti izoelektrik seviyededir. Bu seviye ST segmenti olarak bilinmektedir.

Karıncık kaslarının hızlı repolarizasyonu sonucunda T dalgası oluşur. EKG işaretinde dalgalar arasında kalan uzaklıklara *aralık* adı verilir. PR aralığı, his demeti iletim zamanını gösterir. PR süresi, P dalgasının başından QRS kompleksinin başlangıcına kadar olan süredir. 0.12-0.20s arasındadır. PR aralığının 0,2s'den uzun olması, his demetinde bir ileti bozukluğu olduğunu belirtir. ST aralığı özel olarak ST segmenti olarak isimlendirilir ve normalde izoelektrik seviyededir. ST segmentinin izoelektrik seviyeden olan sapmaları ve anormal uzunluğu da kalbin normal çalışmayışının göstergesi olarak kullanılmaktadır. İki dalga arası uzaklığa RR aralığı veya bir kalp periyodu adı verilir. RR aralığı, kalbin dakika vurum hızının bir göstergesidir ve kalp rahatsızlıklarının teşhisinde kullanılan parametrelerden biridir.



Şekil 2.1 EKG işaretinde gözlenen önemli dalgalar

2.1.1 Derivasyonlar

EKG işaretini kaydetmek üzere elektrodların hasta ile olan bağlantısı çeşitli şekillerde olmaktadır. Bağlantı şekilleri derivasyon olarak adlandırılmakta ve tıp literatüründe bu derivasyonlar standart isimlerle anılmaktadır: I, II, III nolu standart bipolar derivasyonlar (D_1, D_2, D_3); VR, VL, VF unipolar derivasyonlar (veya %50 oranında kuvvetlendirilmiş halleri: aVR, aVL, aVF) ve V_1-V_6 göğüs derivasyonları. Tez çalışmasında kullanılan MIT-BIH veri tabanında [3] bulunan EKG kayıtları iki farklı derivasyon ile vücuttan alınmışlardır. Derivasyonlardan biri II nolu standart

bipolar derivasyondur (D_2). Diğeri ise çoğunlukla V_1, zaman zaman V_2 veya V_5 göğüs derivasyonudur. Çalışma içerisinde II nolu standart bipolar derivasyon ile alınan EKG kayıtları kullanılmıştır.

2.2 EKG Vuruları

Bu bölümde, tez içerisinde sınıflandırılmaya çalışılan 1'i normal, 9'u anormal olmak üzere 10 farklı EKG vurusu anlatılacaktır [4]. Bu vurular ve etiketleri: normal vuru (N), sol dal blok vurusu (L, left bundle branch block beat), sağ dal blok vurusu (R, right bundle branch block beat), erken karıncık kasılması (V veya PVC, premature ventricular contraction), yapay vuru (P, paced beat), iletilmeyen P dalgası (p, non-conducted P-wave), anormal kulakçık erken vurusu (a, aberrated atrial premature beat), karıncık ve normal vuru birleşimi (F, fusion of ventricular and normal beat), yapay ve normal vuru birleşimi (f, fusion of paced and normal beat), karıncık kaçak vurusu (E, ventricular escape beat)'dur.

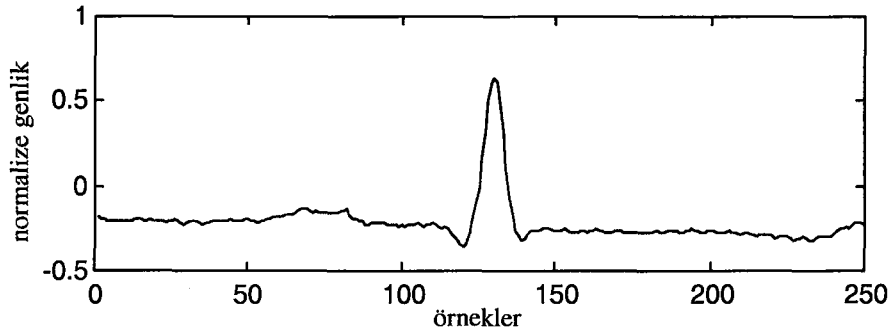
2.2.1 Normal Vuru

Normal vuru şekil 2.2'de görülen dalga şekline sahiptir. Normalde P dalgasının süresi, 0.10 saniyedir. 0.10-1.12s arasında ise geniş P dalgasından bahsedilebilir. PR süresi ortalama 0.16s bulunur. PR süresi genellikle kalp hızının artması ile kısalır, yaş ile uzar.

QRS süresi erişkinlerde 0.06-0.10s arasındadır. V_1-V_6 derivasyonlarında QRS süresi genellikle 0.01-0.02 saniye daha uzundur. P, PR, QRS süreleri normal olarak bipolar derivasyonlarda ölçülür. QRS süresinin uzaması, karıncıklar içinde iletimin gecikmiş olduğu anlamına gelir. En çok dal bloklarında görülür.

QT süresi, QRS kompleksinin başlangıcından T dalgasının sonuna kadar olan aralıktır. En belirgin olan herhangi bir derivasyondan ölçülebilir. QT süresi kalp hızına bağlı olarak değişiklik gösterir. Bu bakımdan hıza göre gerçek değerinin hesaplanması gerekir. Bu süre normal erişkinlerde 0.35-0.44s arasındadır.

T dalgasının normal süresi, 0.10-0.25s arasındadır. Ancak QT süresinin içine girmiş olduğu için klinik elektrokardiyografide T dalgasının süresi ayrıca ölçülmez.

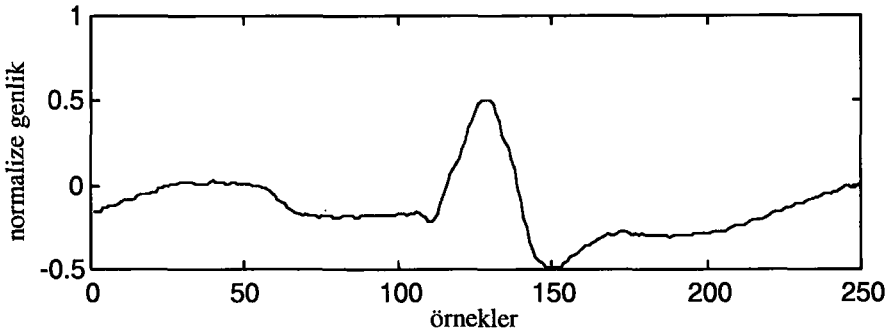


Şekil 2.2: Normal EKG vurusu (N tipi EKG vurusu)

2.2.2 Sol Dal Blok Vurusu

Sol dal bloğunda, sol karıncık aktivasyonu gecikir. Sol karıncık aktive olmaya başladığı zaman da sağ karıncığın bu potansiyellere karşı gelen potansiyelleri artık kalmamıştır. Bu yüzden karşılıksız kalan, dengelenemeyen sol karıncık potansiyelleri büyük sol karıncık vektörleri oluşturur. Bundan dolayı sol dal bloğunda ortaya çıkan QRS vektörlerinin genliği yüksek olabilir. QRS süresi 0.12 saniye veya daha uzundur. Karıncık aktivasyonunun gecikmesi, QRS'nin orta ve son kısımlarında olur. QRS dalgasının aksi yönde ST-T dalgası oluşur.

Klinikte sol dal bloğunun önemi, hasta hikayesi ve fizik muayene bulguları ile tespit edilir. Ancak, istatistik olarak sol dal bloğu vakalarının büyük çoğunluğunda (%90-95) önemli bir kalp hastalığı vardır.

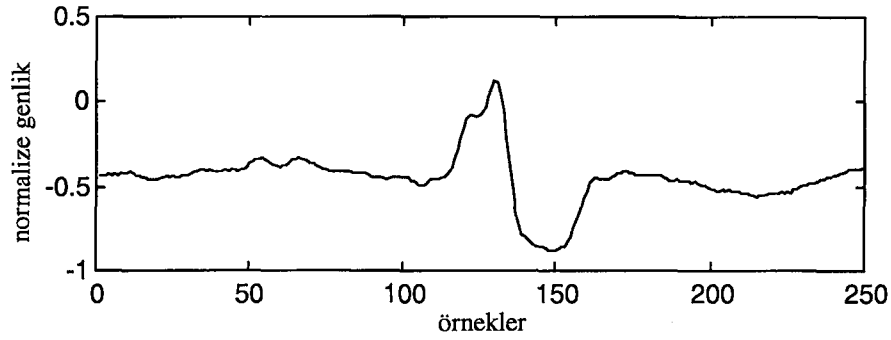


Şekil 2.3: EKG işaretinde gözlenen bir sol dal blok vurusu (L tipi EKG vurusu)

2.2.3 Sağ Dal Blok Vurusu

Sağ dal bloğunda, sağ karıncık aktivasyonunda gecikme olur. Bütün dal bloklarında olduğu gibi, sağ dal blokunda da QRS süresi uzamıştır. QRS süresi 0.11 saniye ve daha fazladır. Komplikasyonsuz sağ dal bloklarında QRS'nin başlangıç ve orta

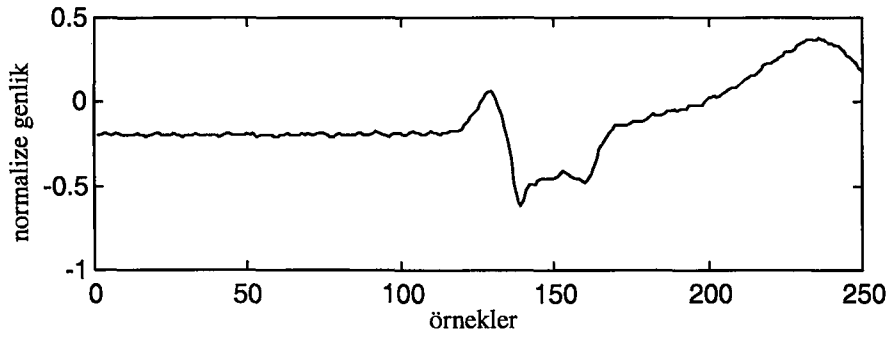
kısımları normaldir. Sadece terminal kısımları anormallik gösterir. QRS'nin terminal kısmında karıncık aktivasyonu yavaşlamış ve gecikmiştir. Terminalde oluşan bu anormal aktivasyon, V_{5-6} derivasyonlarında geniş S dalgasının ortaya çıkmasına neden olur. Ayrıca, D_{1-2} derivasyonlarında S dalgası halinde etkisini gösterir.



Şekil 2.4: EKG işaretinde gözlenen bir sağ dal blok vurusu (R tipi EKG vurusu)

2.2.4 Erken Karıncık Kasılması

Erken karıncık kasılması birçok kalp ritminde ortaya çıkan bir şekil bozukluğudur. AV düğümü altındaki bölgeden kaynaklanan elektriksel işaret, beklenenden daha önce oluşursa bu PVC olarak kendini gösterir. Karıncıklar bir PVC oluşturduğunda, kulakçıklar depolarize olamayabilirler. Depolarizasyonun gerçekleşmemesi durumunda P dalgası oluşmaz. Kulakçık depolarizasyonu meydana gelince, karıncık depolarizasyonunun etkisiyle P dalgası QRS kompleksi içerisinde kaybolur. QRS kompleksi geniş ve anormal bir görünümdedir. QRS süresi 0.12 saniyeden uzundur ve ters yöne bir sapma gösterebilir. PVC'yi hemen takip eden T dalgası, genellikle PVC'ye ait QRS kompleksine zıt yöndedir. Bu nedenle ST segmentinde anormallikler görülebilmektedir. Dakikada 10'un üzerinde PVC vurusu gelmesi hayati tehlike durumudur.

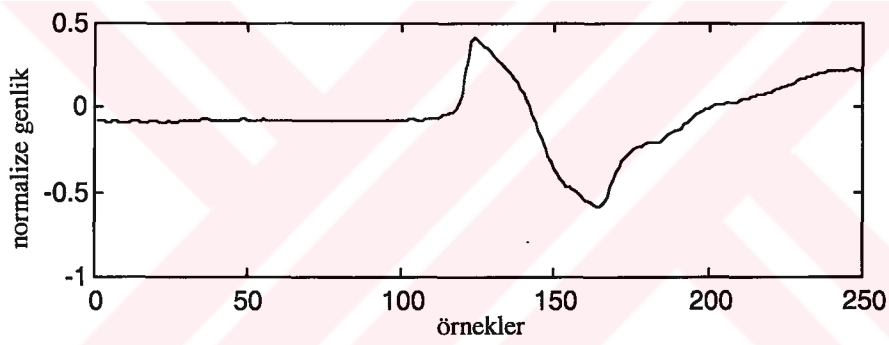


Şekil 2.5 Erken karıncık kasılması (PVC) içeren EKG kaydı (V tipi EKG vurusu)

2.2.5 Yapay Vuru

Kalp, kendi üretip oluşturduğu elektrik akımını yine kendi içindeki yollardan kalp kaslarına dağıtarak, kasılmasını sağlar. Kalbin doğal ileti sisteminde belirli aksaklıklar olması durumunda, kalp kasının kasılması için muhtaç olduğu elektriği yapay olarak sağlayan 'pacemaker' adı verilen sistemler kullanılmaktadır. Pacemaker, sağ karıncık, sağ kulakçık ve her ikisinden ardışıl olmak üzere yapay vuruyu kalbe vermektedir.

Yapay vuruda P dalgası nadiren görülmektedir. PR aralığı ölçülemez. QRS kompleksi, yapay vurudan sonra görülür. Genellikle 0.12 saniyeden uzundur. Eğer varsa, PP aralığı değişik değerler alır. Kalp vuru hızı, RR aralığı, yapay vuru aralığı pacemaker ayarına göre farklılıklar gösterir.



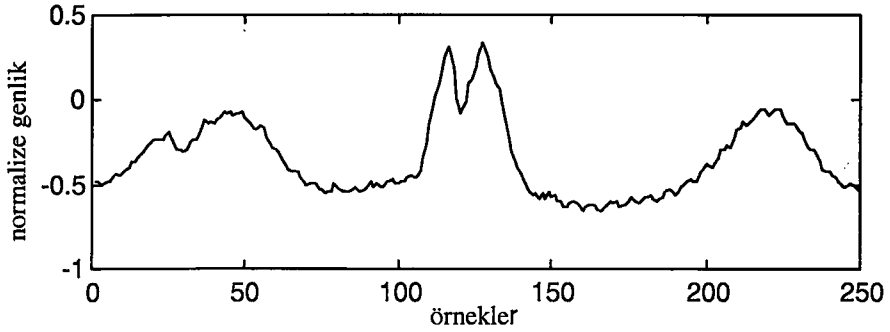
Şekil 2.6: Yapay vuru kaydı (P tipi EKG vurusu)

2.2.6 Anormal Kulakçık Erken Vurusu

İki kulakçıktan herhangi birinde oluşan ikincil uyaranlar sonucunda kulakçık erken vurusu oluşur. Uzun bir aralığın sonunda kısa kuplaj aralıklı bir kulakçık erken vurusunun oluşumu anormal kulakçık erken vurusu olarak adlandırılır.

Normal bir PP aralığından daha kısa süre içerisinde P dalgası gözlenir. QRS-T kompleksini, normal RR aralığından farklı bir bekleme süresi takip eder. PR aralığı, sinüsten iletilen vuruların PR aralığından daha kısa, uzun ya da eşit uzunlukta olabilir.

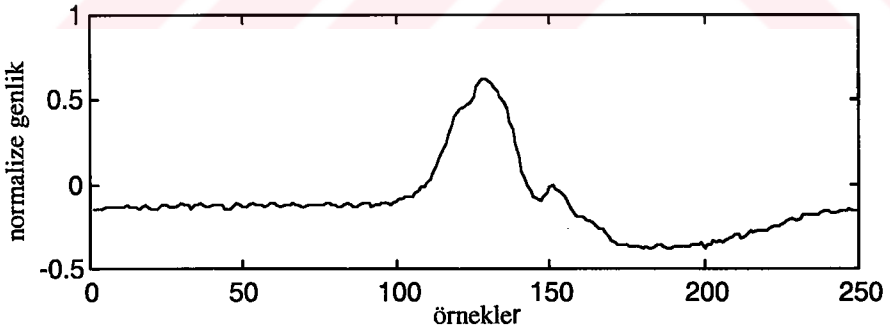
Bu vuru normal insanlarda da sıkça görülür. Yaş arttıkça oluşma sıklığı da artar.Tek başına organik bir kalp hastalığının işaretçisi değildir, ancak kulakçık taşikardisi ve kulakçık fibrilasyonunun habercisi olabilir.



Şekil 2.7: Anormal kulakçık erken vurusu içeren EKG kaydı (a tipi EKG vurusu)

2.2.7 Karıncık Kaçak Vurusu

Kalpteki SA ve AV düğümlerinin darbe üretim hızı, karıncıklardaki kaçak darbe üreten merkezlerin hızından düşük olduğunda ya da SA ve AV'de üretilen darbeler bloke olup, karıncıklara ulaşamadığında karıncık kaçak vurusu oluşur. Vurular sürekli veya geçici olabilir. Sürekli vurular gözlendiğinde ciddi bir kalp hastalığından söz edilebilir. Kalp vurum hızı dakikada 30-40 civarındadır. Kulakçıklarda herhangi bir aktivasyon meydana gelmediği için P dalgası gözlenmez. EKG kaydı, var olmayan P dalgası ve dakikada yaklaşık 40 kez düzenli olarak görülen geniş (0,12 saniyeden uzun) QRS kompleksleri ile karakterize edilir.

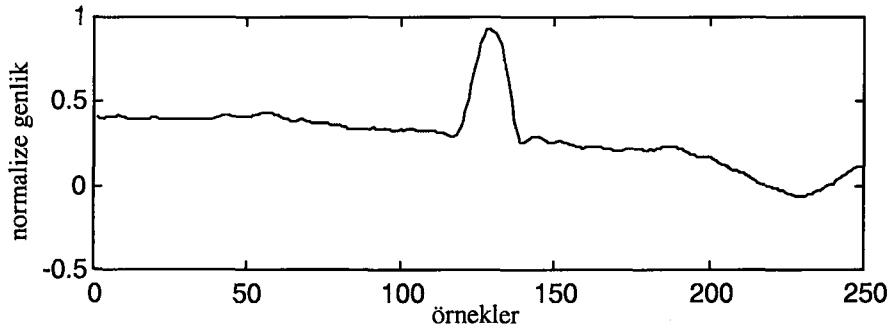


Şekil 2.8: Karıncık kaçak vurusu içeren EKG kaydı (E tipi EKG vurusu)

2.2.8 İletilmeyen P Dalgası

Normal bir vurunun hemen ardından bir P dalgası oluşması durumunda, bu sırada AV düğümü ve karıncık bekleme döneminde olacaklarından, işarete cevap veremeyeceklerdir (P dalgası bloke olacaktır). Bu durumda gözlenen vuruya, iletilmeyen P dalgası adı verilir. P dalgası herhangi bir derivasyonda gözlenemeyebilir.

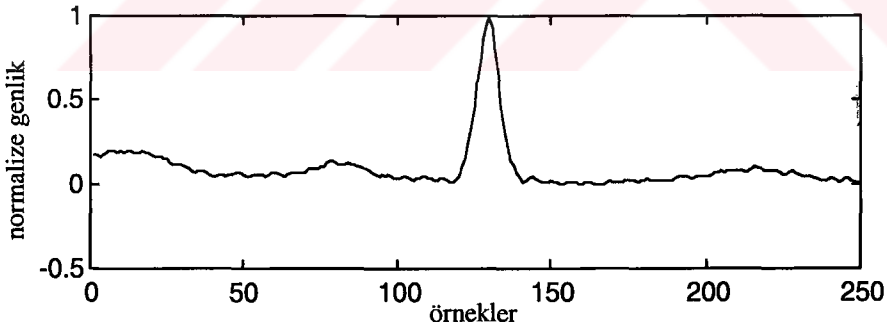
Normal şahıslarda da iletilmeyen P dalgası görülebilir. Bu türden bir vuru ile karşılaşıldığında sadece elektrokardiyografik verilere dayanarak, ciddi bir kalp rahatsızlığının varlığından sözetmek doğru olmaz; ancak, bu vuru kulakçık taşikardisi ve kulakçık fibrilasyonunun habercisi olabilmektedir.



Şekil 2.9: İletilmeyen P dalgası (p tipi EKG vurusu)

2.2.9 Karıncık ve Normal Vuru Birleşimi

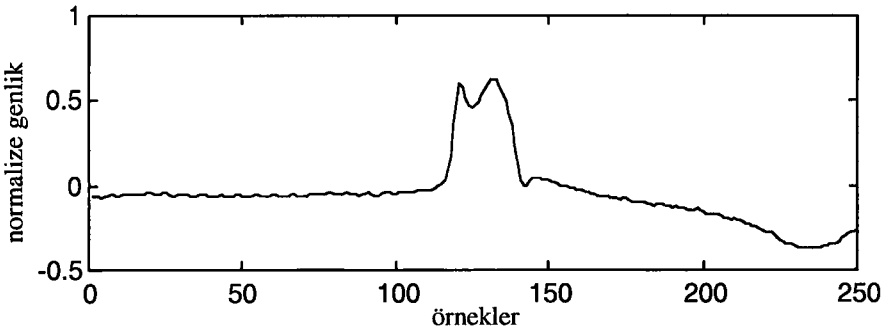
Kalbin normal iletimi sırasında karıncıkta ikincil uyarılar oluşması durumunda ortaya çıkan işaret, karıncık ve normal vuru birleşimi olarak adlandırılır. Geniş QRS kompleksleri (0,12 saniyeden uzun) söz konusudur.



Şekil 2.10: Karıncık ve normal vuru birleşimi (F tipi EKG vurusu)

2.2.10 Yapay ve Normal Vuru Birleşimi

Kalbin normal iletim sisteminde gecikme oluşması durumunda 'pacemaker' sistemi devreye girer ve yapay olarak ürettiği vuruları kalbe iletir. Bu sırada kalp de doğal yollardan vuru oluşturabilir ve her iki vuru aynı anda gözlenebilir. Ortaya çıkan birleşik vuruda genişlemiş QRS kompleksi söz konusudur. P dalgası belirgin değildir.



Şekil 2.11: Yapay ve normal vuru birleşimi (f tipi EKG vurusu)

2.3 MIT-BIH Veri Tabanı

MIT-BIH (Massachusetts Institute of Technology-Beth Israel Hospital Data Base) veri tabanında bulunan EKG kayıtları, Beth-Israel Hospital Aritmi Laboratuvarı tarafından 1975-1979 yılları arasında alınmış olan 4000'i aşkın uzun-dönem Holter kayıtlarından oluşturulmuş bir kümedir [3]. Bu veri tabanında 48 adet EKG kaydı bulunmaktadır. Bu 48 kaydın herbiri 30 dakika uzunluktadır.

EKG kayıtları yaşları 32-89 arasında değişen 25 erkek ve 23-89 yaşları arasındaki 22 kadından alınmıştır. Hastaların tümüne iki farklı derivasyon uygulanmıştır. Kayıtların birçoğunda derivasyonlardan biri II nolu standart bipolar derivasyondur. Diğeri ise çoğunlukla V_1, zaman zaman V_2 veya V_5, kayıtların birinde de V_4 derivasyonudur. Normal QRS kompleksleri genellikle II nolu standart bipolar derivasyonla alınan işarette belirgindir.

İşaret 360 Hz ile örneklenmiş olup ±5 mV bölgesinde 11-bit rezolüsyona sahiptir. Örnek değerleri 0-2047 kapalı aralığında değerler alırlar ve 1024 değeri 0 V'a karşılık gelmektedir.

2.4 EKG İşaretleri İçin Öznitelik Çıkartma Yöntemleri

Literatürde EKG vurularını sınıflamak için kullanılan öznitelik çıkartma yöntemlerini üç grup altında toplayabiliriz.

1) Direkt yöntem: Bu yöntemde QRS kompleksi içinden R tepesi belirlenir ve QRS'yi kapsayacak şekilde R tepesi etrafında seçilen bir pencere içindeki genlik değerleri direkt alınarak öznitelik vektörleri oluşturulur.

Öznitelik vektörlerinin oluşturulması işlemi basittir ve sınıflama sonucunun elde edilmesi için geçen süreye ilave bir hesap yükü getirmez. Ancak öznitelik vektörleri R tepesinin doğru olarak belirlenmesinden çok etkilenir. QRS kompleksini içine alacak şekilde R tepesinin etrafında seçilen verilerin sayısı boyut sayısını belirleyecektir ve boyut, QRS'nin tamamını içine alması durumunda 100'ler mertebesine yakındır. Dolayısıyla ağların giriş uzayları büyük olduğu için ağda bir düğüm için harcanan parametre sayısı aşırı büyük olacaktır. Aşağıda bu gruba giren bazı çalışmalardan örnekler verilmektedir.

Lurtz Reinhardt ve arkadaşlarının gerçeklediği çalışmada iki tip dalga şeklinin sınıflandırılması temel alınmıştır [5]: Anterior ve inferior miyokardiyal enfarktüs. Öznitelik uzayı, vücut üzerine yerleştirilen 12 elektroddan alınan genlik değerleri ile oluşturulur. Öznitelik uzayının boyutu 12'dir. Sınıflayıcı olarak kullanılan LVQ ağı %86 başarı vermiştir.

Kuo Yonghong ve arkadaşlarının MIT-BIH veri tabanı üzerinde gerçekleştirdikleri çalışmada, normal, yapay, 'fusion' ve erken karıncık kasılması vurularının sınıflandırılması temel alınmıştır [6]. Öznitelik vektörleri QRS işaretinin direkt genlik değerlerinden oluşturulmaktadır. Özniteliklerin gürültüye bağımlılığını azaltmak için bir ön işlem kullanılmıştır. Çalışmada ART2 modelli bir sınıflayıcı kullanılmıştır; ancak başarımlar hakkında bilgi verilmemiştir.

Marcelo R. Risk ve arkadaşlarının HMS-MIT-FFMS veri tabanı üzerinde gerçekleştirdikleri çalışmada, normal ve 'ektopik' vuruları sınıflandırılmıştır [7]. Öznitelik vektörleri QRS genlik değerlerinin direkt alınmasıyla oluşturulmaktadır. Sınıflayıcı olarak öz-düzenleyen harita (self organizing map, SOM) kullanılmıştır. Çalışmada sınıflama başarımı hakkında bilgi verilmemiştir.

Surekha Palreddy ve arkadaşlarının gerçekleştirdiği çalışmada SOM ve LVQ kullanılarak %92 başarım ile dört farklı EKG vurusu sınıflandırılmıştır: normal vuru, karıncık ektopik vurusu, karıncık ile normal vuruların birleşimi ve bunların dışındaki vurular [8]. Kayıtlar MIT-BIH veri tabanından alınmıştır. 56 boyutlu öznitelik vektörleri oluşturulmuştur. Özniteliklerin 51 tanesi direkt QRS genlik değerlerinden, diğerleri ise RR aralığı, QRS genişliği gibi şekil bilgilerinden oluşturulmuştur.

Philip de Chazal ve arkadaşının gerçekleştirdiği çalışmada 7 tip EKG vurusu %70,9 başarım ile sınıflandırılmıştır [9]: Normal vuru, sol, sağ ve 'bilateral karıncık

hipertropileri', 'anterior', 'inferior' ve karışık 'miyokardiyal enfarktüsler'. Sınıflayıcı olarak yapay sinir ağları kullanılmıştır. Hastaların yaş-cinsiyet bilgileri, bazı özniteliklerin kendi aralarında oranları, genlik, süre ve alan gibi bazı şekil bilgileri kullanılarak 229 boyutlu öznitelik vektörleri oluşturulmuştur.

Rosario Silipo ve arkadaşının gerçekleştirdiği çalışmada 7 tip EKG vurusu %65 başarım ile sınıflandırılmıştır [10]: Normal vuru, sol, sağ ve 'bilateral karıncık hipertropileri', 'anterior', 'inferior' ve karışık 'miyokardiyal enfarktüsler'. Sınıflayıcı olarak hibrit bir yapay sinir ağı kullanılmıştır. Öznitelik vektörleri, EKG işaretinin 39 bileşene indirgenmiş genlik değerlerinden meydana getirilmiştir. EKG işaretlerinin analizinde MIT-BIH ve ST-T veri tabanı kullanılmıştır.

Rosario Silipo ve arkadaşlarının gerçekleştirdikleri çalışmada 5 tip EKG vurusu %97 başarım ile sınıflandırılmıştır [11]: Normal vuru, 'supraventricular' aritmi, karıncık aritmisi, 'aberrate' vuru ve iletilmeyen 'supraventicular' vuru. Sınıflayıcı olarak bağlaşımsal bellek kullanılmıştır. Öznitelik vektörleri, EKG işaretleri üzerinden direkt olarak alınan 108 adet genlik değerinden meydana getirilmiştir. EKG işaretlerinin analizinde MIT-BIH ve VALE veri tabanları kullanılmıştır.

Yu Hen Hu ve arkadaşlarının gerçekleştirdikleri çalışmada 4 tip EKG vurusu %97 başarım ile sınıflandırılmıştır [12]: Normal vuru, karıncık aritmisi, karıncık ve normal vurunun birleşimi ve sınıflanamayan vurular. Sınıflayıcı olarak Kohonen ve LVQ ağları kullanılmıştır. EKG işaretleri üzerinden direkt olarak alınan 29 genlik değeri ana bileşen analizi kullanılarak 9 değere indirgenmiştir. Öznitelik vektörlerinin boyutu, bu 9 değere şekil bilgisi de ilave edilerek 12'ye çıkarılmıştır. EKG işaretlerinin analizinde MIT-BIH veri tabanı kullanılmıştır.

G. Bortolan ve arkadaşının gerçekleştirdiği çalışmada 5 tip EKG vurusu sınıflandırılmıştır [13]: normal vuru, sol ve sağ karıncık hipertropileri, anterior ve inferior miyokardiyal enfarktüsler. Beş tane YSA'nın performansı karşılaştırmalı olarak analiz edilip maksimum %83 başarım sağlanmıştır. Ağlara 'pruning' teknikleri uygulanarak boyut indirgemenin yapay sinir ağları üzerindeki etkisi incelenmiştir.

Rosaria Silipo ve arkadaşlarının gerçekleştirdiği çalışmada 7 tip EKG vurusu %65 başarım ile sınıflandırılmıştır [14]: Normal vuru, sol, sağ ve 'bilateral karıncık

hipertropileri', 'anterior', 'inferior' ve karışık 'miyokardiyal enfarktüsler'. Kohonen ağı ve hibrit sistemler kullanılmıştır.

2) Dönüşüm yöntemi: Bu yöntemde de QRS içinden R tepesi belirlenir. QRS'yi kapsayacak şekilde R tepesi etrafında seçilen bir pencere içindeki genlik değerleri direkt alınmaz; bir dönüşüm işleminden sonra öznitelik vektörleri oluşturulur.

Bu dönüşüm işlemi ile gürültünün, R tepesinin düzgün belirlenemeyişinin ve istenmeyen diğer etkenlerin öznitelik vektörü üzerindeki etkisi azaltılır. Dönüşüm işleminin karmaşıklığı sınıflama sonucunun elde edilmesi işlemine ilave bir hesap yükü getirecektir.

Öznitelik vektörü boyutunun da mümkün olduğu kadar küçük tutulması istenir. Bu nedenle dönüşüm sonucu oluşan vektörlerin boyutlarının küçültülmesi için boyut azaltma yöntemleri kullanılır. Sınıflama kararında etkili olmayan bileşenler öznitelik vektöründen çıkarılarak boyut küçültülür.

Tez içerisinde dalgacık [15, 16] ve frekans dönüşümü [17-23] ile öznitelik çıkartma işlemleri üzerinde durulmuştur. Aşağıda bu gruba giren bazı çalışmalardan örnekler verilmektedir.

Kei-ichiro Minami ve arkadaşlarının gerçekleştirdikleri çalışmada 2 tip EKG vurusu (normal ve karıncık vuruları) %98 başarım ile sınıflandırılmıştır [24]. Sınıflayıcı olarak çok katmanlı ağ kullanılmıştır. Öznitelik vektörleri, işaretin 5 ayrık Fourier katsayısı kullanılarak oluşturulmaktadır. EKG işaretlerinin analizinde bir veri tabanı kullanılmamıştır.

Jun Yao ve arkadaşları MIT-BIH veri tabanı üzerinde gerçekledikleri çalışmada normal vuru, dal-blok vurusu, enfarktüs ve erken karıncık kasılması vurularını %99 başarı ile sınıflamışlardır [25]. İki katmanlı bir yapay sinir ağı kullanılmıştır. İkinci katman tek katmanlı 'perceptron' şeklindedir. Birinci katman dalgacık dönüşümü kullanılarak gerçeklenmiştir. Literatürde bu yapılara dalgacık sinir ağı adı verilmektedir. Öznitelik uzayının boyutu 170'tir.

Valtino X. Afanso ve arkadaşlarının MIT-BIH veri tabanı üzerinde gerçekleştirdikleri çalışmada, normal ve yapay vurular %91 başarı ile sınıflandırılmıştır [26]. Öznitelik vektörleri band geçiren filtreler kullanılarak elde

edilmektedir. İşaret alt bantlara ayrılarak, bu alt bantlardan elde edilen bilgiler kullanılarak vektörler oluşturulur. Sınıflama işleminde karar ağacı algoritması kullanılmıştır.

Joseph S. Paul ve arkadaşlarının MIT-BIH veri tabanı üzerinde gerçekleştirdikleri çalışmada, normal vuru ve erken karıncık kasılmasına ait dalga şekilleri sınıflandırılmıştır [27]. Öznitelik vektörleri iki aşamada elde edilmiştir. Önce EKG işaretine ayrık kosinüs dönüşümü uygulanmış ve daha sonra bu dönüşüme özbağlaşımlı (AR) modelleme uygulanarak öznitelik vektörü elemanları oluşturulmuştur. İşaretten özniteliklerin çıkartılması işlemi uzun zaman almaktadır. Çalışma iki tip işaretin birbirinden ayrılması için gerekli işlemlerin analizine yönelmiştir. Sınıflama işlemi incelenmediği için başarım bilgisi verilmemiştir.

Ismail Jouny ve arkadaşlarının gerçekleştirdikleri çalışmada herbir EKG vurusu, yapay sinir ağları kullanılarak normal ve anormal olmak üzere iki sınıftan birine dahil edilmiştir [28]. Kayıtlar MIT-BIH veri tabanından alınmıştır. Dalgacık temelli öznitelik çıkartma yöntemi kullanılmıştır. Sınıflama başarımı %70 olarak belirtilmiştir.

Fredric M. Ham ve arkadaşının gerçeklediği çalışmada 2 tip EKG vurusu %99 başarım ile sınıflandırılmıştır [29]: Normal ve erken karıncık vurusu. Öznitelik vektörleri 3 boyutludur ve vektör elemanları doğrusal öngörü katsayıları (LPC) kullanılarak oluşturulmuştur. Yapay sinir ağı olarak adaptif rezonans teorisi kullanılmıştır. EKG işaretlerinin analizinde MIT-BIH veri tabanı kullanılmıştır.

Branko G. Celler ve arkadaşının gerçekleştirdiği çalışmada 7 tip EKG vurusu sınıflandırılmıştır [30]: Normal vuru, sol, sağ ve bilateral karıncık hipertropileri, anterior, inferior ve karışık miyokardiyal enfarktüsler. Saklı katmanı olmayan bir yapay sinir ağı kullanılarak %68,8 başarım elde edilmiştir. Öznitelik vektörleri sadece QRS kompleksinden elde edilmiştir, P ve T dalgalarına ait bilgi yoktur. Güç spektral yoğunluğu ve iki farklı dalgacık dönüşümünden elde edilen bazı istatiksel parametreler ile yaş/cinsiyet bilgileri kullanılarak öznitelik vektörleri oluşturulmuştur.

3) Karakteristik noktalar yöntemi: EKG işareti içinde P, Q, R, S ve T noktalarının pozisyonları bulunur. Bu pozisyonlardaki genlik değerleri, birbirleri arasındaki

süreler ve diğer şekle dayalı (morfolojik) bileşenler kullanılarak öznitelik vektörleri oluşturulur.

Şekle dayalı parametrelerin sayısı az olduğu için öznitelik vektörlerinin boyutu küçük olacaktır. Karakteristik noktaların çıkartılması için bir ön işlem gerekmektedir. Ayrıca bu noktaların yerlerinin bulunması işlemi gürültüden de etkilenmektedir. Literatürde sınıflama işlemine yönelik olmayan sadece karakteristik noktaları belirleyen yöntemler bulunmaktadır. Bu çalışmaların bazılarında karakteristik noktaların belirlenmesinde gürültünün etkisinin de araştırıldığı gözlenmektedir.

Aşağıda karakteristik noktaların analizine giren bazı çalışmalardan örnekler verilmektedir.

Oliver Wieben ve arkadaşlarının MIT-BIH veri tabanı üzerinde gerçekledikleri çalışmada sadece PVC vurularının sınıflandırılması temel alınmıştır [31]. Öznitelik uzayı, QRS üzerinden 9 farklı şekil bilgisi çıkartılarak oluşturulur. Öznitelik uzayının boyutu 9'dur. Bulanık sınıflayıcı kullanılarak %80 başarı elde edilmiştir.

Sandor Miklos Szilágyi'nin MIT-BIH veri tabanı üzerinde gerçeklediği çalışmada, normal ve PVC vurularının sınıflandırılması temel alınmıştır [32]. Öznitelik uzayı, EKG işareti içinde P, Q, R, S, ve T karakteristik noktalarının çıkartılmasıyla elde edilir. Çalışmada bu noktalar kullanılarak elde edilen öznitelikler tam belirtilmemiştir. QRS kompleksinin belirlenmesinde %99,8 başarı olduğu belirtilirken sınıflama başarımının düşük olduğu ifade edilmektedir. Karakteristik noktaların çıkartılmasından önce gürültü süzme işlemlerinin gerçeklenmesi üzerinde durulmaktadır. Sınıflayıcı olarak SOM kullanılmaktadır.

H. Gholam Hosseini ve arkadaşının MIT-BIH veri tabanı üzerinde gerçeklediği çalışmada, dört farklı EKG vurusu sınıflandırılmıştır [33]. Yazıda dört farklı vurunun isimleri ve her vuru için başarım yüzdesi tam olarak ifade edilmemiştir. Sadece, normal ve erken karıncık kasılmasına ait vuruları sınıflamada %99'a varan bir başarımın elde edildiği belirtilmektedir. Şekil bilgisini kullanarak öznitelik çıkartmanın gürültüye ve hastaya çok bağlı olduğu; ancak istatiksel yöntemlerin daha bağımsız veriler ürettiği ifade edilmektedir. Çalışmada, ortalama ve kovaryans hesabını temel alan sınıflayıcılar kullanılmıştır.

A. Elias ve arkadaşlarının gerçekledikleri çalışmada, normal, sol karıncık hipertropisi, sağ karıncık hipertropisi, iki-karıncık hipertropisi, anterior miyokardiyal enfarktüs ve inferior miyokardiyal enfarktüs vuruları sınıflandırılmıştır [34]. Öznitelik vektörleri, EKG şekli içinde P, Q, R, S ve T noktalarının genlik/süre ve hastaların yaş-cinsiyet bilgileri kullanılarak oluşturulmuştur. Öznitelik vektörleri hakkında yeterli bilgi çalışmada verilmemiştir. Çok katmanlı ağ kullanılarak %70 başarı elde edilmiştir.

Ming-Yao Yang ve arkadaşlarının gerçekleştirdiği çalışmada 7 tip EKG vurusu sınıflandırılmıştır [35]: normal vuru, sol ve sağ dal blok vuruları, PVC, Wolff-Parkinson-White sendromu, miyokardiyal iskemi ve miyokardiyal injuri. Yapay sinir ağları (geriye yayılma algoritması, SOM algoritması) kullanılarak %97.77'nin üzerinde bir başarım sağlanmıştır. Kayıtlar MIT-BIH veri tabanından alınmıştır. Dalgacık dönüşümü ile EKG içerisindeki belli başlı dalgaların başlangıç, bitiş ve tepe pozisyonları belirlenmiş ve bu bilgilerden 12 adet öznitelik (genlik, süre, alan, vs.) belirlenerek 12 boyutlu öznitelik vektörleri oluşturulmuştur.

2.5 Frekans ve Dalgacık Dönüşümleri ile Özniteliklerin Çıkartılması

Öznitelik çıkartma işlemi ile sınıfların öznitelik uzayında farklı bölgelerde öbekleşmesi sağlanır. Öznitelik çıkartma işleminin performansı, öbeklerin çapına ve öbeklerin birbirinden uzaklığına bağlıdır. Karar verme mekanizması içinde minimum öbek çapı ve maksimum öbek uzaklığını veren öznitelik çıkartma işlemi bulunmalıdır. Özniteliklerin sınıfları birbirinden ayırma yeteneği sayesinde karar verme mekanizması içine genelleme özelliği katılır. Bu özellik sayesinde eğitim kümesi içinde bulunmayan vektörler de doğru olarak sınıflanır. Seçilen özniteliklerin mümkün olduğu kadar basit işlemlerle oluşturulması, sınıflama ve eğitim zamanlarının küçülmesine böylece algoritmanın hızlı çalışmasına yol açacaktır.

Sınıfları ayırdedici özniteliklerin bulunması işlemi, seçilen problemin yapısına bağlıdır. Az sayıda örnek, sınıfların sınırlarının doğru olarak belirlenmesini zorlaştıracaktır. Ayrıca sınıfları ayırdedici özniteliklerin tam olarak belirlenememesi, öznitelik uzayında sınıfların saçılmasına sebep olmaktadır. Öznitelik uzayındaki karmaşık sınıf dağılımlarının sınırları, gelişmiş sınıflayıcı yapıları kullanılarak

belirlenmeye çalışılır. Çalışma içersinde sınıflayıcı olarak yapay sinir ağlarının kullanılması planlanmıştır. Böylece, karmaşık sınıf dağılımları yapay sinir ağları ile temsil edilebilecektir.

YSA'ların, bir eğitim kümesindeki vektörleri kullanarak saçılan sınıfları en iyi temsil edecek ağırlıkları kendi kendine öğrenmesi ve genelleme özelliğini kullanarak eğitim kümesine olan bağımlılığın etkisini azaltması, sınıflayıcı olarak seçilmelerinin en önemli sebepleridir.

2.5.1 Diverjans Analizi

Tez içinde öznitelik uzayında sınıfların saçılımını ve sınıflar arası uzaklığı incelemek üzere diverjans hesabı kullanılmaktadır. Eşitlik 2.1'de diverjans hesabında kullanılan denklem verilmiştir.

$$W_i^j = \sum_t (\underline{\beta}_i^{tj} - \underline{\hat{\mu}}_i^j) \cdot (\underline{\beta}_i^{tj} - \underline{\hat{\mu}}_i^j)^T \qquad j=1,2,\ldots,K; \quad i=1,2,\ldots,n$$

$$\underline{\hat{\mu}}_i = \sum_{k=1}^{K} \underline{\hat{\mu}}_i^k \qquad W_i = \sum_{k=1}^{K} W_i^k$$

$$B_i = \sum_{k=1}^{K} (\underline{\hat{\mu}}_i - \underline{\hat{\mu}}_i^k) \cdot (\underline{\hat{\mu}}_i - \underline{\hat{\mu}}_i^k)^T$$

$$D_i = \mathrm{tr}((W_i)^{-1} B_i) \qquad (2.1)$$

$\underline{\beta}_i^{tj}$, j-inci sınıfa ait t-inci i boyutlu öznitelik vektörünü; $\underline{\hat{\mu}}_i^j$, j-inci sınıfa ait i boyutlu öznitelik vektörlerinin ortalamasını; W_i^j, j-inci sınıfın sınıf-içi saçılım matrisini; B_i, sınıflar-arası saçılım matrisini; K, sınıf sayısını; D_i, i-boyutta diverjans değerini; tr(.) ise bölüm sonucu ortaya çıkan matrise iz (trace) işleminin uygulanmasını ifade etmektedir.

Diverjans değeri, seçilen öznitelikler ile oluşturulan vektörlerin saçılımı hakkında bir bilgi vermektedir. Küçük diverjans değerleri, vektörlerin öznitelik uzayında saçıldığının; büyük değerler ise vektörlerin bu uzayda öbekleştiğinin bir göstergesidir. Vektörlerin öznitelik uzayında saçılması, sınıfların kapladığı hacmi

karmaşıklaştıracak ve sınıflama performansının düşmesine sebep olacaktır. Bu nedenle vektörlerin sınıf içi dağılımını küçük, sınıflar arası uzaklığını büyük tutacak (yani diverjans değerini büyük yapacak) en iyi öznitelikler araştırılır. Çalışma içersinde diverjans analizi, iki farklı öznitelik çıkartma işleminin karşılaştırılmasında ve en iyi özniteliklerin belirlenmesinde kullanılmaktadır.

Dinamik programlama ile vektörlerin boyutu 1'den n sayısına kadar adım adım artırılır. Her adımda en iyi öznitelikler diverjans hesaplanarak bulunur. Çalışma içersinde her iki yöntemde de n sayısı 15 olarak seçilmiştir. Vektör boyutunu artırmak diverjans değerini büyütecektir. Dolayısıyla büyük diverjans değerine sahip dağılımlar için daha iyi sınıflama başarımı elde edilecektir. Ancak, boyut arttığı için hesap yükünde de bir artış gözlenecektir. Dalgacık ve frekans temelli yaklaşım ile elde edilen öznitelikler üzerinde 40 boyuta kadar diverjans değerleri analiz edilmiş ve 15 boyuttan sonra değerlerde fazla bir artış olmadığı, eğrinin bir asimptota doğru gittiği gözlenmiştir.

2.5.2 Frekans Dönüşümü

Mühendislik açısından incelendiğinde pratikte karşılaşılan birçok işaret, zaman domeni işaretidir. Yani ölçülen büyüklük zamanın bir fonksiyonudur. Çoğu durumda bu ham işaretten yeterli bilgi elde edilemez. İşaret, matematiksel bir dönüşüm uygulanarak farklı bir domene taşınır ve bu domende işareti temsil eden bileşenlerden bilgi sağlanır. Örneğin elektrik mühendisliğinde sıkça kullandığımız Fourier dönüşümü (FD) ile işaretin frekans spektrumu (frekans bileşenleri) elde edilir. Zaman domeninde saklı bilgi, frekans domeninde açığa çıkarılmış olur.

Zaman domenindeki ham işaretin frekans içeriğini belirlemek için kullanılan Fourier dönüşümü aşağıdaki iki eşitlik ile ifade edilmektedir:

$$X(f) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j2\pi ft} \cdot dt$$
$$x(t) = \int_{-\infty}^{\infty} X(f) \cdot e^{j2\pi ft} \cdot df \qquad (2.2)$$

Fourier dönüşümü ile işaret, farklı frekanslara sahip kompleks üstel fonksiyonlara ayrıştırılır. Denklemlerde görülen t, zamanı; f ise frekansı belirtmektedir. x, zaman domenindeki işareti, X ise frekans domenindeki işareti göstermektedir. 2.2

eşitliğinde, üstte x(t)'nin Fourier dönüşümü, altta ise ters Fourier dönüşümü gösterilmektedir.

Denklem 2.2 incelenecek olunursa, x(t) işareti belirli bir f frekansındaki üstel bir terim ile çarpılmış ve çarpımın eksi sonsuzdan artı sonsuza tüm zaman üzerinden integrali alınmıştır. Dikkat edilirse, f frekanslı bileşen zamanın hangi anında ortaya çıkarsa çıksın integrasyona etkisi aynı olacaktır. f frekanslı bileşenin t_1 ya da t_2 anında ortaya çıkması integrasyon sonucunu değiştirmeyecektir. Fourier dönüşümü yalnızca belirli bir frekans bileşeninin var olup olmadığını belirtmektedir (FD ile işaretin sadece spektral içeriği elde edilir).

Çalışma içersinde ayrık Fourier dönüşümleri kullanılmaktadır. Ayrık Fourier dönüşümleri de ±sonsuz aralığında tanımlıdır. İşaretin sabit bir pencere içerisinde tanımlanması durumunda aşağıdaki ifadeler elde edilir.

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j2\pi kn/N}$$

$$x(n) = \frac{1}{N} \cdot \sum_{k=0}^{N-1} X(k) \cdot e^{j2\pi kn/N} \qquad k, n \in Z,\ N=\text{pencere genişliği} \qquad (2.3)$$

Denklemlerde görülen x(n), ayrık zaman domeni işareti, X(k) ise ayrık frekans domeni işaretidir. Büyük pencere seçiminde frekans ekseni daha iyi çözünürlük ile tanımlanırken, bu frekans bileşenlerinin zamanda temsilleri kötüleşir. Pencerenin küçük seçilmesi ise frekansta kötü bir çözünürlük verirken zamanda temsillerinin daha iyi olmasını sağlar.

Ayrık Fourier katsayılarını hesaplamak için kullanılan pencere, 360 Hz'te örneklenmiş 256 veriden oluşmaktadır. Örnek sayısı, pencere içinde sadece bir EKG periyodu bulunacak şekilde 256 olarak seçilmiştir. Bu veriler, R tepesi pencerenin ortasında bulunacak şekilde ayarlanır.

Çalışmada eğitim kümesi 256 boyutlu 500 adet vektörden meydana gelmektedir. Vektörlere önce normalizasyon işlemi daha sonra frekans dönüşümü uygulanır. İşaret 256 veriden oluştuğu için frekans spektrumunda 128 bileşen bulunmaktadır. Ancak bunlardan 40 tanesinin anlamlı olduğu, geri kalan bileşenlerin ise sıfıra çok yakın

olduğu gözlenmiştir. Boyutu daha da indirgemek için yapılan analizde dinamik programlama ile diverjans hesabı birlikte kullanılmıştır. Dinamik programlama ile vektörlerin boyutu 1'den 15'e kadar adım adım artırılırken diverjans hesabı ile vektörlerin dağılımı hakkında bilgi elde edilmiştir. Bölüm 5 içersinde en anlamlı 15 bileşenin frekans spektrumundaki yerleri ve modül genlik dağılımları hakkında ayrıntılı bilgi verilecektir.

2.5.3 Dalgacık Dönüşümü

Çalışma içerisinde ayrık dalgacık dönüşümü (ADD) kullanılmıştır. Ancak temel kavramların anlaşılabilmesi için aşağıda sürekli dalgacık dönüşümü (SDD) hakkında kısa bir bilgi verilecektir [36, 37].

Kısa-zaman Fourier dönüşümünde (KZFD) durağan olmayan işaret, zamanda durağan kabul edilebilecek küçük parçalara bölünür. FD'den farklı olarak, işarete dar pencerelerden bakılır ve pencere içinde kalan işaretin durağan olduğu varsayılır. Aşağıda KZFD'nin ifadesi görülmektedir:

$$KZFD(\tau,f) = \int_{-\infty}^{\infty} \left[x(t)\cdot w^*(t-\tau)\right]\cdot e^{-j2\pi ft}\,dt \qquad (2.4)$$

x(t), orijinal işareti; w(t), pencere fonksiyonunu ve *, kompleks konjugeyi göstermektedir. f, frekans; τ ise zamanda öteleme miktarıdır. Denklemden görüldüğü gibi KZFD, bir pencere fonksiyonu ile çarpılan x(t)'nin FD'den oluşturulmaktadır. Her τ ve f için yeni bir KZFD katsayısı hesaplanır. FD sadece frekansın bir fonksiyonu iken, KZFD hem frekansın hem de zamanın bir fonksiyonudur ve dönüşüm bu haliyle iki boyutludur (genlik de gözönüne alınırsa boyut üç olacaktır).

İşaretin zaman-frekans temsili elde edilmesine rağmen, seçilen pencerenin genişliği dönüşümün etkinliğinde önemli rol oynamaktadır. KZFD'de pencere genişliği ile ilişkili bir çözünürlük problemi bulunmaktadır.

Pencere genişliği durağanlık varsayımını geçerli kılacak kadar dar olmalıdır. Dar bir pencere seçilmesi durumunda hem bu varsayım geçerliliğini koruyacak hem de FD'de sağlanamayan *zamanda çözünürlük* iyileşecektir. Fakat dar bir pencere seçilmesi durumunda kötü bir frekans çözünürlüğü elde edilir. Pencere genişledikçe

frekans çözünürlüğü artar; ancak zamanda çözünürlük azalır. Sonuçta işarete KZFD'yi uygulamadan önce bir ikilemle karşılaşılır: Zamanda ya da frekansta çözünürlüğün sağlanması.

KZFD, tüm zamanlarda sabit çözünürlük verdiğinden KZFD'nin çözünürlük ile ilgili problemlerini gidermek üzere zamanda değişken çözünürlük veren dalgacık dönüşümü geliştirilmiştir. Dalgacık dönüşümüne frekans cevabı zamanla değişen durağan olmayan işaretlerin analizinde ihtiyaç duyulmaktadır.

Dalgacık analizi KZFD'ye benzer şekilde yapılır: İşaret, KZFD'de bir pencere fonksiyonu ile çarpılırken, dalgacık dönüşümünde *dalgacık* olarak adlandırılan bir fonksiyonla çarpılır. SDD aşağıdaki şekilde ifade edilir:

$$SDD_x^{\psi}(\tau,s) = \Psi_x^{\psi}(\tau,s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t) \cdot \psi^*\left(\frac{t-\tau}{s}\right) dt \qquad (2.5)$$

Görüldüğü gibi SDD, τ ve s değişkenlerinin (öteleme ve ölçek parametreleri) bir fonksiyonudur. $\psi(t)$ dönüşüm fonksiyonudur ve *ana dalgacık* olarak adlandırılır. Dönüşümde kullanılan farklı genişliğe sahip diğer pencere fonksiyonları ana dalgacıktan ölçekleme yoluyla türetilir. Öteleme terimi, KZFD'de rastlanılan şekliyle pencerenin yerini ifade etmektedir. Pencere, işaret üzerinde gezdirilir. Dönüşümden zaman bilgisi öteleme ile sağlanır. KZFD'deki gibi bir frekans parametresi yoktur; bunun yerini 1/frekans olarak tanımlanan ölçek parametresi (s) almıştır.

Ölçek değeri büyük ise (düşük frekanslar) işaret hakkında global bir bilgi elde edilir. Ölçek değeri küçük ise (yüksek frekanslar) işaretteki ayrıntılar (kısa süren değişimler) yakalanır. Ölçek parametresinin değeri değiştirilerek, ana dalgacığın sıkıştırılması ya da genişletilmesiyle SDD'de kullanılan pencereler elde edilir. Morlet, Daubechies ve Meksika şapkası dalgacıkları pencere fonksiyonu olarak kullanılan fonksiyonlara örnektir.

2.5.3.1 Ayrık Dalgacık Dönüşümü

Ayrık dalgacık dönüşümü (ADD) hesap yükünü azaltmasının yanısıra orijinal işaretin analiz ve sentezi için yeterli bilgiyi de sağlamaktadır.

Ayrık dalgacık dönüşümünde temel düşünce sürekli dalgacık dönüşümündekinin aynıdır. Sayısal filtreleme teknikleri kullanılarak sayısal işaretin zaman-ölçek temsili elde edilmektedir. SDD farklı ölçeklerdeki dalgacık ile işaret arasındaki ilişkiyi (korelasyonu) belirtmektedir. Burada benzerlik ölçütü ölçektir (ya da frekans). SDD, analiz penceresinin ölçeği değiştirilerek ve bu pencere zamanda kaydırılarak, işaretle pencere çarpılıp tüm zaman üzerinden integrali alınarak hesaplanır.

Ayrık durumda EKG işaretini farklı ölçeklerde analiz etmek için farklı kesim frekanslarına sahip filtreler kullanılmaktadır [38, 39]. İşaretteki yüksek frekanslı değişimleri analiz etmek için işaret yüksek geçiren filtreler serisinden, alçak frekanslı değişimleri analiz etmek için ise alçak geçiren filtreler serisinden geçirilir.

İşaretteki ayrıntı bilgisinin miktarının ölçütü olan işaret çözünürlüğü, filtreleme işlemi ile değiştirilmektedir. Üst-örnekleme ve alt-örnekleme işlemleri ile de ölçek değiştirilmektedir. Alt-örnekleme, örnekleme oranının düşürülmesine ya da işaretten bazı örnekleri atmaya karşılık gelmektedir. Üst-örnekleme ise işarete yeni örnekler ilave edilerek işaretin örnekleme oranının artırılmasına karşılık gelmektedir.

ADD katsayıları genellikle bir düzlem üzerindeki SDD'den örneklenir. Yani, $s_0=2$ ve $\tau_0=1$ olmak üzere, $s=2^j$ ve $\tau=k.2^j$ ($s=s_0^j$ ve $\tau=k.s_0^j.\tau_0$, $j\in Z$) alındığında ikili (dyadic) dalgacık dönüşümü elde edilir.

Ayrık zaman işaretini (x(n), n tamsayı) darbe cevabı h(n) olan yarım bantlı sayısal alçak geçiren filtreden geçirerek işlemler başlatılır.

$$x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k).h(n-k) \tag{2.6}$$

Yarım bantlı alçak geçiren filtre işaretteki en yüksek frekansın yarısından büyük olan tüm frekansları yok eder. Örneğin bir işaret en fazla 1000Hz'lik bileşene sahipse, yarım bantlı alçak geçiren filtreleme ile 500Hz'in üzerindeki tüm frekanslar yok edilir. İşaretteki en yüksek frekans bileşeni filtreleme işleminden sonra yarıya düştüğünden, Nyquist kriteri uyarınca, işaretteki örneklerin yarısı atılabilir. Böylece işaret 2 faktörüyle alt-örneklenir ve yarı sayıda noktaya (örneğe) sahip olur. Böylece işaretin ölçeği 2 katına çıkmıştır.

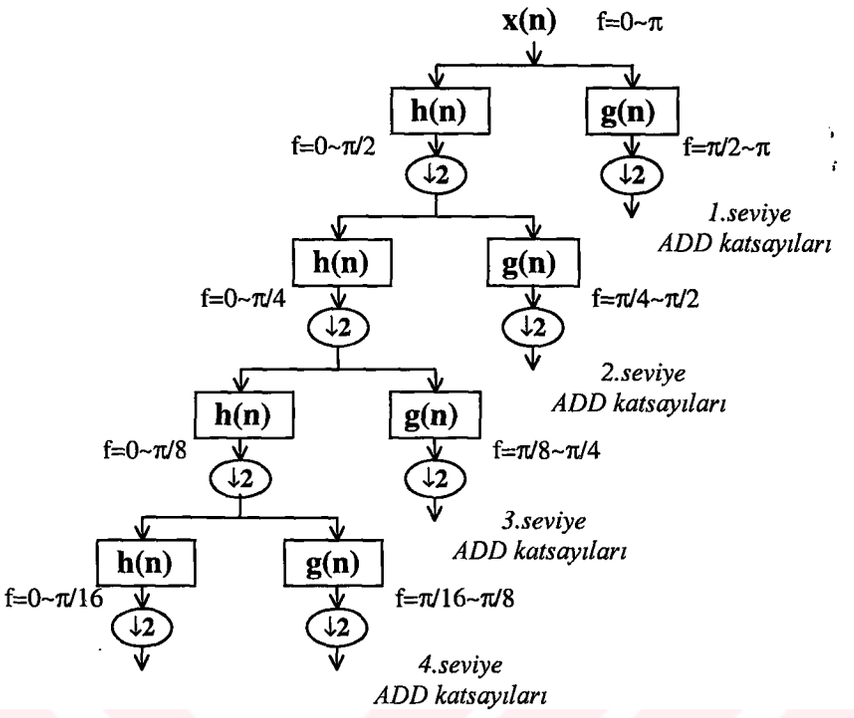
Oysaki alçak geçiren filtreleme ile yüksek frekans bilgisi yok edilmiş ama ölçek değişmeden kalmıştı. Ölçek sadece alt-örnekleme ile değişmektedir. Çözünürlük ise işaretteki bilgi miktarıyla ilişkili olduğundan filtreleme işlemlerinden etkilenir. Yarım bantlı alçak geçiren filtreleme, frekansların yarısını yok ederek bilginin de yarısının kaybolmasına neden olur. Böylece filtreleme sonrasında çözünürlük yarıya düşer. Ancak dikkat edilirse, filtrelemenin ardından alt-örnekleme işlemi çözünürlüğü etkilememektedir. Çünkü işaretin spektral bileşenlerinin yarısını çıkarmak zaten örneklerin de yarısını gereksiz kılar. Yarı sayıda örnekten bilgi kaybına uğramadan kurtulmak mümkündür. Özetle alçak geçiren filtreleme çözünürlüğü yarıya düşürürken ölçeği değiştirmez. Yarı sayıda örneğe artık ihtiyaç olmadığından işaret daha sonra 2 ile alt-örneklenir. Böylece ölçek 2 katına çıkar. Anlatılanları matematiksel olarak şöyle ifade edebiliriz;

$$y(n) = \sum_{k=-\infty}^{\infty} h(k).x(2n-k) \tag{2.7}$$

ADD'nin nasıl hesaplandığı incelersek; ADD işareti, kaba bir yaklaşık işarete ve ayrıntı (detay) işaretine ayrıştırarak işareti farklı frekans bantlarında farklı çözünürlüklerde analiz eder. ADD iki fonksiyon kümesi kullanır: sırasıyla alçak geçiren ve yüksek geçiren filtrelere karşılık gelen *ölçek* fonksiyonu ve *dalgacık* fonksiyonu.

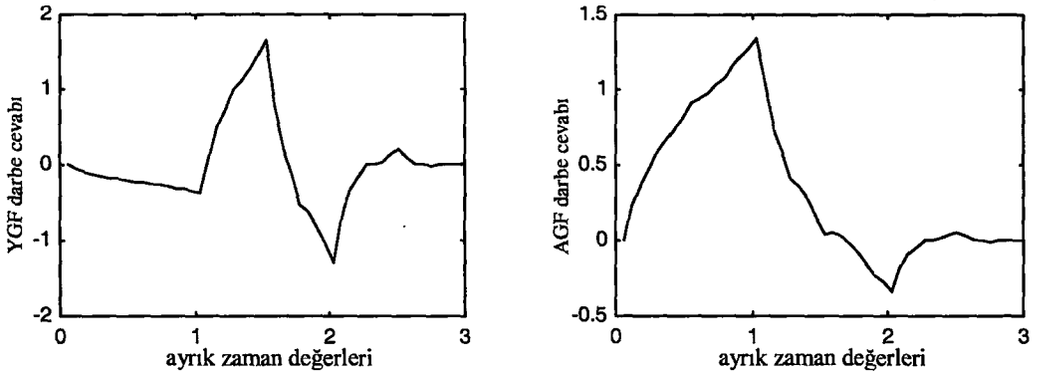
İşarette baskın olan frekanslar ait oldukları frekans bandı bölgesinde yüksek genlik gösterirler. Bunların dışında kalan frekans bandlarını saklamak gereksizdir. Bu nedenle orijinal işareti belirli bir hata ile temsil etmek için yüksek genlikli bileşenlerin bulunduğu frekans bantlarına karşılık gelen seviyelerdeki ADD katsayılarını kullanmak yeterli olur. Çalışmada anlamlı bileşenleri araştırma işlemi, diverjans hesabı ile dinamik programlama kullanılarak gerçeklenmiştir.

Şekil 2.12'de ayrık dalgacık dönüşümü kullanılarak elde edilen katsayılar gösterilmektedir. Alçak geçiren filtre çıkışındaki işaretin alt-örneklenmesiyle elde edilen işaret *yaklaşıklık katsayıları* olarak adlandırılır. Yüksek geçiren filtre çıkışındaki işaretin alt-örneklenmesiyle elde edilen işaret ise *ayrıntı (detay) katsayıları* olarak adlandırılır.



Şekil 2.12 Dalgacık ağacı

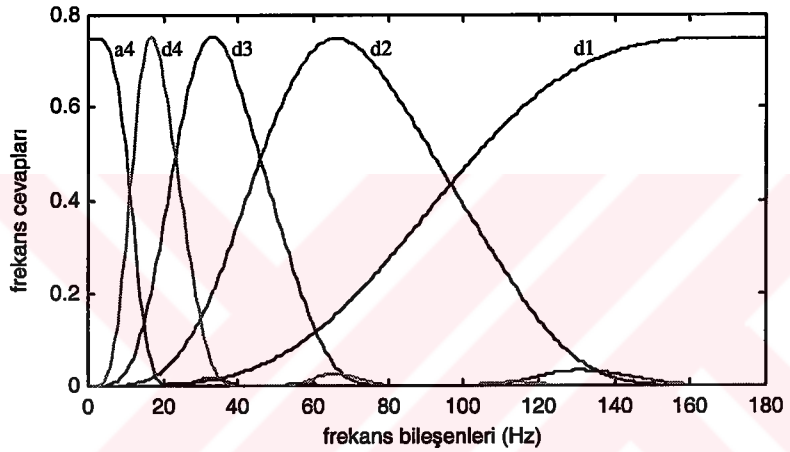
Şekil 2.12'deki h(n) ve g(n) filtreleri için Daubechies dalgacık filtreleri kullanılmıştır [37]. Şekil 2.13'te g(n) yüksek geçiren filtre (YGF) ile ilişkili ana dalgacık fonksiyonu, h(n) alçak geçiren filtre (AGF) ile ilişkili ölçek fonksiyonu gösterilmektedir.



Şekil 2.13 a) Daubechies-2 dalgacık fonsiyonu b) ölçek fonksiyonu

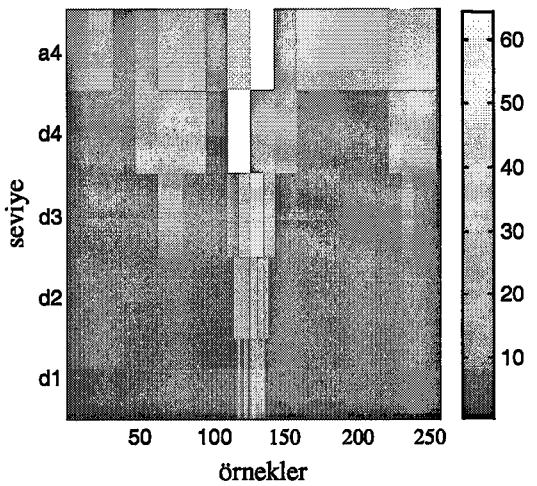
Daubechies tarafından geliştirilen dalgacıklar (Daubechies-M, M=1,2,....), ortonormaldir. Dalgacıkların ortonormal özelliğe sahip oluşu orijinal işarete geri dönülebilmesini (sıkıştırma tekniklerinde) sağlamaktadır.

Şekil 2.14'te 360 Hz'de örneklenmiş EKG işaretini analiz için kullanılan bantlar gösterilmektedir. Şekil üzerinde sağdan sola doğru yerleştirilen frekans bantları, şekil 2.12'deki dalgacık ağacında 1. seviyeden 4. seviyeye kadar gösterilen çıkışlara karşılık gelmektedir. Çalışma içersinde Daubechies-2 (M=2) dalgacıkları kullanılmaktadır. M sayısı, temel filtreyi tanımlamaktadır. Daubechies-M dalgacığı ile ilişkili filtreler, 2×M (çalışmada 4) katsayıdan meydana gelmektedir. Şekilden filtrelerin frekans bantlarının yumuşak geçişli olduğu gözlenmektedir. M değeri büyüdükçe filtreler, kesim frekanslarında daha keskin geçişler yapar; ancak filtre çıkış değerini bulmak için daha fazla hesaplama yapılmaktadır.



Şekil 2.14 Daubechies-2 dalgacıklarına ait filtrelerin frekans spektrumları

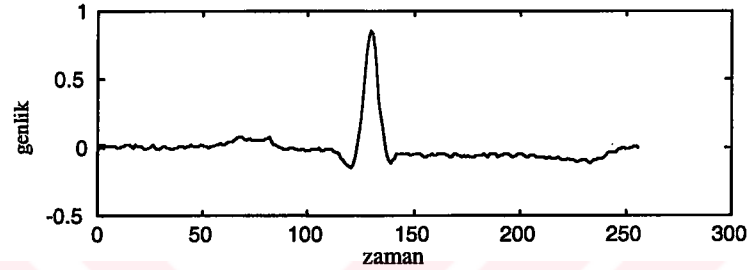
Dalgacık analizinde ölçeklere ait dalgacık katsayıları, düşey eksen ölçeği yatay eksen zamanı göstermek üzere bir düzlem üzerinde gösterilmektedir. Şekil 2.15'te normal bir EKG işaretine ait dalgacık düzlemi gösterilmektedir.



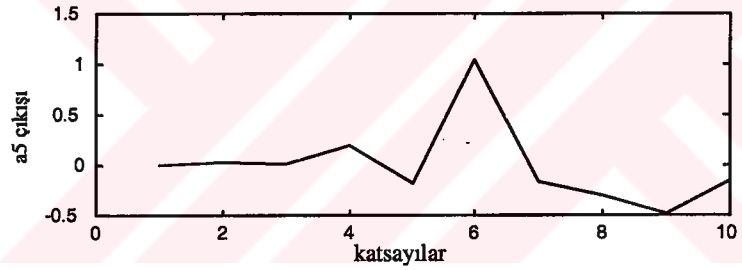
Şekil 2.15 Normal EKG vurusuna ait dalgacık düzlemi

Düzlemdeki renklendirme bağıl genlik değerlerini ifade etmek için kullanılmaktadır. Düzlemde gösterilen yataydaki (belirli bir ölçeğe ait) işaretler, şekil 2.12'deki dalgacık ağacında ifade edilen filtre çıkışlarını göstermektedir.

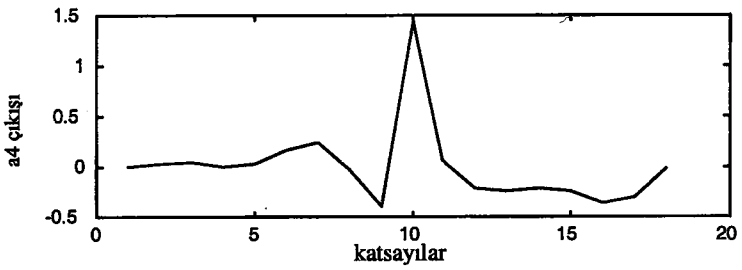
10 sınıfa ait EKG vuruları için yapılan analizler, 4. seviyeden sonra elde edilen katsayıların orijinal işaretleri yeteri kadar iyi temsil edemediğini göstermektedir. Şekil 2.16.a-h'da normal bir EKG vurusu için 5. seviyeye kadar oluşturulan dalgacık ayrıntı ve yaklaşıklık katsayıları ayrı ayrı gösterilmiştir.



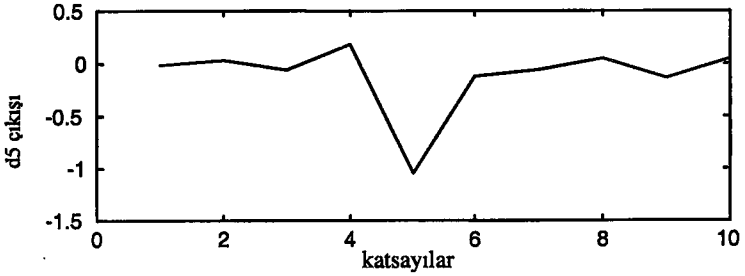
2.16.a Orijinal EKG işareti



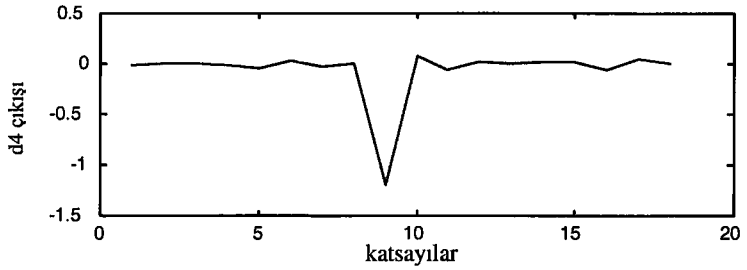
2.16.b Beşinci seviye dalgacık yaklaşıklık katsayıları



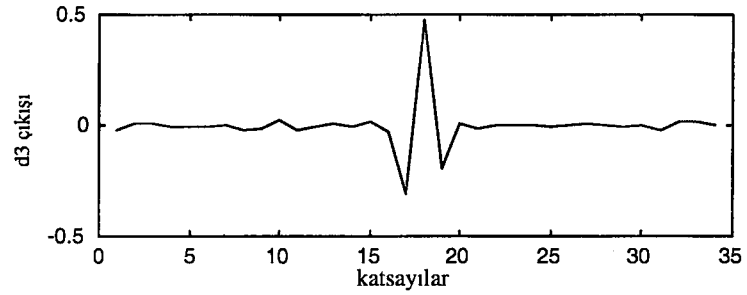
2.16.c Dördüncü seviye dalgacık yaklaşıklık katsayıları



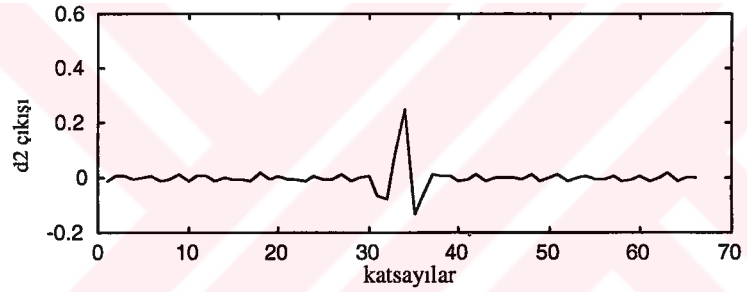
2.16.d Beşinci seviye dalgacık ayrıntı katsayıları



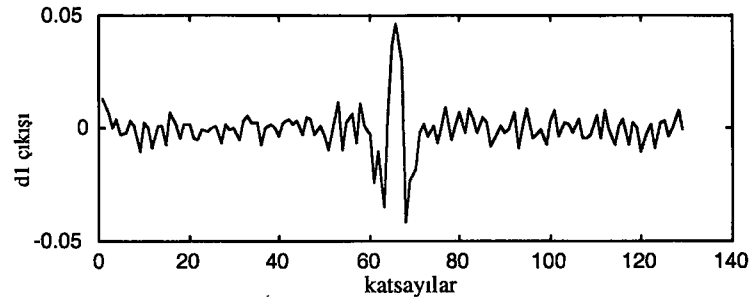
2.16.e Dördüncü seviye dalgacık ayrıntı katsayıları



2.16.f Üçüncü seviye dalgacık ayrıntı katsayıları



2.16.g İkinci seviye dalgacık ayrıntı katsayıları



2.16.h Birinci seviye dalgacık ayrıntı katsayıları

Çalışmada 256 boyutlu tüm vektörlere önce normalizasyon işlemi daha sonra 4. seviyeye kadar olan katsayıları elde etmek için dalgacık dönüşümü uygulanır. Ayrıca 4. seviye dalgacık yaklaşıklık katsayılarının otokorelasyonları da hesaplanır. Bulunan tüm öznitelikler içerisinde en iyi 15 bileşen, diverjans analizi ile araştırılır. Böylece EKG işaretini en iyi karakterize eden vektör bileşenleri belirlenmiş olur. Bölüm 5 içersinde en anlamlı 15 bileşenin dalgacık düzlemindeki yeri ve sınıflar içindeki dağılımı hakkında ayrıntılı bilgi verilmiştir.

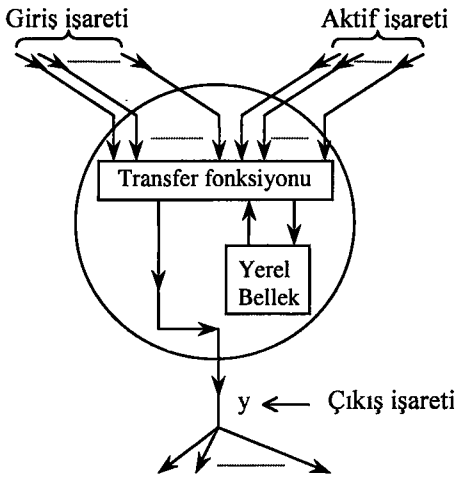
3. YAPAY SİNİR AĞLARI

3.1 Yapay Sinir Ağının Tanımı

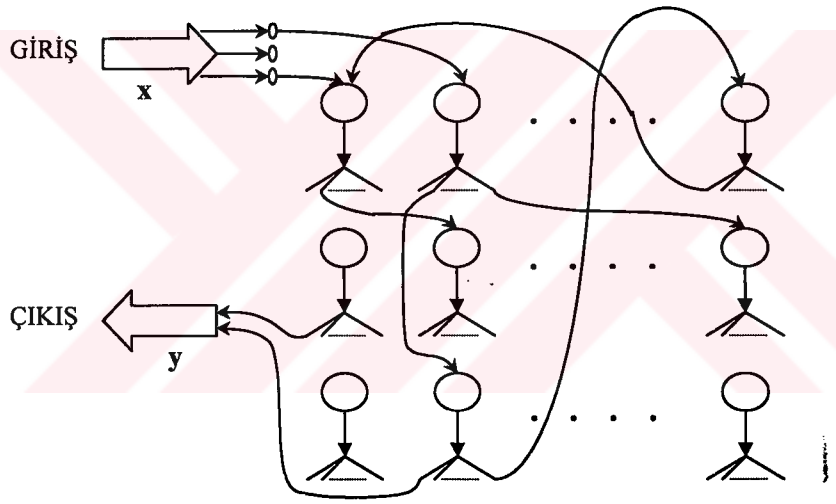
Yapay sinir ağı, aşağıdaki tanımlamaları ve sınırlamaları içeren paralel bilgi işleme özelliğine sahip yönlü bir graftır [40].

a) Yönlü grafın düğümleri *işlem elemanı* olarak tanımlanır.
b) Yönlü dallar bağlantılara karşılık düşer ve tek yönlü işaret iletim yolu olarak çalışırlar.
c) Her bir işlem elemanı, belirli bir sayıda giriş bağlantısına sahiptir.
d) Her bir işlem elemanı, belirli bir sayıda çıkış bağlantısına sahiptir. Ancak çıkış işaretlerinin değeri aynı olmalıdır.
e) İşlem elemanları yerel belleklere sahip olabilir.
f) Her işlem elemanı, giriş işaretini ve yerel belleği kullanan bir transfer fonksiyonuna sahiptir. Bu fonksiyon işlem elemanının çıkış değerini oluşturur. Transfer fonksiyonu sürekli veya ayrık olarak çalıştırılabilir. Ayrık çalıştırma modunda fonksiyon, bir aktif işareti ile kontrol edilir.
g) Yönlü bağlantı ve işlem elemanlarının bir araya gelmesi ile oluşan yapıya, yapay sinir ağı (YSA) ismi verilir. Bu yapı, dış çevreden giriş bağlantıları yolu ile işaret alır ve çıkış bağlantıları yolu ile işaret gönderir.

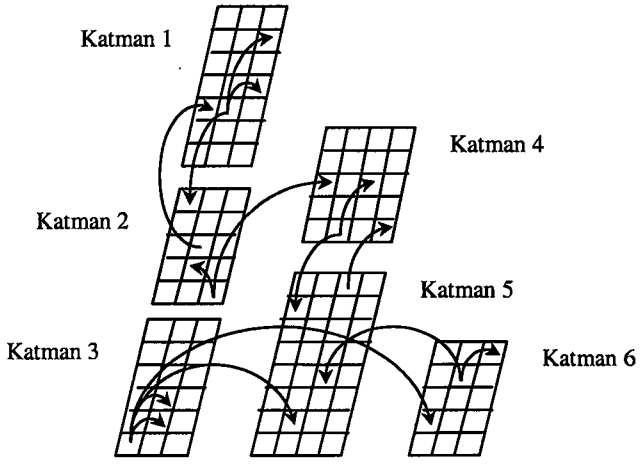
Şekil 3.1'de işlem elemanının iç yapısı gösterilmiştir. İşlem elemanının transfer fonksiyonu, diğer elemanlardan ve yerel belleğinden gelen bilgileri giriş olarak alır. Transfer fonksiyonunun çıkışı, diğer elemanlara gönderilmek üzere kopyalanır. Şekil 3.2'de işlem elemanlarının birbirleriyle yaptığı bağlantılar gösterilmiştir. Yapay sinir ağları, katman olarak isimlendirilen alt kümeler içinde aynı yapıda işlem elemanlarına sahiptir. Bu alt kümeler içindeki tüm işlem elemanları aynı transfer fonksiyonunu kullanır. Şekil 3.3'te altı katmana sahip bir ağ gösterilmiştir. Aynı katman içindeki işlem elemanları birbirleriyle veya diğer katmandaki elemanlar ile bağlantı oluşturabilir.



Şekil 3.1 İşlem elemanının yapısı



Şekil 3.2 Yapay sinir ağı

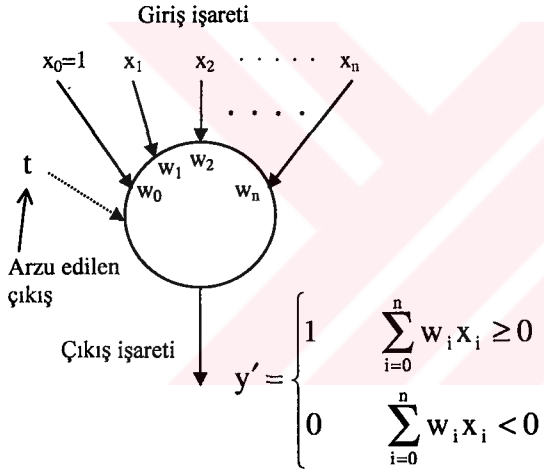


Şekil 3.3 Altı katmanlı bir yapay sinir ağı

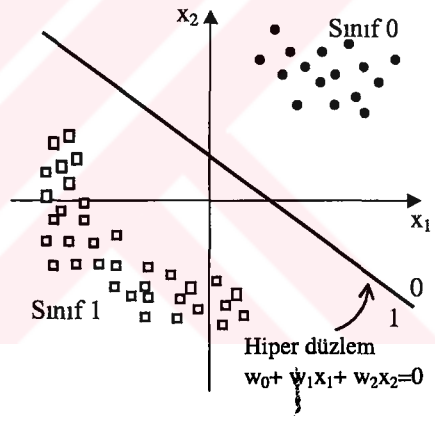
İşlem elemanı, transfer fonksiyonu ve öğrenme ile ilgili kavramaların daha iyi anlaşılması için, yapay sinir ağlarında en temel yapı olan '*perceptron*'u inceleyelim. 'Perceptron', şekil 3.4.a'da gösterilen birden fazla girişi, en az bir çıkışı, bir transfer fonksiyonu ve giriş sayısı kadar yerel bellek elemanı olan en basit yapıda yapay sinir ağıdır. Eşitlik 3.1'de 'perceptron'un matematiksel ifadesi verilmiştir.

$$y = \sum_{i=0}^{n} w_i \cdot x_i \qquad x_0 = 1 \qquad y' = F(y) = \begin{cases} 1 & y \geq 0 \\ 0 & y < 0 \end{cases} \qquad (3.1)$$

Burada x_i, giriş vektörünün i. elemanını; w_i, yerel belleğin i. elemanını; n, boyutu; F(y), ise kırpıcı adı verilen ve lineer olmayan transfer fonksiyonunu temsil etmektedir.



Şekil 3.4.a 'Perceptron'un yapısı



b. 'Perceptron'un eğitimi

Eğitim işlemi ile 'perceptron'daki sadece w_i yerel bellek elemanları değiştirilir. Bu elemanlar çoğu zaman sinir hücresinin *ağırlık vektörü* olarak tanımlanır. Hücrenin ağırlıkları önceden belirlenen amaç ölçütünü sağlaması için bir eğitim algoritması kullanılarak değiştirilir. Öğrenme işlemini, bu amaç ölçütünü en iyi sağlayan w_i ağırlık vektörünü bulmaya doğru adım adım yaklaşma olarak tanımlayabiliriz.

Şekil 3.4.b'de iki boyutlu (n=2) örnek uzayda iki sınıf bulunmaktadır. Daire ve kare şekilleri her iki sınıftan vektörleri temsil etmektedir. Amaç ölçütü, perceptron'un iki sınıfı birbirinden ayırması olarak seçilmiştir. Öncelikle iki sınıftan eşit sayıda vektör alınarak bir eğitim kümesi oluşturulur. Öğrenme algoritması, eğitim kümesinden

alınan sınıf dağılım bilgisini kullanarak, amacı sağlayan optimum w_i değerlerini araştırır. Şekilde görülen düz çizgi, aranılan w_i ağırlık değerleri için oluşan doğruyu göstermektedir. Ancak, amaç ölçütünün her zaman bir düğüm (hücre) ile sağlanması mümkün olmaz; bu ölçüt için çok sayıda düğüm gerekebilir. Ağ, incelenen problem için amaç ölçütünü sağlayabilecek yapıya sahip olmalıdır; aksi taktirde bu ağ için hiçbir eğitim algoritması istenilen sonucu veremeyecektir.

3.2 Yapay Sinir Ağlarında Öğrenme

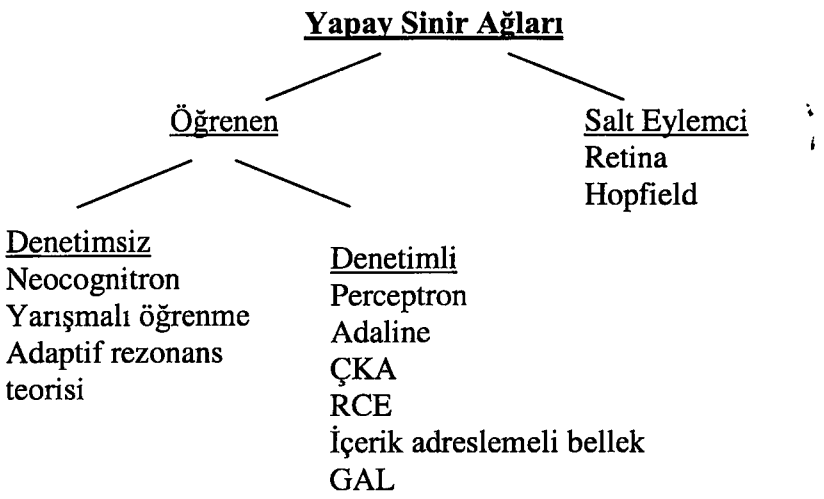
Giriş ve çıkış arasındaki ilişkiyi istenilen şekilde oluşturmak için değişik eğitim algoritmaları kullanılır. Literatür incelendiğinde her ağ için farklı bir eğitim algoritması kullanıldığı gözlenmektedir. Bu algoritmalar, öğrenme yöntemlerine göre aşağıdaki gibi sınıflandırılabilir [41].

a) Salt Eylemci: İç yapı çevre ile etkileşim altında değişmez.
b) Öğrenen
  b.1) Denetimsiz öğrenme
  b.2) Denetimli öğrenme
    b.2.1) Eğitici doğru sonucu söyler
    b.2.2) Eğitici sadece ödül–ceza verir

Genel olarak bir yapay sinir ağının öğrenmesi, çevreden ve varsa eğiticiden aldığı girişler uyarınca (çoğunlukla belirli bir amacı daha iyi yapabilmek için) giriş-çıkış işlevini, iç yapısını uyarlayarak değiştirmesi olarak tanımlanır. İç yapıdaki değişiklikler; yeni bağlantılar oluşturmak, varolan bağlantıları yok etmek veya varolan bağlantıların ağırlıklarını değiştirmek şeklinde gerçekleşir. Şekil 3.5'te bazı bilinen ağların, öğrenme yöntemlerine göre sınıflandırılması verilmiştir.

3.3 Denetimli Yapay Sinir Ağları

Denetimli öğrenme yöntemlerinde ağ, kendi çıkışı ile arzu edilen çıkış arasındaki farkı azaltacak şekilde parametrelerini değiştirir. Ağın eğitim kümesi her sınıftan aynı sayıda rastgele seçilmiş vektörlerden oluşturulur. ÇKA, GAL ve RCE ağları denetimli öğrenme yöntemini kullanan ağlara verilen en temel örneklerdir.



Şekil 3.5 Eğitim yöntemlerine göre yapay sinir ağları

3.3.1 Çok Katmanlı Ağ

Çok katmanlı ağ, giriş ve çıkış düğümleri arasında birden fazla katmana sahip ileri beslemeli bir ağdır. İlave katmanlar, giriş ve çıkıştaki düğümlerle direkt bağlantısı olmayan saklı düğümlerden meydana gelir. Çok katmanlı ağın yeteneği, katmanlı bir yapıya sahip olmasından ve düğüm çıkışlarında lineer olmayan bir transfer fonksiyonu kullanılmasından kaynaklanmaktadır.

3.3.1.1 Çok Katmanlı Ağın Yapısı

Şekil 3.6'da üç katmanlı bir ağ gösterilmiştir. Aşağıda bu yapı içinde kullanılan sembollerin anlamları ve giriş-çıkış düğümleri arasındaki ilişkiler verilmiştir [42].

I^0 ve O^0 , sıfırıncı seviyedeki giriş ve çıkış vektörünü tanımlar. Bu iki vektör giriş katmanında birbirlerine eşittir.

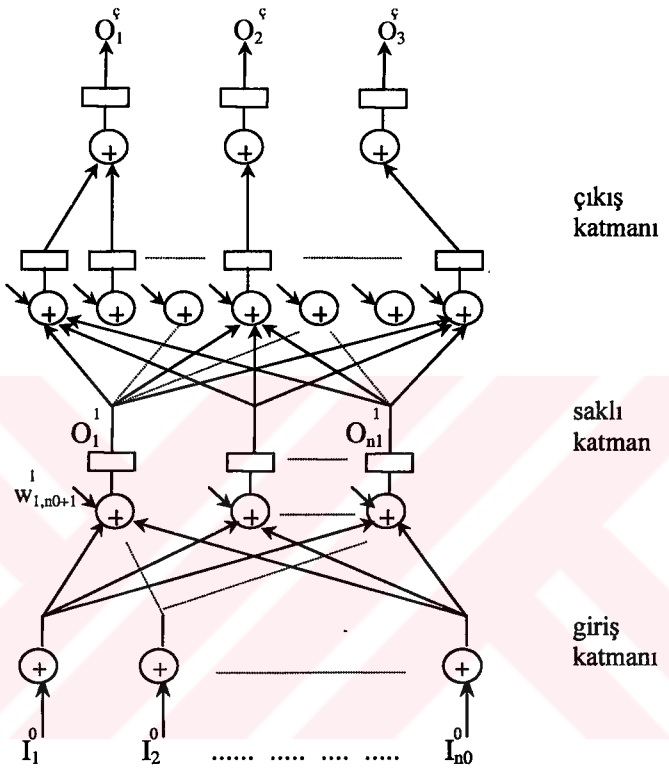
$$\bar{I}^0 = (I_1^0, I_2^0, \ldots\ldots, I_{n0+1}^0) \qquad I_{n0+1}^0 = 1 \qquad \bar{O}^0 = (O_1^0, O_2^0, \ldots\ldots, O_{n0+1}^0) \qquad (3.2)$$

$$\bar{I}^0 = \bar{O}^0$$

n0 değeri ağın girişindeki düğüm sayısını gösterir. Birinci katmandaki giriş ve çıkış arasındaki ilişki denklem 3.3'te verilmiştir.

$$\bar{I}^1 = (I_1^1, I_2^1, \ldots\ldots, I_{n1}^1) \qquad I_j^1 = \sum_{i=1}^{n0+1} w_{ji}^1 \cdot O_i^0 \tag{3.3}$$

w^1_{ji}, sıfırıncı katmandaki i. düğüm ile 1. katmandaki j. düğüm arasındaki bağlantıyı; n1 ise 1. saklı katmandaki düğüm sayısını göstermektedir.



Şekil 3.6 Çok katmanlı ağın yapısı

Giriş katmanının çıkışı,

$$\bar{O}^1 = [F(I_1^1), F(I_2^1), \ldots\ldots, F(I_{n1}^1)]; \qquad O_{n1+1}^1 = 1 \tag{3.4}$$

$$F(x) = \frac{1}{1+e^{-x}}$$

şeklindedir. Çıkış katmanındaki giriş-çıkış ilişkisi ise

$$\bar{I}^{ç} = (I_1^{ç}, I_2^{ç}, \ldots\ldots, I_{nç}^{ç}) \qquad I_j^{ç} = \sum_{i=1}^{nç} w_{ji}^{ç} \cdot O_i^{ç-1}$$

$$\bar{O}^{ç} = [F(I_1^{ç}), F(I_2^{ç}), \ldots\ldots, F(I_{nç}^{ç})] \tag{3.5}$$

şeklindedir. nç, çıkış katmanındaki düğüm sayısını; ç ise çıkış katmanını göstermektedir.

3.3.1.2 Çok Katmanlı Ağın Eğitimi

Ağın eğitimi için *genelleştirilmiş δ-kuralı* (generalized δ-rule) olarak bilinen bir eğitim algoritması anlatılacaktır. Bu öğrenme algoritmasında, eşitlik 3.6'daki amaç ölçütünü minimum yapacak w ağırlık değerleri araştırılmaktadır.

$$E = \frac{1}{2} \cdot \sum_{m=1}^{nç} (t_m - O_m^{ç})^2 \qquad \bar{T} = [t_1, t_2, \ldots\ldots, t_{nç}] \tag{3.6}$$

Denklemde görüken $\bar{T}$ vektörü, arzu edilen çıkışı ifade etmektedir. Önce çıkış düğümleri daha sonra saklı katmandaki düğümler için aşağıdaki ifadeler hesaplanır.

$$\frac{\partial E}{\partial w_{ji}^{ç}} = \frac{\partial E}{\partial O_j^{ç}} \cdot \frac{\partial O_j^{ç}}{\partial w_{ji}^{ç}} = -(t_j - O_j) \cdot \frac{\partial O_j^{ç}}{\partial w_{ji}^{ç}} = -(t_j - O_j^{ç}) \cdot F'(I_j^{ç}) \cdot O_i^{ç-1}$$

$$F(x) = \frac{1}{1+e^{-x}} \qquad F'(x) = F(x) \cdot (1 - F(x)) \tag{3.7}$$

$$\frac{\partial E}{\partial w_{ji}^{ç}} = -(t_j - O_j^{ç}) \cdot O_j^{ç} \cdot (1 - O_j^{ç}) \cdot O_i^{ç-1} \qquad \delta_j^{ç} = (t_j - O_j^{ç}) \cdot O_j^{ç} \cdot (1 - O_j^{ç})$$

$$\Delta w_{ji}^{ç} = \eta \cdot \delta_j^{ç} \cdot O_i^{ç-1} \qquad w_{ji}^{ç}(k+1) = w_{ji}^{ç}(k) + \eta \cdot \delta_j^{ç} \cdot O_i^{ç-1}$$

Saklı katmandaki ağırlık katsayılarını bulmak için eşitlik 3.8'deki ifadeler kullanılır.

$$\frac{\partial E}{\partial w_{ji}^{ç-1}} = \sum_{m=1}^{nç} \left((t_m - O_m^{ç}) \cdot \frac{\partial O_m^{ç}}{\partial w_{ji}^{ç-1}} \right) = -\sum_{m=1}^{nç} \delta_m^{ç} \cdot w_{mj}^{ç} \cdot O_j^{ç-1} \cdot (1 - O_j^{ç-1}) \cdot O_i^{ç-2} = \delta_j^{ç-1} \cdot O_i^{ç-2}$$

$$\delta_j^{ç-1} = O_j^{ç-1} \cdot (1 - O_j^{ç-1}) \cdot \sum_{m=1}^{nç} \delta_m^{ç} \cdot w_{mj}^{ç}$$

$$w_{ji}^{ç-1}(k+1) = w_{ji}^{ç-1}(k) + \eta \cdot \delta_j^{ç-1} \cdot O_i^{ç-2} \qquad (3.8)$$

Aşağıda çok katmanlı ağın eğitim algoritması verilmiştir. Eşik değerleri, girişleri +1 olan ağırlıklar olarak düşünülmüştür.

Adım 1) Tüm ağırlıklara –1 ile +1 arasında rastgele değerler ver. İterasyon sayısını belirle.

Adım 2) Eğitim kümesinden rastgele bir vektör seç. Bu vektör için ağın çıkış cevabını hesapla ve arzu edilen çıkış cevabı $\overline{T}$ 'yi belirle.

Adım 3) Çıkış katmanından başlayarak sırayla geriye, birinci katmana doğru ağırlıkları aşağıdaki denklemleri kullanak değiştir.

$$\text{Giriş vektörü} = \overline{I}^0 = \overline{O}^0$$

$$w_{ji}^{p}(k+1) = w_{ji}^{p}(k) + \eta \cdot \delta_j^{p} \cdot O_i^{p-1}$$

Eğer çıkış katmanındaki ağırlıklar değiştirilecek ise p=ç'dir ve ağırlıklar aşağıdaki şekilde değiştirilir.

$$\delta_j^{ç} = (t_j - O_j^{ç}) \cdot O_j^{ç} \cdot (1 - O_j^{ç})$$

$$w_{ji}^{ç}(k+1) = w_{ji}^{ç}(k) + \eta \cdot \delta_j^{ç} \cdot O_i^{ç-1} \qquad (3.9)$$

Eğer saklı katmandaki ağırlıklar değiştirilecek ise p<ç'dir ve ağırlıklar aşağıdaki şekilde değiştirilir.

$$\delta_j^{p} = O_j^{p} \cdot (1 - O_j^{p}) \cdot \sum_{m=1}^{nç} \delta_m^{p+1} \cdot w_{mj}^{p+1}$$

$$w_{ji}^{p}(k+1) = w_{ji}^{p}(k) + \eta \cdot \delta_j^{p} \cdot O_i^{p-1} \qquad (3.10)$$

Adım 4) İterasyon sayısını azalt. İterasyon sayısı 0'a eşit değilse, adım 2'ye git. Aksi taktirde eğitim algoritmasını sonlandır.

3.3.2 RCE Ağı

RCE ağı, gerçek zaman uygulamalarında kullanılmaktadır. Ağın düğümleri, eğitim kümesinden alınan vektörlerden meydana getirilir. Eğitim algoritması ile ağ içindeki düğümlerin temsil ettiği hiper kürelerin yalnızca yarıçapları giriş vektörüne göre ayarlanır.

Eğitim tamamlandıktan sonra gelen giriş vektörünün sınıfı, içine düştüğü hiper küre tarafından belirlenir. Giriş vektörünün sınıfı, hiper kürenin sınıfı olarak atanır.

3.3.2.1 RCE Ağının Yapısı

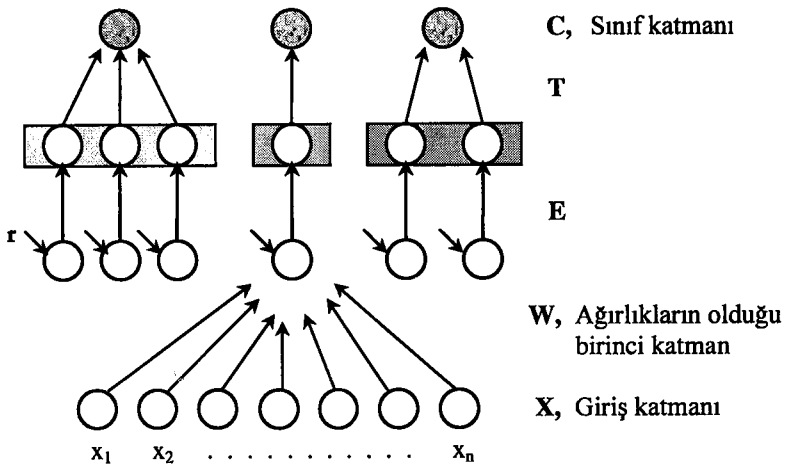
Şekil 3.7'de RCE ağının yapısı gösterilmiştir. RCE ağı, iki katmandan meydana gelir. Ağın içindeki düğümlerin yapısı aşağıdaki denklemlerle ifade edilir [43].

$$D_j = \sum_{i=1}^{n} \left(x_i - w_{ji}(k)\right)^2 \qquad E_j = \begin{cases} 1 & D_j \le r_j^2(k) \\ 0 & \text{aksi taktirde} \end{cases} \qquad (3.11)$$

$$\text{Çıkış katmanı} = C_c = \sum_e E_e \cdot T_{ec}$$

x_i, giriş vektörünün elemanlarını; w_j, j-inci hiper kürenin merkezini; E_e, birinci katmandaki düğümlerin çıkışını; T_{ec} ise sadece 0 veya 1 değerini alan OR'lama işlemini gerçekleyen bağlantı katsayısını; n ise boyutu gösterir.

Bu ifade, öznitelik uzayında kapalı bir hacim oluşturmaktadır ve bu hacim hiper kürelerle temsil edilmektedir.



Şekil 3.7 RCE ağının yapısı

Genel olarak öznitelik uzayındaki bir sınıf, dağılımının yapısına bağlı olarak birden fazla hiper küre kullanılarak temsil edilebilir. Bu nedenle ikinci katman, aynı sınıfı temsil eden hiper kürelerin çıkışlarını lojik olarak OR'lamak için kullanılır. Birinci katmanın çıkışını ikinci katmandaki sınıfı belirleyen düğüme bağlayan ağırlıklara ise, lojik OR işlemini gerçeklemesi için öğrenme algoritması tarafından 1 değeri verilir. Başlangıçta bu katsayılar 0 değerindedir.

3.3.2.2 RCE Ağının Eğitimi

Eğitim algoritması, düğüm sayısını otomatik olarak kendi belirler. Dolayısıyla, eğitim öncesinde düğüm sayısını belirleme ihtiyacı yoktur. Aşağıda RCE ağının eğitim algoritması verilmiştir.

Başlangıç olarak herbir sınıf için bir vektör alınır. Bu vektörün tüm bileşenleri, birinci katmandaki bir düğümün ağırlık katsayıları olarak tanımlanır. Alınan vektörleri kendi sınıflarını temsil eden ağ çıkışlarına bağlamak için ikinci katmandaki ilişkili bağlantı katsayılarına 1 değeri verilir. Alınan vektörlere büyük bir yarıçap değeri verilir. İterasyon sayısı belirlenir.

Adım 1) Eğitim kümesinden rastgele bir vektörü ağa giriş olarak ver.

Adım 2) Giriş vektörü ile düğüm ağırlıkları arasında eşitlik 3.11'deki işlemleri yap. E_j'nin 1 olduğu en az bir j düğümü varsa adım 4'e git.

Adım 3) Birinci katmanda boş olan bir düğüm bul. Giriş vektörünü bu düğümün bağlantı katsayılarına eşitle ve bu düğüm için büyük bir eşik (yarıçap) değeri ver. İkinci katmandaki ilişkili bağlantı katsayısına 1 değerini ver. İterasyon sayısını azalt. İterasyon sayısı sıfıra eşit değilse adım 1'e git. Aksi taktirde eğitim algoritmasını sonlandır.

Adım 4) Giriş vektörünün sınıfı ile içine düştüğü hiper kürelerin sınıfı aynı ise adım 1'e git. Aksi taktirde içine düştüğü hiper kürelerin hepsinin çapını, bir defa daha bu durumla karşılaşıldığında çakışma olmayacak şekilde azalt. Adım 1'e git.

3.3.3 GAL Ağı

GAL ağı [44], öznitelik uzayındaki sınıf sınırlarını en yakın mesafe ölçütüne göre belirler. Giriş vektörü ile ağdaki tüm düğümlere olan mesafeler hesaplanır. Giriş

vektörünün sınıfı, ağdaki düğümlere en yakın mesafede olan düğümün sınıfı olarak belirlenir. Ağın düğüm sayısı, eğitim sırasında ihtiyaca göre otomatik olarak bulunur.

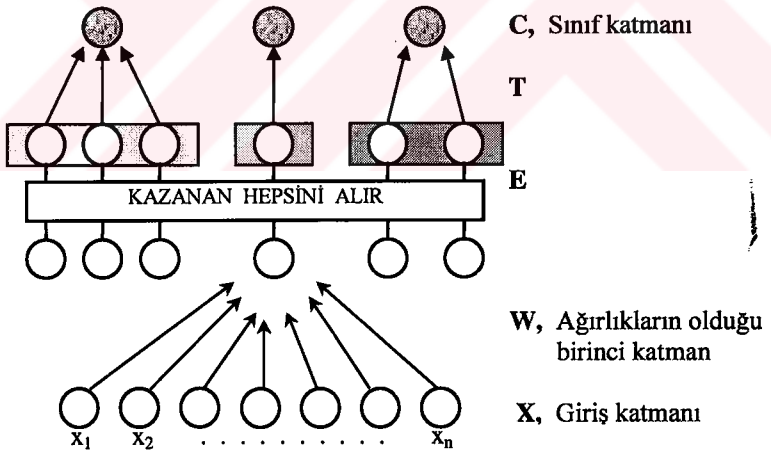
3.3.3.1 GAL Ağının Yapısı

Şekil 3.8'de GAL ağının yapısı gösterilmiştir. GAL ağı, RCE ağı gibi iki katmandan meydana gelir. Ağın içindeki düğümlerin yapısı aşağıdaki denklemle ifade edilir.

$$D_j = \sum_{i=1}^{n} \left(x_i - w_{ji}(k)\right)^2 \qquad E_e = \begin{cases} 1 & D_e = \min_j \{D_j\} \\ 0 & \text{aksi taktirde} \end{cases} \qquad (3.12)$$

$$\text{Çıkış katmanı} = C_c = \sum_e E_e \cdot T_{ec}$$

E_e, birinci katmandaki hücrelerin çıkışını; T_{ec} ise sadece 0 veya 1 değerini alan OR'lama işlemini gerçekleyen bağlantı katsayısını gösterir.



Şekil 3.8 GAL ağının yapısı

İlk katman, düğüm ağırlıkları ile giriş vektörü arasında minimum mesafeyi bulmada kullanılırken; ikinci katman, ağdaki düğümlerin ait oldukları sınıfı tanımlamak için kullanılır. İkinci katmandaki ağırlıklar başlangıçta 0 değerini alır, eğitim sırasında bu bağlantılar 1'lenir. İkinci katman, aynı sınıftan çıkışları lojik olarak OR'lamak için kullanılır.

3.3.3.2 GAL Ağının Eğitimi

GAL ağının en önemli özelliği, düğüm sayısının eğitim sırasında problemin yapısına bağlı olarak kendiliğinden belirlenebilmesidir. GAL ağının yapısı, başlangıçta ağa verilen giriş vektörlerinin sırasına çok bağlıdır. Ağ içinde daha önceden anlamlı olan, ancak ağa yeni düğümler ilavesi ile anlamını yitiren ve bir daha kullanılmayan düğümler oluşmaktadır. Bu düğümler unutma algoritması tarafından ağdan çıkarılır. Unutma algoritmasının amacı, ağdan çıkarıldığı zaman ağın performansını değiştirmeyen düğümleri bulup, bu düğümleri ağdan çıkartmaktır. Aşağıda GAL ağının eğitim algoritması verilmiştir.

Her sınıftan bir vektör alınarak ağın başlangıç düğümleri oluşturulur ve iterasyon sayısı belirlenir.

Adım 1) Eğitim kümesinden rastgele bir vektörü ağa giriş olarak ver.

Adım 2) Giriş vektörü ile ağın düğümleri arasındaki mesafeleri hesapla. Minimum mesafedeki ağ düğümünün sınıfı, giriş vektörünün sınıfı ile aynı değil ise adım 4'e git.

Adım 3) İterasyon sayısını azalt. İterasyon sayısı sıfıra eşit ise öğrenme algoritmasını sonlandır. Aksi taktirde adım 1'e git.

Adım 4) Giriş vektörünü ağa bir düğüm olarak ekle (giriş vektörünün elemanlarını, birinci katmandaki herhangi boş bir çıkışın ağırlıklarına eşitle ve bu çıkış düğümünün ikinci katmandaki ilişkili bağlantısına 1 değerini ver). Adım 1'e git.

Aşağıda GAL ağının unutma algoritması verilmiştir. İterasyon sayısı ağdaki düğüm sayısı olarak atanır.

Adım 1) Sırayla ağdan bir düğümü geçici olarak kaldır ve o düğümü ağa giriş olarak ver.

Adım 2) Giriş vektörü ile ağın düğümleri arasındaki mesafeleri hesapla. Minimum mesafedeki ağ düğümünün sınıfı, giriş vektörünün sınıfı ile aynı ise adım 1'e git.

Adım 3) Geçici olarak çıkarılan düğümü tekrar ağa ilave et.

Adım 4) İterasyon sayısını azalt. İterasyon sayısı sıfıra eşit ise unutma algoritmasını sonlandır. Aksi taktirde adım 1'e git.

3.4 Denetimsiz Yapay Sinir Ağları

Denetimsiz öğrenme yönteminde ağ, sınıfların öznitelik uzayındaki dağılımını inceleyerek kendi parametrelerini değiştirir. Bu yöntemde arzu edilen çıkış yoktur veya eğitim algoritması tarafından kullanılmaz, eğitimde kullanılan denklemler içinde de gözükmez. Kohonen, adaptif rezonans teorisi (ART) ve 'neocognitron' ağı denetimsiz öğrenme yöntemlerinin kullanıldığı başlıca önemli ağlardandır.

Eğitim tamamlanıp düğüm ağırlıkları uzayda belirli pozisyonlara yerleştikten sonra, bir etiketleme yöntemi kullanılarak düğümlerin sınıfları belirlenir.

3.4.1 Kohonen Ağı

Beyindeki hücrelerin yerleşimi belli bir dizilişe göredir ve bu diziliş alınan dış uyarının fiziksel özelliğine bağlıdır. Örneğin beynin ses ile ilgili bölgesinde, hücrelerin dizilişi ses dizgesinin sıklık yanıtını yansıtır. Alçak sıklıktaki işaretler bölgenin bir ucunda, yüksek sıklıktakiler öteki ucunda etkiye yol açarlar. Eğitim algoritması ile n boyutlu uzaydaki sınıf dağılımı iki boyutlu uzaya taşınmaktadır.

3.4.1.1 Kohonen Ağının Yapısı

Şekil 3.9'da Kohonen ağının yapısı gösterilmiştir [45]. Kohonen ağı, RÇE ve GAL ağları gibi iki katmandan meydana gelir. Ağın içindeki düğümlerin yapısı aşağıdaki denklemle ifade edilir.

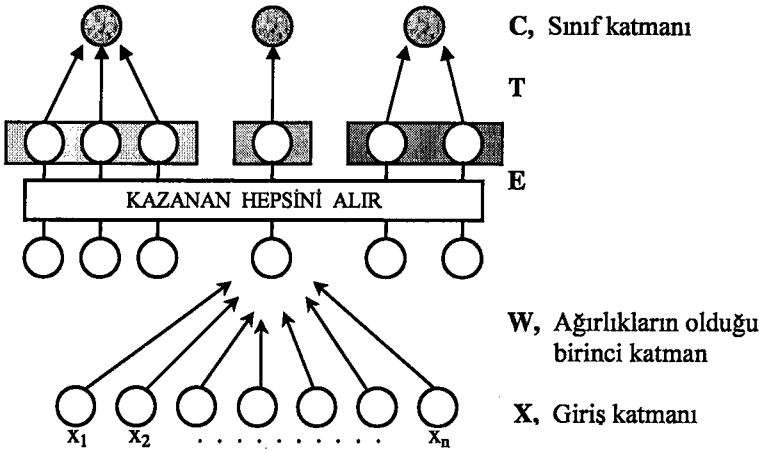
$$D_j = \sum_{i=1}^{n} \left(x_i - w_{ji}(k)\right)^2 \qquad E_e = \begin{cases} 1 & D_e = \min_j \{D_j\} \\ 0 & \text{aksi taktirde} \end{cases} \qquad (3.13)$$

$$\text{Çıkış katmanı} = C_c = \sum_e E_e \cdot T_{ec}$$

E_e, birinci katmandaki düğümlerin çıkışını; T_{ec}, ise sadece 0 veya 1 değerini alan OR'lama işlemini gerçekleyen bağlantı katsayısını gösterir.

İlk katman, düğüm ağırlıkları ile giriş vektörü arasında minimum mesafeyi bulmada kullanılırken; ikinci katman, ağdaki düğümlerin ait oldukları sınıfı tanımlamak için

kullanılır. İkinci katmandaki ağırlıklar, eğitim tamamlandıktan sonra bir etiketleme işlemi ile belirlenir. İkinci katman, aynı sınıfın çıkışlarını lojik olarak OR'lamak için kullanılır.



Şekil 3.9 Kohonen ağının yapısı

3.4.1.2 Kohonen Ağının Eğitimi

Düğümler komşularına meksika şapkası şeklinde bir bağlantı biçimi ile bağlıdır. Bu bağlantı sayesinde yarışmalı öğrenmede bir düğümün kazanması temin edilir. Kohonen ağının eğitiminde sadece kazanan düğümün ağırlığı değil, o düğümün çevresinde seçilen belirli bir komşuluk içindeki düğümlerin ağırlıkları da değiştirilir. Öğrenme esnasında bu komşuluğun sınırları zamanla lineer olmayacak şekilde azaltılır. Aşağıda Kohonen ağının eğitim algoritması verilmiştir.

Başlangıçta, çıkış düğüm sayısı, komşuluğun zamanla değişimi ve iterasyon sayısı belirlenir. Ağın tüm düğümlerinin ağırlıklarına 0 ile 1 arasında rastgele değerler verilir.

Adım 1) Eğitim kümesinden rastgele bir vektörü ağa giriş olarak ver.

Adım 2) Giriş vektörü ile ağın düğümleri arasındaki mesafeleri aşağıdaki ifadeyi kullanarak hesapla.

$$D_j = \sum_{i=1}^{n} \left(x_i - w_{ji}(k)\right)^2 \qquad (3.14)$$

x_i, giriş vektörünü; w_{ji} ise j. çıkışın ağırlık vektörünü temsil etmektedir.

Adım 3) Minimum uzaklığa sahip $J^{\star}$ çıkışını bul.

Adım 4) $J^{\star}$ çıkışını ve komşularının ağırlıklarını aşağıdaki ifadeleri kullanarak değiştir.

$$w_{ji}(k+1) = w_{ji}(k) + \eta(t) \cdot \left(x_i - w_{ji}(k)\right) \tag{3.15}$$

Kazanç terimi olarak ifade edilen $\eta(t)$, 0 ile 1 arasında ve zamanla azalacak şekilde değer alır.

Adım 5) İterasyon sayısını azalt. İterasyon sayısı sıfıra eşit değil ise adım 1'e git. Aksi taktirde eğitim algoritmasını durdur.

Eğitim tamamlandıktan sonra çıkış düğümlerine sınıf etiketleri vermek için her çıkış düğümüne sınıf sayısı kadar bellek (sınıf etiketi gözü) atanır. Eğitim kümesi içindeki herbir vektörün hangi çıkış düğümüne yakın olduğu araştırılır. İncelenen giriş vektörüne en yakın ağ düğümünün sınıf etiket gözü, giriş vektörünün sınıfı ile ilişkili olarak artırılır. Eğitim kümesindeki tüm vektörler için bu işlem tekrarlanır. İşlem tamamlandıktan sonra etiket gözü içinde en yüksek değeri veren sınıf araştırılır. Bulunan sınıf, çıkış düğümüne sınıf etiketi olarak atanır.

3.5 YSA'ların Sınıflayıcı Olarak Kullanımı

Çalışma içersinde yapay sinir ağları sınıflayıcı olarak kullanılmıştır. Ağların başarımında aşağıda ifade edilen 4 parametre incelenecektir.

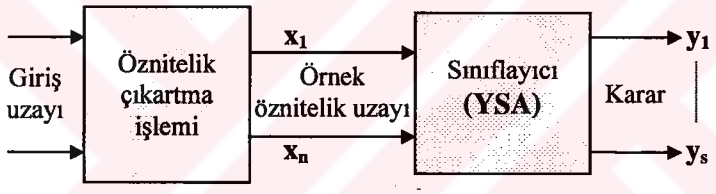
i) Eğitim zamanı
ii) Ağ düğüm sayısı
iii) Test kümesindeki vektörlerin doğru sınıflanma yüzdesi
iv) Fiziksel gerçekleme

Eğitim zamanı, ağın eğitim kümesindeki vektörler arasındaki bilgiyi ne kadar hızla öğrendiğini gösteren bir ölçüdür. Düğüm sayısı ise öğrendiği bilgiyi ne derece güçlü bir kodlama ile kendi içinde sakladığını ifade eder. Test kümesini sınıflama başarımı, ağın eğitim kümesinde bulunmayan vektörler hakkında karar verme yeteneğini (genelleme özelliğini) gösteren bir ölçüdür. İncelenen ağ yapılarının basit birimlerle oluşturulabilmesi, fiziksel gerçeklemeye olanak sağlamaktadır.

Yukarıda açıklaması yapılan parametreler hakkında daha fazla bilgi elde etmek için test amaçlı iki boyutlu bir örnek uzay tanımlanır. Örnek uzayda eğitim algoritmasının düğümleri yönlendirmesi (ağırlıkların değişimi) daha açık olarak gözlenebilecektir. Bu uzay sayesinde düğüm sayısının probleme bağlı olarak artışı incelenebilecektir. Daha çok vektör kullanılarak test yapıldığı için ağın genelleme yeteneği daha doğru belirlenebilecektir. Bu analiz aynı zamanda, yeni oluşturulan ağ düğüm yapılarının örnek uzayda görünümü ve uzayı bölmeleme yetenekleri hakkında görsel bir bilgi vermektedir.

Örnek uzayın matematiksel ifadesini ağın giriş/çıkış değerleri ile birlikte ele alırsak aşağıdaki gibi bir ilişki gözlenecektir:

$$F(x_1, x_2, ..., x_n) = \{ y_1, y_2, ...y_k, ..., y_s \mid k, s, y_k \in Z, \quad x_1, .., x_n \in R \} \qquad (3.16)$$



Şekil 3.10 Karar verme mekanizması

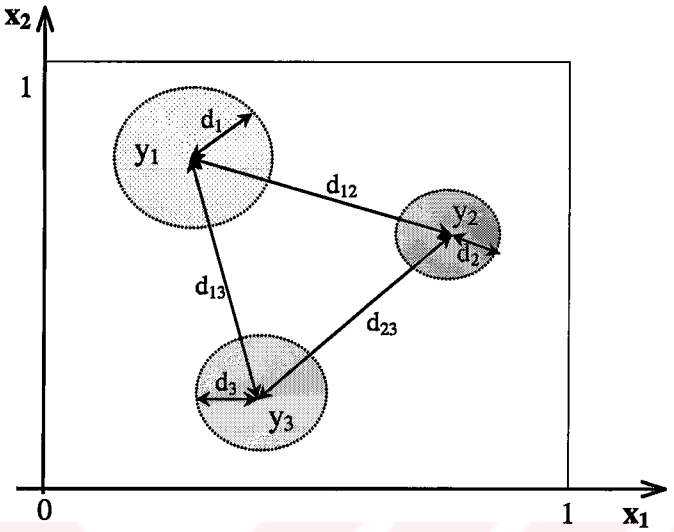
Bu ifadelerde s sınıf sayısını, n öznitelik uzayının boyutunu, y_k sınıfları birbirinden ayıran sınıf etiketini (gri değeri), $\overline{X} = \{x_1, x_2, ..., x_n\}$ ise n boyutlu öznitelik vektörünü ifade etmektedir. Doğal olarak $F(x_1, x_2)$ fonksiyonu iki boyutta bir görüntü oluşturacaktır. Test için iki boyutlu uzay seçilmesi, fonksiyonun dağılımının ve sınıflama sonucunun görsel olarak incelenmesine olanak sağlamaktadır.

Örnek uzay, 3.16'daki formül gereği herhangi bir fonksiyon olabilir. Ancak ağın yeteneğini araştırmak için bu fonksiyon üzerinde bir ölçüde sınırlamalar getirilecektir. Örnek olarak bir $F(x_1, x_2)$ fonksiyonunu üç sınıf (s=3) için inceleyelim.

$$F(x_1, x_2) = \{ y_1, y_2, y_3 \mid y_1, y_2, y_3 \in Z; \; x_1, x_2 \in R \} \qquad (3.17)$$

Yukardaki fonksiyonun şekil 3.11'deki gibi bir sınıf dağılımına sahip olduğunu düşünürsek öznitelik çıkartma işleminin başarımını, d_1, d_2, d_3'ün küçük değerlerde

d_{12}, d_{13} ve d_{23}'ün büyük değerlerde oluşu belirleyecektir. Böyle bir sınıf dağılımına sahip öznitelik uzayı için lineer olmayan bir yapıya sahip sınıflayıcılar kullanmak gereksizdir.

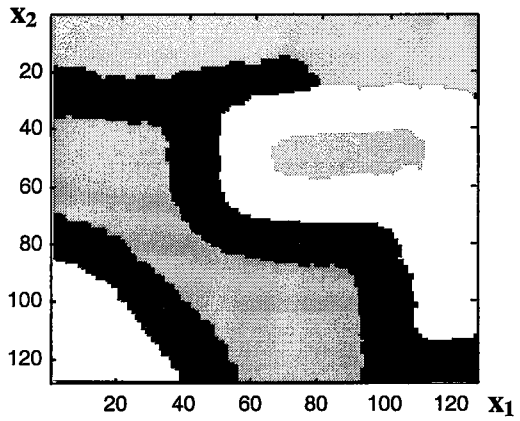


Şekil 3.11 Optimum sınıf dağılımı

Yeteri kadar ayrık değer her zaman elde edilemediği için (dolayısıyla eğitim kümesinde bulunan vektör sayısı azdır) sınıf dağılımını ifade eden bir fonksiyon tam olarak bulunamaz. Bazı durumlarda öznitelik uzayında, aynı sınıftan farklı bölgelerde öbekleşmeler hatta sınıfların birbirlerinin içine girdiği karmaşık dağılımlar oluşabilmektedir.

Şekil 3.12'de üç sınıflı bir örnek uzay ($F(x_1, x_2)$ fonksiyonu) tanımlanmıştır. Bu uzayda siyah renkle ifade edilen bölgeler hiçbir sınıfa dahil değildir. Bazı sınıfların örnek uzayda birbirlerine bitişik olması veya aynı sınıfın farklı yerlerde öbekleşmeler göstermesi, öznitelik çıkartma işleminin başarılı olamaması durumunu içeren bir örnek teşkil etmektedir. Şekil 3.11 ve 3.12 için bulunan diverjans değerleri sırasıyla 27.8299 ve 0.4900'dur. Küçük diverjans değeri (0.49), örnek uzayda vektörlerin saçıldığının sayısal bir göstergesidir.

Örnek uzay sayesinde ağın eğitim algoritmasının stratejisi, düğümlerin bu uzayda hareketi (yerleştikleri pozisyon), düğüm sayısı ve genelleme özelliği görsel olarak analiz edilecektir. Bu analiz, bölüm 4 içinde geliştirilen yeni yapay sinir ağlarının yapısına ışık tutacaktır.



Şekil 3.12 İki boyutlu örnek uzay

Yapay sinir ağlarının paralel çalışma yeteneği, öğrenerek kendini geliştiren bir eğitim yöntemi, donanım olarak kolay gerçeklenebilir olması ve genelleme yeteneği en önemli özelliklerindendir. Bu özellikleri nedeniyle yapay sinir ağlarının sınıflayıcı olarak biyomedikal alanında sıkça kullanıldığı gözlenmektedir [46-52].

Özniteliklerin düzgün belirlenememesi durumunda bile yapay sinir ağlarının yetenekleri sayesinde doğru sınıflama kararları elde edilir. Karar veren sistemin tasarımında, sınıflara ait özniteliklerin belirlenmesi için aşırı bir araştırma yapılmadan YSA'ların yetenekleri kullanılarak çözümler üretilir.

4. YAPAY SİNİR AĞLARININ GENETİK ALGORİTMALAR İLE EĞİTİLMESİ

4.1 Giriş

Literatür incelendiğinde ağların, minimum sayıda düğüm ile yüksek sınıflama performansı verecek şekilde eğitilmediği ve çoğu eğitim algoritmasında da yapıyı ihtiyaca göre büyütecek bir özelliğin bulunmadığı gözlenmektedir. Bazı eğitim algoritmaları, ağın ağırlıklarını yerel optimumlara götürmektedir. Yeni bir ağ yapısı tasarlandığında, ağı yerel optimumlara götürmeden eğitecek algoritmaları bulmak son derece zordur. Bu nedenle tez çalışması içerisinde doğada işlerliği gözüken, işlemleri basit ve türev varlığı gibi bazı sınırlamalardan etkilenmeyen bir eğitim algoritması incelenmiştir. Genetik algoritmalar (GA) olarak bilinen bu yeni eğitim algoritması çeşitli ağlar üzerinde denenmiş ve eski yöntemlerle bulunan sonuçlar karşılaştırılmıştır. Eğitim işlemi sırasında ağın düğüm sayısı da kontrol edilerek, ihtiyaca göre ağa yeni düğümlerin katılması sağlanmıştır. Böylece ağ, performansını artırmak için kendine yeni düğümler temin ederek kontrollü büyür.

4.2 Genetik Algoritmalar

Genetik algoritmalar temel olarak üç işlemden meydana gelir: *Kopyalama*, *çaprazlama* ve *mutasyon* [53, 54]. Genetik algoritmalarda bu üç temel işleme ilaveten ters çevirme, sıralama, 'dominance', 'diploidy', 'abevance' işlemleri de kullanılır. Son üç işlem, giriş uzayındaki sınıfların dağılımının durağan olmadığı durumlarda kullanılmaktadır. Tez içinde incelenen tüm giriş uzayları durağan olarak kabul edilmiştir. Çalışmada kullanılmayacağı için bu son beş işlemin matematiksel ifadeleri verilmemiştir.

Genetik işlemler ikili (binary) dizilerden meydana gelen, *havuz* adı verilen bir yığın içinde gerçeklenir. Giriş veya sistem parametreleri havuz içinde genellikle ikili diziler olarak temsil edilir. Havuz içindeki diziler başlangıçta rastgele değerlerden meydana gelir. Eğitime başlamadan önce havuz içindeki dizi sayısı belirlenir ve bu sayı eğitim süresince sabit tutulur.

Bir nesil 4 işlemden meydana gelir: i) Uyumluluk değeri hesabı, ii)Kopyalama, iii)Çaprazlama, iv)Mutasyon. Uyumluluk fonksiyonu, sistem parametrelerine bağlı optimizasyon kriteri olarak tanımlanır. Herbir nesilde, havuz içindeki dizilerin uyumluluk değerleri hesaplanır, daha sonra bu dizilere kopyalama, çaprazlama ve mutasyon işlemleri uygulanır. Optimum sonuçları elde etmek için aynı işlemler birkaç nesil devam ettirilir.

4.2.1 Kopyalama İşlemi

Uyumluluk değeri, kopyalama işleminin gerçeklenmesinde kullanılmaktadır. Havuz içindeki herbir dizi için uyumluluk değeri hesaplanır. Bu değerler kullanılarak ortalama uyumluluk değeri bulunur. Daha sonra aşağıdaki eşitlikler kullanılarak herbir dizinin havuzda kopyalanma sayısı tespit edilir.

$$F = \frac{1}{N} \cdot \sum_{i=1}^{N} f_i \qquad K_i = \frac{f_i}{F} \tag{4.1}$$

Denklemlerde N, havuzdaki dizi sayısını; f_i , herbir dizinin uyumluluk değerini; F, ortalama uyumluluk değerini; K_i, i. dizinin havuz içinde kopyalanma sayısını göstermektedir. K_i'ler ondalık değerler olarak bulunur. Havuzdaki dizi sayısında bir artma olmayacak şekilde denklem 4.2 kullanılarak K_i'ler tam sayı olarak seçilir.

$$N = \sum_{i=1}^{N} K_i \tag{4.2}$$

Ortalama uyumluluk değerine bağlı olarak bazı dizilerin sayısı havuz içinde artarken, bazı diziler ise havuzdan yok edilir. Böylece her nesilde havuz içinde, performans kriterine göre daha uygun dizilerin sayısında bir artış sağlanır. Kopyalama işlemi havuz içinde yeni araştırma dizileri oluşturmaz. Bu işlem ile uyumluluk değerleri ortalamaya göre büyük olan dizilerin havuz içinde yaşama şansı artırılır.

4.2.2 Çaprazlama İşlemi

Kopyalama işlemi ile havuz içinde daha uygun dizilerin sayısı artırılmıştır. Bu işlem ile araştırma uzayında (genetik havuz içinde) yeni vektörler (diziler) oluşturulmaz. Dolayısıyla sadece kopyalama işlemi ile uzayın farklı bölgelerinde araştırma gerçeklenemez. Havuz içinde yeni dizilerin (vektörlerin) oluşturulması gerekir. Yeni dizileri oluşturma görevini çaprazlama işlemi üstlenir. Literatürde iki tip çaprazlama işleminin sıkça kullanıldığı gözlenmiştir: Basit çaprazlama, üniform çaprazlama.

Basit çaprazlama işleminde, önce çaprazlama olasılığı seçilir. Bu değer dizi içinde kaç ikilide bir çaprazlama yapılacağını (çaprazlama noktalarını) gösterir. İki dizi havuz içinde rastgele seçilir ve bu olasılığın belirttiği iki çaprazlama noktası arasında kalan ikililer, her iki dizide yer değiştirir. Örnek olarak 30 ikiliden meydana gelen bir dizi içinde çaprazlama yapılmak istenirse, bu işlem ve sonuç olarak elde edilen A′ ve B′ yeni dizileri aşağıda gösterilmiştir.

Eski diziler:

A dizisi ⟶ 110010101110100100011100010101

çaprazlanacak bölgeler

B dizisi ⟶ 100110110101110011110011101001

Yeni diziler:

A′ dizisi ⟶ 110010110110100110111111101101

çaprazlanmış bölgeler

B′ dizisi ⟶ 100110101101100100110000010001

Üniform çaprazlama işleminde, dizilerdeki her ikili için belirli bir olasılığa göre çaprazlama mümkündür. Basit çaprazlama işleminde bölgeler çaprazlanırken, üniform çaprazlamada ikililer tek tek çaprazlanır. Üniform adı, herbir ikili için çaprazlama olasılığının 0.5 olarak seçilmesinden kaynaklanmaktadır. Bu olay, dizi içindeki her ikili için yazı-tura atılmasına benzetilir.

4.2.3 Mutasyon İşlemi

Yukarda tanımlanan iki işlem tarafından yerel bir optimuma yakınsama oluşursa, yerel optimumdan genetik dizileri kurtarma görevi mutasyon işleminindir. Havuz içindeki dizilerin ikilileri belirli bir olasılığa göre 1 ise 0, 0 ise 1 yapılır. Bu olasılık değeri, dizi içinde kaç ikilide bir değişim yapılacağını gösterir. Mutasyon olasılığı büyük tutulmaz, küçük bir değerdedir. Büyük değerde olması havuzda rastgeleliği artırır ve optimum çözüm etrafında salınım oluşmasına neden olur. Literatürde bazen bu olasılık değeri nesile göre değiştirilir, önce büyük seçilir sonra bu değer azaltılır.

4.3 YSA'ların Genetik Algoritmalar ile Eğitimi

Literatürde, genetik algoritmalar ile yapay sinir ağlarının kullanımının üç şekilde gerçekleştiği gözlenmektedir [55]: Birinci tip uygulamada genetik algoritmalar, yapay sinir ağlarının eğitiminde kullanılacak en uygun öznitelikleri araştırır [56]. İkinci tip uygulamada genetik algoritmalar, incelenen ağın eğitim algoritmasında bulunan parametrelerin belirlenmesinde kullanılır [57, 58]. Ağın ağırlıkları, genetik algoritmalar tarafından değiştirilmez; ağırlıklar ağın kendi eğitim algoritması tarafından değiştirilir. Eğitimde hem genetik algoritmalar hem de ağın kendi öğrenme algoritması birlikte kullanıldığı için harcanan zaman normal eğitimdeki süreye göre daha uzundur. Üçüncü tip uygulamada ağın ağırlık katsayıları ve topolojisi ya birlikte [59-63] veya ayrı ayrı [64] genetik algoritmalar ile bulunur. Tez içerisinde sunulan çalışmalar üçüncü tip uygulamaya aittir ve ağların sadece birinci katmanındaki ağırlıkları genetik algoritmalar ile bulunmuştur. Ağların birinci katmanındaki düğümler eğitim sırasında kontrollü olarak ihtiyaca göre artırılmıştır.

Genetik algoritmaların yapay sinir ağları ile kullanımında karşılaşılan en büyük problem, eğitimin uzun zamanda gerçekleşmesidir. Genetik havuzdaki dizi boylarının aşırı derecede büyük olması, araştırmanın optimum bölgeye yaklaşmasını geciktirici etki yapar. Bu etkiyi azaltabilmek için literatürde dizi boylarını küçültmeye yönelik çalışmalar gözlenmektedir [64, 65]. Yapay sinir ağlarının hem ağırlıkları hem de topolojilerinin değiştirilmesi durumunda, genellikle diziler tüm ağın ağırlık katsayılarını temsil eder. Bu nedenle dizi boyları (araştırma uzayının boyutu) çok büyür. Tez içersinde (2 yapı hariç) gerçeklenen 3 farklı çalışmada da

genetik havuzdaki her dizi sadece bir düğümü temsil etmektedir. Dizilerin boyu küçük olduğu için (araştırma uzayının boyutu küçük) aranılan bölgelere yakınsama hızla gerçekleşir.

4.4 Kohonen Ağının Genetik Algoritmalar ile Eğitimi

Genel anlamda çok boyutlu öznitelik uzayında sınıflar, bir veya birden fazla ayrık hacimsel bölgeyi işgal edebilmektedir. Kohonen ve GAL ağının eğitim algoritmaları ile bu bölgeleri, sınıflarına uygun olarak temsil edecek düğüm pozisyonları araştırılır. Eğitim sonunda ağın düğümleri, öznitelik uzayında sınıflara ait bölgeleri doğru temsil edecek pozisyona getirilir.

Kohonen ağının eğitim algoritması hızlıdır. Ancak, ağda başlıca üç problemle karşılaşılmaktadır:

1) Çıkış katmanındaki düğüm sayısının ağın eğitiminden önce belirlenmesi gerekir. Düğüm sayısı küçük seçilirse öznitelik uzayındaki sınıflar doğru olarak temsil edilemez; büyük seçilirse eğitim için uzun zaman harcanır ve ayrıca yapı içinde anlamsız düğümler oluşur.
2) Komşuluk zayıflatma katsayısının belirlenmesi konusunda bir yöntem mevcut değildir, denemelerle en iyi sonuç bulunur.
3) Eğitim algoritması öznitelik uzayındaki sınıfların iç bölgelerini de temsil etmek için düğüm oluşturur. Böylece sınıfların iç bölgeleri de gereksiz yere çıkış düğümleri tarafından temsil edilir.

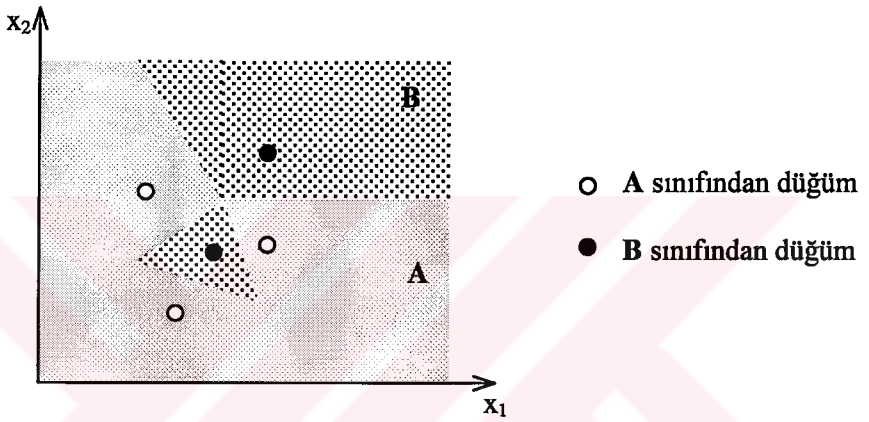
GAL ağının eğitim algoritması hızlıdır. Ancak, ağda başlıca iki problemle karşılaşılmaktadır:

1) Ağ, eğitim kümesindeki vektörlerden oluşturulur. Eğitim işlemi sırasında düğümlerin ağırlıkları değiştirilmez. Dolayısıyla ağ düğümleri, eğitim kümesi dışında bir bilgi taşımaz. Ağın eğitiminde ezberci bir yöntem kullanılmaktadır. Eğitim kümesi sınıflar hakkında yeterli bilgi taşımadığı taktirde, ağın genelleme yeteneği kötüleşir.
2) Ağın eğitim algoritması düğümleri sınıf sınırlarına doğru çekmektedir. Öznitelik uzayında sınıf sınırlarının biribirine yaklaşması durumunda, ağın eğitim algoritması aşırı sayıda düğüm üretmeye yönelmektedir.

Kohonen ve GAL ağlarındaki bu problemleri gidermek ve daha uygun düğüm pozisyonları bulmak için aşağıda, Kohonen ağının genetik algoritmalar kullanılarak eğitilmesi önerilmiştir. Eğitim algoritmasındaki farklılıktan dolayı bu ağa, genetik temelli Kohonen (GetKoh) ağı ismi verilmiştir.

4.4.1 GetKoh Ağı ile Öznitelik Uzayının Bölmelenmesi

Şekil 4.1'de iki boyutlu uzayda ağ düğümleri yardımıyla sınıfların birbirinden ayrılması gösterilmiştir.



Şekil 4.1 Düğümler yardımıyla sınıf sınırlarının oluşturulması

Sınıflama kararında en yakın mesafe kuralı kullanıldığı için iki farklı sınıftan düğüm öznitelik uzayında bir doğru ile birbirinden ayrılmaktadır: İki boyutlu uzayda farklı iki sınıftan düğüm arasındaki karar sınırı, bu iki sınıfı temsil eden düğümlere eşit mesafede bulunan bir doğruyu (üç boyutlu uzayda düzlem, daha yukarı boyutlarda hiper düzlem) tanımlar. Ağın düğümünün öznitelik uzayında kapalı bir bölge oluşturması boyuta değil, diğer düğümlerin bu uzaydaki konumlarına bağlıdır. Kapalı olmayan bir bölgenin oluşturulması daha kolaydır. Bazı durumlarda bir ağ düğümü, diğer düğümlerin uzaydaki yerine bağlı olarak en fazla 1 kapalı bölge oluşturabilir.

GetKoh ağındaki bir düğüm, n değişkeni öznitelik uzayının boyutu olmak üzere n adet parametre ile temsil edilmektedir. Eğitim sonrasında M adet çıkış düğümü için ağda M×n adet parametre ile sınıf dağılım bilgisi saklanır.

4.4.2 GetKoh Ağının Yapısı

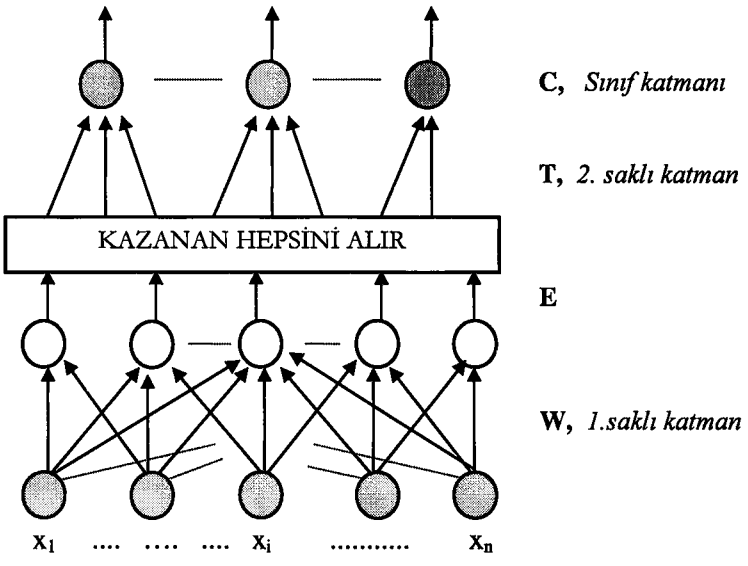
GetKoh ağında herbir düğüm, sınıfların dağıldığı hacmin bir parçasını saklar. Sınıflar hakkındaki bilgi düğümlere dağıtılarak saklanmamıştır. Dolayısıyla herbir düğüm sınıf dağılımı hakkında yerel bir bilgiye sahiptir. Düğüm ölümü ile saklanan bu bilgi de yok olur. Bilginin bu şekilde dağıtılmadan saklanması, ağın fiziksel olarak iki katman şeklinde gerçeklenmesine imkan vermektedir.

Şekil 4.2'de ağın yapısı gösterilmiştir [15, 18]. GetKoh ağı, Kohonen ve GAL ağı gibi iki katmandan meydana gelir. Ağın içindeki düğümlerin yapısı aşağıdaki denklemlerle ifade edilir.

$$D_j = \sum_{i=1}^{n} \left(x_i - w_{ji}\right)^2 \qquad E_e = \begin{cases} 1 & D_e = \min_j \{D_j\} \\ 0 & \text{aksi taktirde} \end{cases} \tag{4.11}$$

$$\text{Çıkış katmanı} = C_c = \sum_e E_e \cdot T_{ec}$$

x_i, giriş vektörünün i. elemanını (i. giriş düğümü); w_{ji}, birinci saklı katmandaki j. düğüm ile i. giriş düğümü arasındaki bağlantıyı; E_e, birinci katmandaki düğümlerin çıkışını; T_{ec}, ise sadece 0 veya 1 değerini alan OR'lama işlemini gerçekleyen bağlantı katsayısını gösterir.

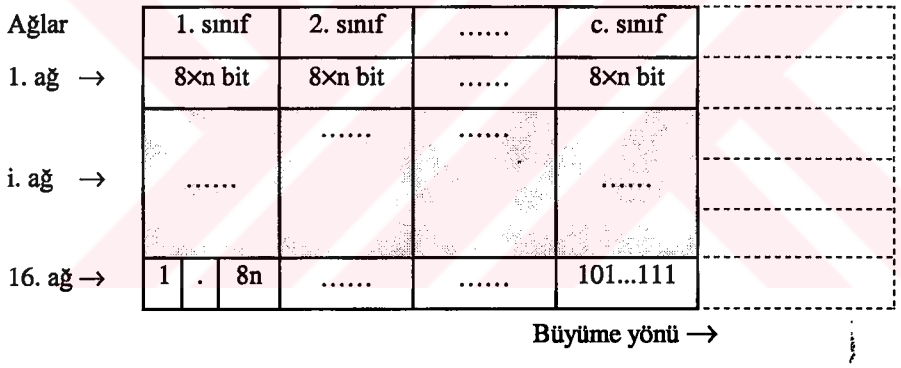


Şekil 4.2 GetKoh ağının yapısı

İlk katman, düğüm ağırlıkları ile giriş vektörü arasında minimum mesafeyi bulmada kullanılırken, ikinci katman ağdaki düğümlerin ait oldukları sınıfı tanımlamak için kullanılır. İkinci katmandaki ağırlıklar başlangıçta 0 değerini alır, öğrenme esnasında bu bağlantılar 1'lenir. İkinci katman, aynı sınıftan çıkışları lojik olarak OR'lamak için kullanılmaktadır.

4.4.3 GetKoh Ağının Eğitimi

Genetik havuzdaki dizilerin herbiri farklı bir ağı temsil eder. Havuzdaki diziler, ağdaki düğümlerin ağırlıklarının ard arda eklenmesiyle oluşturulur. Şekil 4.3'te genetik havuzdaki diziler gösterilmiştir. c ve n parametreleri sırasıyla sınıf sayısını ve öznitelik vektörünün boyutunu temsil etmektedir. Çalışma içerisinde genetik havuzun derinliği 16 dizi seçilmiştir. Her nesilde 4 işlem gerçeklenir: Uyumluluk değeri hesabı, kopyalama, çaprazlama ve mutasyon işlemleri.



Şekil 4.3 GetKoh ağının eğitiminde kullanılan genetik havuz

Eğitimin başlangıcında her sınıftan bir vektör olmak üzere eğitim kümesinden sınıf sayısı kadar vektör alınarak havuz oluşturulur. 10 nesil sonunda en yüksek uyumluluk değeri araştırılır. Bu değer, 0.95'in altında ise tüm dizilerin boyutu bir vektör ilave edilerek artırılır. Diziler başlangıçta c (sınıf sayısı) adet vektör içerdiği için ilave vektör ile dizi boyları c+1 vektöre çıkar. (c+1)inci vektörler başarımı iyileştirmek için en kötü sınıflanan sınıfın vektörlerinden meydana getirilir. 10 nesil için yeniden araştırma başlatılır. En yüksek uyumluluk değeri 0.95 olana kadar dizi boyları büyütülerek araştırmaya devam edilir.

GetKoh ağının eğitiminde kullanılan uyumluluk değeri (UD) aşağıda tanımlanmıştır.

$\#\{DSV_k\}$: k. ağ (dizi) tarafından doğru sınıflanan vektörlerin sayısı

#{E} : Eğitim kümesindeki vektör sayısı

UD_k : k. dizinin uyumluluk değeri

$$UD_k = \frac{\#\{DSV_k\}}{\#\{E\}} \tag{4.12}$$

GetKoh ağının eğitiminde kullanılan adımlar aşağıda verilmiştir.

Adım 1) Herbiri bir sınıftan olmak üzere her sınıf için eğitim kümesinden bir vektör seçerek genetik havuzu oluştur. Nesil sayısını belirle.

Adım 2) Genetik havuzdaki her dizi için uyumluluk değeri hesapla.

Adım 3) Havuz içindeki tüm dizilere kopyalama, çaprazlama ve mutasyon işlemlerini uygula. Nesil sayacını 1 azalt. Nesil sayacı sıfıra eşit değilse adım 2'ye git.

Adım 4) Nesil sayacını başlangıç değeri ile kur. Havuz içinde en yüksek uyumluluk değerini veren diziyi bul.

Adım 5) Uyumluluk değeri 0.95'ten büyük ise eğitim algoritmasını sonlandır.

Adım 6) En iyi uyumluluk değeri veren dizinin sınıflama sonuçlarını inceleyerek başarımı düşük olan sınıfı bul. Havuzdaki tüm dizilerin boyunu bir artır ve bu ilave alanları, eğitim kümesinde başarımı düşük olan sınıfa ait vektörler ile doldurarak havuzu yeniden oluştur. Adım 2'ye git.

4.5 Çok Katmanlı Ağın Genetik Algoritmalar ile Eğitimi

Çok katmanlı ağ, literatürde sınıflama işlemi için sıkça kullanılmasına rağmen ağın başlıca dört problemi bulunmaktadır:

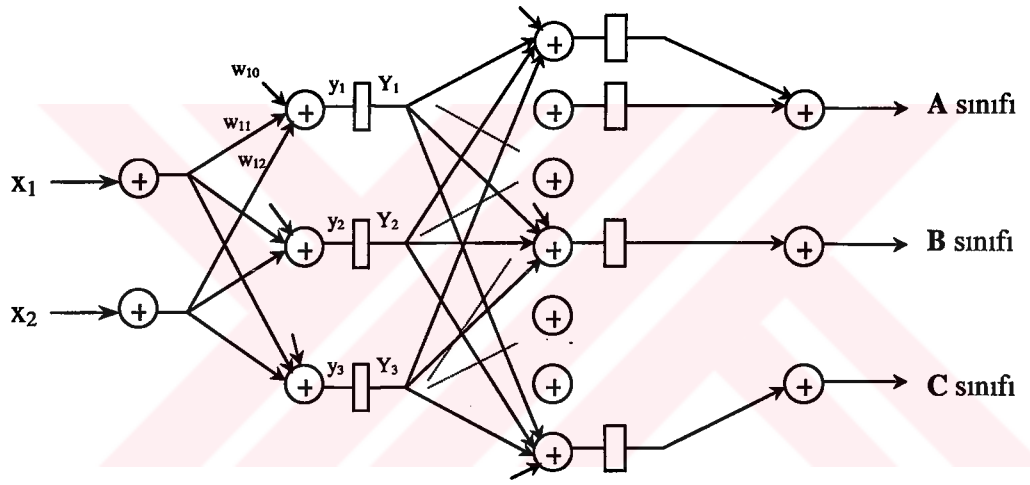
1) Çok katmanlı ağın katmanlarındaki düğüm sayısı, eğitime başlamadan önce belirlenmelidir. Seçilen düğüm sayısı sınıflama performansını doğrudan etkileyecektir. Katmanlardaki düğüm sayısını belirleyecek bir yöntem de mevcut değildir.
2) Geriye yayılma algoritması, çok katmanlı ağın eğitiminde sıkça kullanılmaktadır. Ancak bu eğitim algoritması ağı, yerel optimuma götürebilmektedir.
3) Geriye yayılma algoritması, literatürde kullanılan birçok öğrenme algoritmasından yavaş kalmaktadır.

4) Çok katmanlı ağın eğitiminde düğüm sayısını ihtiyaca göre büyütecek bir mekanizma bulunmamaktadır.

Eğitim sırasında karşılaşılan problemleri gidermek ve daha az düğümle daha iyi başarım elde etmek için aşağıda çok katmanlı ağa, genetik temelli (GetÇKA) bir eğitim önerilmiştir.

4.5.1 GetÇKA ile Öznitelik Uzayının Bölmelenmesi

Şekil 4.4'te gösterilen iki girişli, üç çıkışlı, üç katmanlı bir ağ yardımıyla şekil 4.5'teki öznitelik uzayının sınıflanmasını inceleyelim.



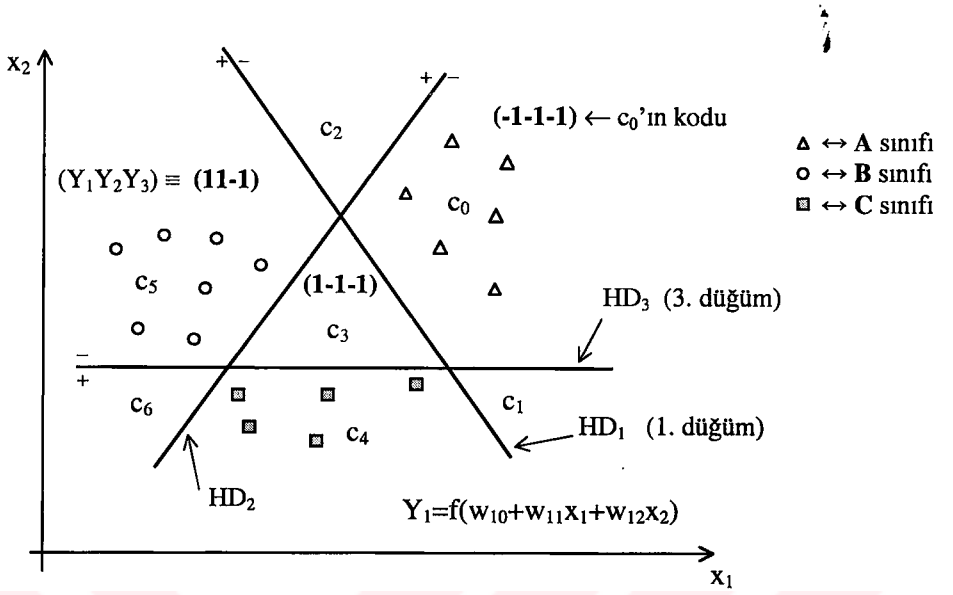
Şekil 4.4 İki girişli, üç çıkışlı, üç katmandan oluşan ÇKA

Şekil 4.5'te gösterilen üç doğru, ağın birinci katmanındaki çıkışları temsil etmektedir. İncelenen ağın tüm çıkış düğümlerinde kırpıcı adı verilen aşağıdaki özelliklere sahip bir fonksiyon kullanılmıştır.

$$Y_j = f(y_j) = \begin{cases} +1 & y_j \geq 0 \\ -1 & y_j < 0 \end{cases} \qquad (4.13)$$

$$y_j = \sum_{i=0}^{n0} x_i \cdot w_{ji} \qquad x_0=1 \qquad n0: \text{ giriş uzay boyutu}$$

Öznitelik uzayında üç sınıf bulunmaktadır, $(Y_1Y_2Y_3)$ olarak gösterilen ifadeler, ağın birinci katmanındaki çıkış değerleridir.



Şekil 4.5 İki boyutta üç tane doğru ile bölmelenmiş örnek uzay

Çok katmanlı ağın ilk katmanı ayrık sınıfları içine alacak şekilde örnek uzayı bölmeler. Böylece örnek uzaydaki sınıflar doğrular (çok boyutta hiper düzlemler, HD) yardımıyla birbirlerinden ayrılır. Örnek uzay doğrular ile ne kadar çok bölmelenirse, sınıf dağılımları o kadar iyi tanımlanabilecektir. Örnek uzayda ağın birinci katmanındaki çıkışlar, gelen giriş vektörleri için 7 farklı değer üretmektedir. Giriş vektörü, aynı bölge içinde farklı yerlere düşerse, ağın birinci katmanında sabit bir vektör (kod) üretilecektir. Böylece, doğruların kesişimi ile oluşturulan bölgeler, örnek uzayda belirli kodlarla temsil edilir.

İkinci katman çıkış uzayının boyutu, birinci katmanın çıkışındaki bölge sayısı ile belirlenir. Bu katman, bölgeleri tanımlayan kodları tek bir çıkışa dönüştürmek için kullanılır. Çok boyutlu kodları bir tek çıkışa dönüştürdüğü için bu katmanın lojik AND işlemi gerçeklediği düşünülebilir.

Üçüncü katman giriş uzayının boyutu, ikinci katmanın çıkışındaki düğüm sayısı ile belirlenir. Görevi, ikinci katman çıkışında aynı sınıfa ait düğümleri, tek bir sınıf düğümüne birleştirmektir. Bu görev incelenirse, bir anlamda lojik OR işlemini gerçeklediği gözlenir.

Bu inceleme ile çok katmanlı ağın özellikle birinci katmanındaki düğüm sayısının ve bu düğümlerin ağırlıklarının, sınıflama performansını son derece etkilediği gözlenmektedir. Aşağıda d-boyutlu bir örnek uzaya hiper düzlem ilave edildiğinde oluşabilecek maksimum bölge sayısı ile boyut arasındaki ilişki araştırılmıştır [66].

d boyutlu bir uzayda, p adet doğrunun kesişmesi sonucu C(p,d) bölge oluşurken; aynı uzayda p-1 doğrunun kesişimi sonucu C(p-1,d) bölge oluştuğunu varsayalım. Önce p. doğru eklendiğinde oluşan ilave bölge sayısını bulalım.

p. doğrunun, C(p-1,d) bölgenin M tanesini kestiğini düşünelim. p. doğru yardımıyla kesilen her bölge 2 parçaya bölünür. Bu nedenle oluşturulan ilave bölge sayısı M'dir. p. doğru tarafından oluşturulan M bölge sayısı önceki p-1 doğrunun, p. doğruyu kesmesi sonucu oluşan bölme sayısına eşittir. p. doğru d-1 boyutlu bir uzay olduğu için, p-1 tane doğru p. doğrunun d-1 uzayında C(p-1, d-1) bölge oluşturacaktır. Bu durumda ilave bölge sayısı M=C(p-1,d-1)'dir ve C(p,d) aşağıdaki şekilde ifade edilir.

$$C(p, d) = C(p-1, d) + C(p-1, d-1)$$

Aşağıdaki başlangıç değerleri ile bu ardışıl denklem hesaplanırsa 4.14'teki ifadeyi elde ederiz.

$$C(0, d) = 1, \quad d \geq 0; \qquad C(p, 0) = 1, \quad p \geq 1$$

$$C(p,d) = \begin{cases} 2^p & p \leq d \\ \sum_{i=0}^{d} \binom{p}{i} & p > d \end{cases} \tag{4.14}$$

Burada, $\binom{p}{i} = \frac{p!}{i!(p-i)!}$ olarak hesaplanır.

Aşağıda, 4.14'teki ifade kullanılarak bir boyut ilave etmekle oluşan bölge sayısı bulunmuştur.

$$C(p, 1) = 1 + p$$

$$C(p, 2) = 1 + 1/2p + 1/2p^2$$

$$C(p, 3) = 1 + 5/6p + 1/6p^3$$

$$C(p, 4) = 1 + 7/12p + 11/24p^2 - 1/12p^3 + 1/24p^4$$

Yukardaki ardışıl ifadelerden,

$$C(p, d) = C(p, d-1) + \binom{p}{d} \tag{4.15}$$

bulunur. 4.15 ifadesini, p yerine p-1 yazarak yeniden düzenleyelim;

$$C(p-1, d-1) = C(p-1, d) - \binom{p-1}{d}$$

$$C(p, d) = 2 \times C(p-1, d) - \binom{p-1}{d}$$

Böylece boyut aynı iken bir düzlem ilave etmekle oluşan bölge sayısı ifade edilmiş olur. Bulunan bölgeleri iki kısma ayırabiliriz: Kapalı bölge ve açık bölge. Her iki bölgenin toplamı bize, toplam bölge sayısını verir.

$C_o(p, d)$ = Açık bölge sayısı, $\quad C_o(p, d) = C_o(p-1, d) + C_o(p-1, d-1)$

$C_c(p, d)$ = Kapalı bölge sayısı, $\quad C_c(p, d) = C_c(p-1, d) + C_c(p-1, d-1)$

$$C(p, d) = C_o(p, d) + C_c(p, d)$$

Aşağıdaki başlangıç değerleri ile ardışıl denklem hesaplanırsa 4.16'deki sonuçları elde ederiz.

$C_o(0, d) = 1, \quad d \geq 0; \qquad C_o(p, 0) = 0, \quad p \geq 1$

$C_c(0, d) = 0, \quad d \geq 0; \qquad C_c(p, 0) = 1, \quad p \geq 1$

$$C_o(p,d) = \begin{cases} 2^p, & p \leq d \\ 2\sum_{i=0}^{d-1} \binom{p-1}{i}, & p > d \end{cases} \qquad C_c(p,d) = \begin{cases} 0, & p \leq d \\ \binom{p-1}{d}, & p > d \end{cases} \tag{4.16}$$

Bu incelemede p değişkeni birinci saklı katmandaki düğüm sayısını; d değişkeni ise giriş düğümünün sayısını göstermektedir.

Ortaya çıkan bölge sayısını bir örnek üzerinde inceleyelim. Giriş düğümlerinin sayısı 10 olan ve birinci saklı katman çıkışında 25 düğümü bulunan bir ağ, 10 boyutlu uzayda 4.540.386 tane bölge (kapalı/açık bölge) oluşturabilmektedir.

Birinci saklı katmandaki düğüm sayısının önceden doğru belirlenememesi ağın sınıflama performansının aşırı derecede düşmesine yol açacaktır. Çok katmanlı ağın eğitiminde kullanılan klasik yöntemlerde saklı katmandaki düğüm sayısı ihtiyaca göre otomatik olarak belirlenemez. Bu değerleri önceden belirleyecek yöntemler de mevcut değildir.

4.5.2 GetÇKA'nın Yapısı

Sınıflar hakkındaki bilgi, GetÇKA'nın düğümlerine dağıtılarak saklanmıştır. Dolayısıyla herbir düğüm tüm sınıf dağılımı hakkında global bir bilgiye sahiptir. Bilginin bu şekilde dağıtılarak saklanması, ağın fiziksel olarak 2'den fazla katman ile gerçeklenmesine neden olmaktadır.

Şekil 4.6'da GetÇKA'nın yapısı gösterilmiştir. Ağ, üç katmandan meydana gelir. İlk katmandaki düğümler, öznitelik uzayında hiper düzlemlerin kesişmeleri ile bölge oluşturmak için kullanılır. İlk katmandaki herbir düğüm, öznitelik uzayında bir hiper düzleme karşılık gelmektedir. Birinci katmandaki düğümlerin çıkışları aşağıdaki şekilde hesaplanır.

$$\overline{O}^1 = \overline{X} \quad \text{Giriş vektörü}$$

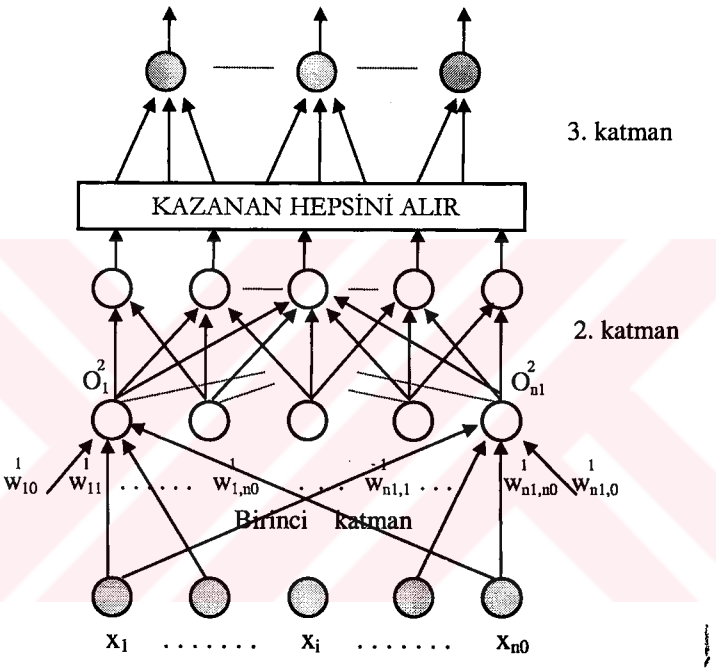
$$\overline{O}^1 = (O_0^1, O_1^1, \ldots\ldots, O_{n0}^1) \qquad O_0^1 = 1$$

$$I_j^1 = \sum_{i=0}^{n0} w_{ji}^1 \cdot O_i^1; \qquad \overline{I}^1 = (I_1^1, I_2^1, \ldots\ldots, I_{n1}^1) \tag{4.17}$$

$$\overline{O}^2 = \left[F(I_1^1), F(I_2^1), \ldots\ldots, F(I_{n1}^1)\right]$$

w_{ji}^1, i. giriş düğümü ile birinci katmandaki j. düğüm arasındaki bağlantıyı; n1 ise 1. saklı katmandaki düğüm sayısını, n0 giriş boyutunu, F(.) ise kırpıcı fonksiyonunu göstermektedir.

Birinci katmandaki çıkışlar, öznitelik uzayında oluşan bölgelerin (örneğin şekil 4.5'te bölgeler: c_0, c_1, c_2....) kodlarını temsil etmektedir. İkinci katmanda ise bölge kodları tek bir düğüme dönüştürülür. Dönüştürme işlemi, bir lojik AND işlemi ile gerçeklenebilir. Bu durumda öğrenme sırasında sadece seçilen eğitim kümesi ile belirlenen bölgeler tanımlanır. Giriş vektörünün pozisyonu, önceden tanımlanan bölgelerin dışında ise ikinci katmanda hiçbir düğümün aktif olmadığı gözlemlenir ve giriş işareti için bir sınıf bulunamaz. Eğitim kümesi dışındaki girişlere ait sınıfların bulunamaması, ağın genelleme yeteneğinin kullanılmadığını göstermektedir.



Şekil 4.6 GetÇKA'nın yapısı

İkinci katmanın AND işlemini gerçeklemesi yerine, herbir çıkış düğümüne olan mesafelerinin araştırılması ve en minimum mesafeye sahip düğüm çıkışının aktif yapılması, ağın genelleme yeteneğini artıracaktır. Böylece ikinci katmanda önceden belirlenen ve bölgeleri temsil eden kodlar, ağın ağırlıklarını oluşturur. İkinci katmandaki düğümlerin çıkışları aşağıdaki gibi hesaplanır.

$$D_j = \text{Hamming uzaklığı } (\overline{O}^2, \overline{W}_j^2) \qquad O_e^3 = \begin{cases} 1 & D_e = \min_j \{D_j\} \\ 0 & \text{aksi taktirde} \end{cases} \qquad (4.18)$$

$$\text{Çıkış katmanı} = C_c = \sum_e \overline{W}_c^3 \cdot \overline{O}^3$$

$\overline{O}^k$, (k-1). katman çıkış vektörünü; $\overline{W}_j^k$, k. saklı katmandaki j. düğümünün ağırlık vektörünü göstermektedir. Üçüncü katman, aynı sınıfa ait çıkış düğümlerinin lojik olarak OR'lanmasından ibarettir.

Yapının ilk katmanı, hiper düzlemleri temsil eden düğümler ile hacimsel bölgeler oluşturmak; ikinci katmanı, minimum mesafe ölçüsünü hesaplamak ve üçüncü katmanı ise aynı sınıfa ait aktif düğümleri tek bir çıkışa yönlendirmek için kullanılır. Eğitim tamamlandıktan sonra birinci katman, gelen giriş vektörünün içine düştüğü bölgenin kodunu üretir. İkinci katman, bu vektöre önceden bulunan kodların en yakın olanını tespit eder ve bulunan koda ait çıkış düğümünü aktif yapar. Üçüncü katman, ikinci katmanda aktif olan düğümün sınıfını belirler.

4.5.3 GetÇKA'nın Eğitimi

Genetik havuzdaki diziler bir önceki bölümde tanımlandığı gibi tüm ağı temsil etmez. Genetik algoritmalar, birinci katmandaki düğümlerin ağırlık değerlerini bulmaktadır, diğer katmandaki düğümlerin ağırlıklarını bulmak için kullanılmamaktadır. Havuz içinde uyumluluk fonksiyonunu en iyi sağlayacak tek bir hiper düzlem araştırılır. Şekil 4.7'de GetÇKA'nın eğitimi için kullanılacak genetik havuz gösterilmiştir.

Ağın birinci katmanındaki düğümler ihtiyaca göre tek tek belirlenir. Böylece çok katmanlı ağın düğüm sayısı eğitim sırasında otomatik olarak belirlenmektedir. Birinci saklı katmandaki düğümlerin giriş uzayındaki bölgeleri kodladıkları belirtilmişti. Eğitim kümesi içindeki vektörler bir bölge içinde aynı sınıftan değil ise, eğitim sonrası bazı giriş değerleri için doğru sınıflama sonuçları üretilemeyecektir. Dolayısıyla uyumluluk fonksiyonu, eğitim sırasında seçilen hiper düzlemler ile oluşan hacimsel bölgelerin içinde aynı sınıftan vektörlerin bulunmasını temin edecek şekilde tanımlanır. Amaç, yeni bir hiper düzlem ilavesi ile oluşan bölgelerde maksimum sayıda aynı sınıftan vektör elde etmektir. Amaç ölçütü için uyumluluk fonksiyonunun (UF) matematiksel ifadesi aşağıda verilmiştir.

$$\{E\}=\bigcup_{i=0}^{M-1}\{c_i\} \qquad \{c_i\}=\bigcup_{k=1}^{S}\{c_{ik}\}=\{c_{i1}\}\cup\{c_{i2}\}\cup.....\cup\{c_{iS}\}$$

$$MAX_i = \#\{c_{ij}\} = Max\ (\#\{c_{i1}\}, \#\{c_{i2}\}, ..., \#\{c_{iS}\})$$

$$UF = \sum_{i=0}^{M-1} \frac{MAX_i}{\#\{E\}} \qquad (4.19)$$

Denklemlerde görülen {E}, eğitim kümesini; $\{c_{ik}\}$, i. bölge içinde k. sınıfa ait vektörlerin kümesini; M, oluşan toplam bölge sayısını ve S, sınıf sayısını göstermektedir.

Başlangıçta havuz, birinci katmandaki ilk düğümün ağırlıklarını temsil eden dizilerden oluşturulur. Her nesilde havuz içindeki tüm dizilere 4 genetik işlem uygulanır. 10 nesil sonunda en yüksek uyumluluk değerinin 0.95'in üstünde olup olmadığı incelenir. Bu başarım sağlanmamışsa, en iyi uyumluluk değeri veren düzlem ağın ilk düğümü olarak havuzdan alınır ve ağın sabitleşmiş düğümü olarak atanır. Gelecek nesillerde bu düğümün ağırlıkları araştırılmaz, ancak uyumluluk değeri hesabında kullanılır. İkinci düğümü temsil etmek üzere rastgele değerler ile havuz yeniden oluşturulur. İkinci düğümü bulmak için uyumluluk değeri hesabında birinci sabit düğüm ile ikinci aday düğümlerin temsil ettikleri hiper düzlemler tek tek kesiştirilir. Kesişim sonucu oluşan bölgeler için uyumluluk değeri hesaplanır.

Düğüm	Öteleme	1. boyut	2. boyut		n. boyut
1. aday →	w_0	w_1	w_2		w_n
i. aday →	.....	.....	.....	.....	.....
16. aday →	8 ikili	8 ikili	.....	.....	101...111

Şekil 4.7 GetÇKA'nın eğitiminde kullanılan genetik havuz

GetÇKA'nın eğitiminde kullanılan adımlar aşağıda verilmiştir.

Adım 1) Rastgele değerler ile genetik havuzu oluştur. Nesil sayısını belirle.

Adım 2) Önceden bulunan ağ hiper düzlemleri (sabitleşmiş düğümler) ile havuzdaki dizilerin temsil ettiği hiper düzlemleri tek tek kesiştir ve herbir dizi için uyumluluk değeri hesapla.

Adım 3) Havuz içindeki tüm dizilere kopyalama, çaprazlama ve mutasyon işlemlerini uygula. Nesil sayacını 1 azalt. Nesil sayacı sıfıra eşit değilse adım 2'ye git.

Adım 4) Nesil sayacını başlangıç değeri ile kur. Havuz içinde en yüksek uyumluluk değerini veren diziyi bul.

Adım 5) Uyumluluk değeri 0.95 değerinden büyük ise eğitim algoritmasını sonlandır.

Adım 6) En iyi uyumluluk değerini veren diziyi, ağın sabitleşmiş bir düğümünü oluşturmak üzere havuzdan al. Adım 1'e git.

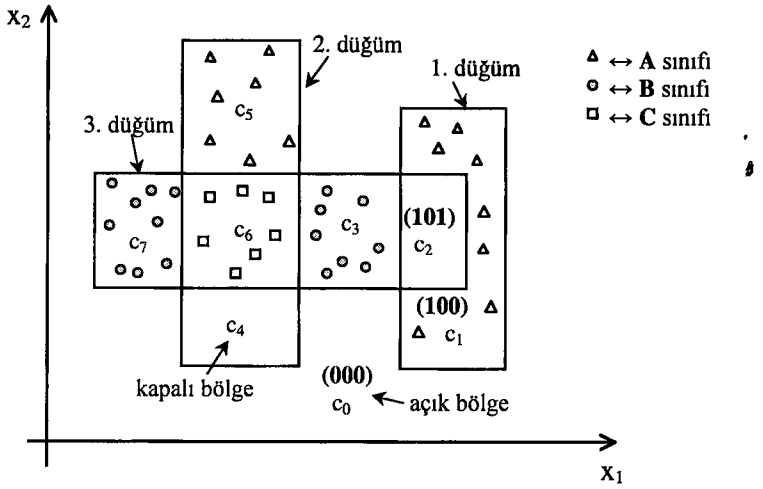
4.6 Prizma Ağının Genetik Algoritmalar ile Eğitimi

Çok katmanlı ağın birinci katmanındaki düğümler yardımıyla öznitelik uzayını bölmeleme yeteneği bir önceki bölümde analiz edilmiştir. Ağın tüm çıkış düğümlerinde kırpıcı kullanımı varsayımı ile yapılan bu incelemeler, birinci katmandaki düğümler yardımıyla uzayı bölmeleme yeteneğinin sınıflama başarımı için son derece önemli olduğunu göstermiştir.

Genetik algoritmalar ile eğitilen prizma ağının (GetPrz) birinci katmanındaki düğümleri, çok boyutlu öznitelik uzayında dikdörtgensel bir hiper prizma şekline sahiptir [48]. Ağın tek bir düğümü bile öznitelik uzayında kapalı bir bölge oluşturabilmektedir. Ağın birinci katmanındaki düğümlerin tanımladığı bölgelerin öznitelik uzayında kesiştirilmeleri, kapalı bölgelerin sayısının artmasına neden olur. Geliştirilen yeni ağ, aynı boyutta ve düğüm sayısında çok katmanlı ağdan daha fazla bölge üretebilme yeteneğine sahiptir.

4.6.1 GetPrz Ağı ile Öznitelik Uzayının Bölmelenmesi

Birinci katmandaki düğümlerin temsil ettiği bölgeler iki parametre ile tanımlanmaktadır [48]: Boyut merkezi ve boyut aralığı. Ağın birinci katmanındaki düğümler, iki boyutlu uzayda dikdörtgensel bir alan temsil etmektedir. n boyutlu öznitelik uzayında ağın birinci katmanındaki her bir düğüm 2×n adet parametre ile temsil edilir. Şekil 4.8'de GetPrz ağının birinci katmanındaki düğümlerin iki boyutlu öznitelik uzayında görünümü verilmiştir.

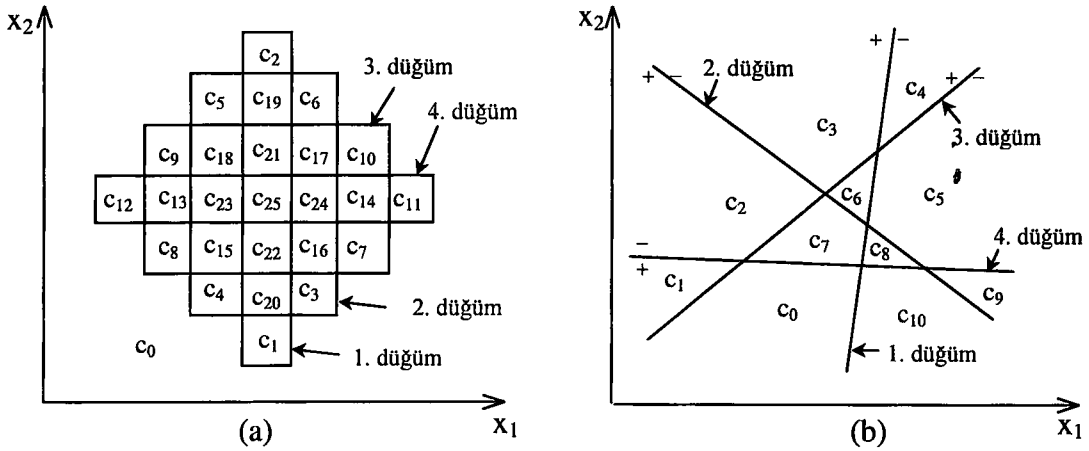


Şekil 4.8 İki boyutlu uzayda GetPrz ağının birinci katmanındaki düğümleri

İki boyutlu örnek uzayda 3 düğüm (dikdörtgensel alan) kullanılmıştır; ancak alanların kesişimleri sonucu oluşan görüntü, 3'ten fazla dikdörtgensel alan kullanılarak elde edildiği izlenimini vermektedir. Bu analiz, düğümlerin temsil ettiği bölgelerin öznitelik uzayında kesiştirilmeleri ile yeni kapalı bölgelerin üretilebileceğini göstermektedir.

Önceki bölümde kesiştirme işleminin avantajı incelenmiştir. Birinci katmandaki düğümlerin temsil ettiği hiper düzlemlerin kesiştirilmesi ile öznitelik uzayında çok sayıda bölge elde edilmiştir. GetPrz ağının birinci katmanındaki düğümlerin temsil ettiği bölgelerin kesiştirilmesi ile oluşan kapalı bölgelerin sayısı, hiper düzlemlerin kesişimi sonucu üretilen bölgelerin sayısından daha büyüktür. Şekil 4.9'da çok katmanlı ağ ile GetPrz ağının birinci katman çıkış düğümlerinin ürettiği bölgeler gösterilmektedir. Her iki ağ, 2 girişe ve birinci saklı katmanda 4 düğüme sahiptir.

Tablo 4.1'de her iki ağ tarafından üretilen kodlar (tek bölge kodu ve çoklu bölge kodu) ve bölgeler gösterilmiştir. Her iki ağda da düğümlerin sırası değiştirildiğinde bölgeleri temsil eden kodlar farklılaşmaktadır; ancak prizma ağı ile üretilen bölge sayısının, aynı boyut ve düğüm sayısında, ÇKA'dan daha fazla olduğu gözlenmektedir. Ayrıca, çoklu bölge kodu ile öznitelik uzayında prizma ağının oluşturduğu kapalı bölgelerin sayısının, üretilebilecek maksimum koddan (2^p, p=çıkış düğüm sayısı) daha fazla olduğu gözlenmektedir. Öznitelik uzayında çoklu bölge kodu ile birden fazla bölge temsil edilebilmektedir. Dolayısıyla öznitelik uzayında aynı sayıda çıkış kodu ile daha fazla bölge üretme imkanı elde edilmiştir.

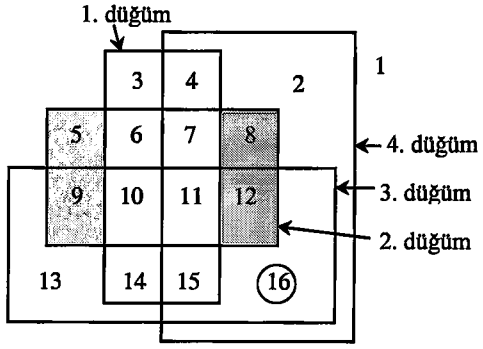


Şekil 4.9 a) GetPrz ağı ile b) ÇKA'nın birinci katman çıkış düğümlerinin ürettiği bölgeler

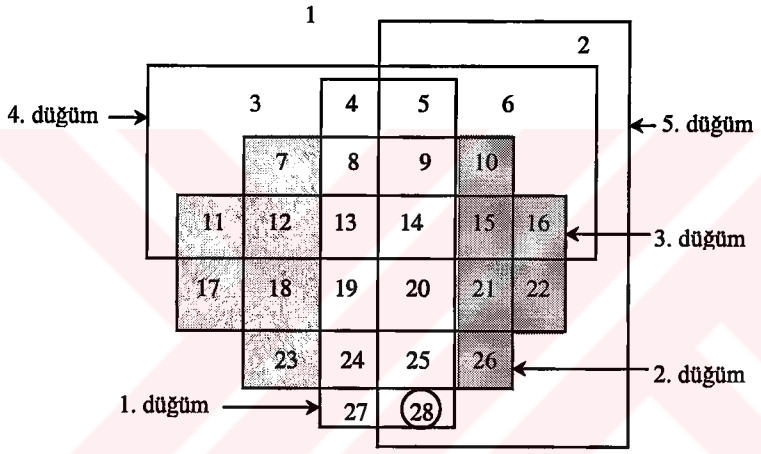
Tablo 4.1 GetPrz ve ÇKA'da oluşan bölge kodları

	GetPrz ağı için anlamlı kodlar		ÇKA için anlamlı kodlar	
	Bölgeler	Kodlar	Bölgeler	Kodlar
	c_0	0000	c_0	1101
Farklı 2 kapalı bölge 1 kodla temsil edilir	c_1, c_2	1000	c_1	1111
	c_3, c_4, c_5, c_6	0100	c_2	1110
	c_7, c_8, c_9, c_{10}	0010	c_3	1010
	c_{11}, c_{12}	0001	c_4	0010
	c_{13}, c_{14}	0011	c_5	0000
Farklı 4 kapalı bölge 1 kodla temsil edilir	c_{15}, c_{16}, c_{17}, c_{18}	0110	c_6	1000
	c_{19}, c_{20}	1100	c_7	1100
	c_{21}, c_{22}	1110	c_8	0100
	c_{23}, c_{24}	0111	c_9	0001
	c_{25}	1111	c_{10}	0101

4 ve 5 düğüm kullanılarak elde edilen bölgeler ve $C_{PMAX}(p,d)$ değeri gösterilmektedir.



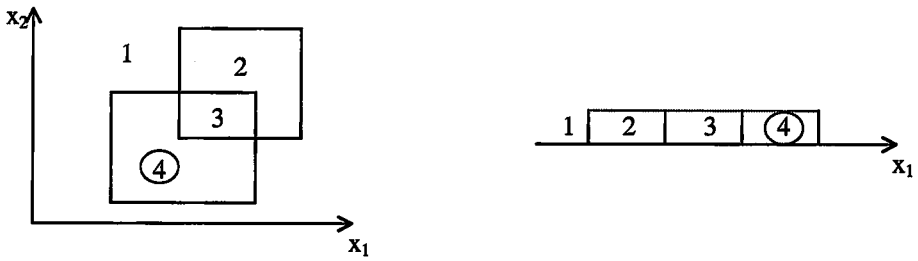
(a)



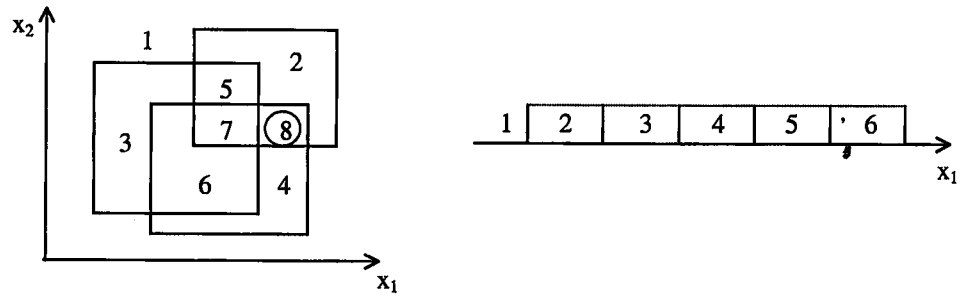
(b)

Şekil 4.10 İki boyutlu uzayda a)4, b) 5 düğüm kullanılarak elde edilen bölgeler

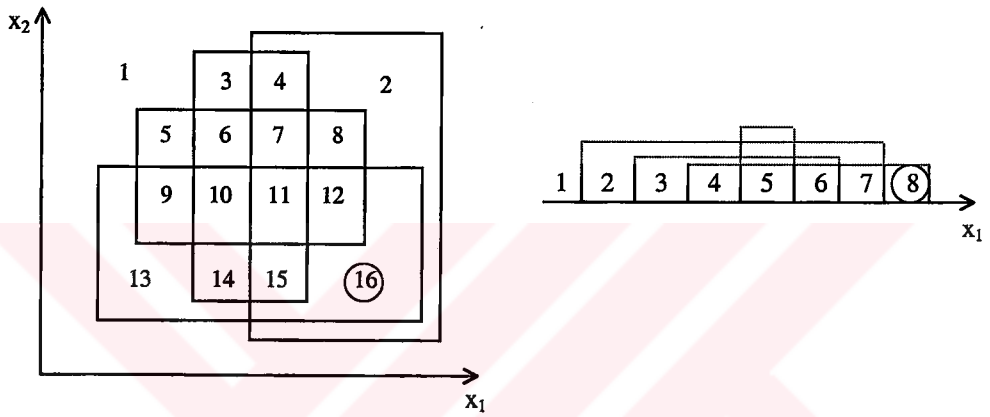
Şekil 4.11'de $C_{PRZ}(2,2)$ ve $C_{PRZ}(2,1)$'in değerleri, şekil 4.12'de $C_{PRZ}(3,2)$ ve $C_{PRZ}(3,1)$'in değerleri, şekil 4.13'te ise $C_{PRZ}(4,2)$ ve $C_{PRZ}(4,1)$'in değerleri gösterilmektedir. Bu değerleri kullanarak 4.20 denklemi ile $C_{PRZ}(5,2)$'yi ardışıl olarak bulalım ve gerçek değerlerle ($C_{PMAX}(p, d)$) karşılaştıralım.



Şekil 4.11 İki ve tek boyutlu uzaylarda 2 düğüm kullanılarak elde edilen bölgeler



Şekil 4.12 İki ve tek boyutlu uzaylarda 3 düğüm kullanılarak elde edilen bölgeler



Şekil 4.13 İki ve tek boyutlu uzaylarda 4 düğüm kullanılarak elde edilen bölgeler

Şekil 4.10'da 4 düğüm ile 16 ve 5 düğüm ile 28 bölge elde edildiği gösterilmiştir. Elde edilen sonuçlar aşağıdaki eşitsizliğin varlığını göstermektedir. Şekil 4.10'da gösterilen koyu gri bölgelerin sayısı, eşitsizliğe sebep olan fark ($C_{PMAX}(p,d) - C_{PRZ}(p,d)$) değerler olarak bulunmuştur. Bu bölgeler, şekil 4.10.a'da 4. ve 4.10.b'de 5. düğüm (en son ilave edilen düğümler) ile kesilmediği için $C_{PRZ}(p,d)$ ifadesi içinde gözükmemektedir. En kötü durumda bile (gerçek değer olan $C_{PMAX}(p,d)$ bulunamadığında $C_{PRZ}(p,d)$ değeri ile) hiper düzlemler kullanılarak oluşturulan bölge sayısından daha fazla bölge elde edilmektedir.

$$C_{CKA}(p,d) \leq C_{PRZ}(p,d) = C_{PRZ}(p-1,d) + C_{PRZ}(p-1,d-1) \leq C_{PMAX}(p,d)$$

$$C_{CKA}(3,2) = 7 < C(3,2) = C_{PRZ}(2,2) + C_{PRZ}(2,1) = 4 + 4 = 8 \leq C_{PMAX}(3,2) = 8$$

$$C_{CKA}(4,2) = 11 < C(4,2) = C_{PRZ}(3,2) + C_{PRZ}(3,1) = 8 + 6 = 14 \leq C_{PMAX}(4,2) = 14 + 2$$

$$C_{CKA}(5,2) = 16 < C(5,2) = C_{PRZ}(4,2) + C_{PRZ}(4,1) = 14 + 8 = 22 \leq C_{PMAX}(5,2) = 22 + 6$$

Prizma ağı kullanılarak oluşturulan tek bölge kodlarının sayısı, çok katmanlı ağ tarafından üretilen kodların sayısından fazladır. Çoklu bölge kodlarının sayısı çalışma içinde incelenmemiştir. Çünkü bir üst uzaya geçildiğinde çoklu kodlar ile temsil edilen bölgeler birleşmekte ve tek bölge haline gelmektedir. Bu bölgelerin birleşmesi sonucunda ortaya çıkan kapalı hiper hacimler, ağın öznitelik uzayını bölmeleme yeteneğini artırmaktadır.

4.6.2 GetPrz Ağının Yapısı

Sınıflar hakkındaki bilgi, GetPrz'nin düğümlerine dağıtılarak saklanmaktadır. Dolayısıyla herbir düğüm tüm sınıf dağılımı hakkında global bir bilgiye sahiptir. Bilginin bu şekilde dağıtılarak saklanması, ağın fiziksel olarak 2'den fazla katman ile gerçeklenmesine neden olmaktadır.

Şekil 4.14'te GetPrz ağının yapısı gösterilmiştir [48]. Ağ, üç katmandan meydana gelir. Öznitelik uzayında ilk katmandaki düğümlerin temsil ettiği hiper prizmaların kesiştirilmesi ile yeni kapalı hacimsel bölgeler oluşturulur. Birinci katmandaki düğümlerin çıkışları aşağıdaki gibi hesaplanır.

$$Y_j = \prod_{i=1}^{n} \left[U(x_i - w_{ji0}) - U(x_i - w_{ji1}) \right] \qquad U(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \qquad (4.21)$$

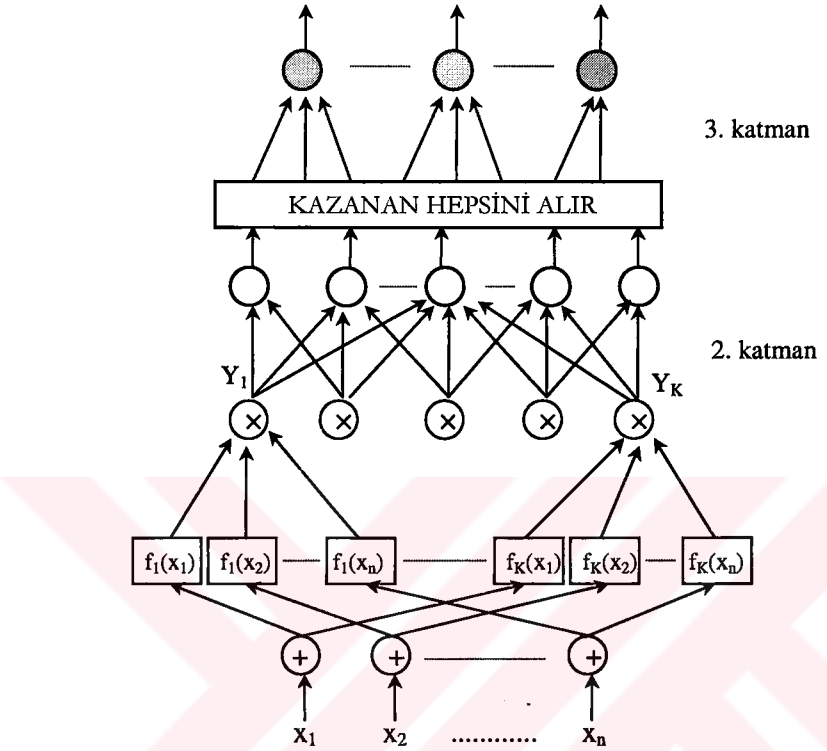
$$w_{ji0} = m_{ji} - g_{ji} \qquad w_{ji1} = m_{ji} + g_{ji}$$

m_{ji}, j. düğümün i. boyut eksenindeki merkez koordinatını; g_{ji} j. düğümün i. boyut eksenindeki aralık değerini tanımlamaktadır. Bu parametrelerin iki boyutta gösterilimi şekil 4.15'te verilmiştir.

Birinci katmandaki çıkışlar, öznitelik uzayında oluşan bölgelerin kodlarını temsil etmektedir. GetPrz ağı, GetÇKA gibi hibrit bir yapıya sahiptir. GetPrz ağının ikinci ve üçüncü katmanı bölüm 4.5.2'de tanıtılan yapıya aynen benzer.

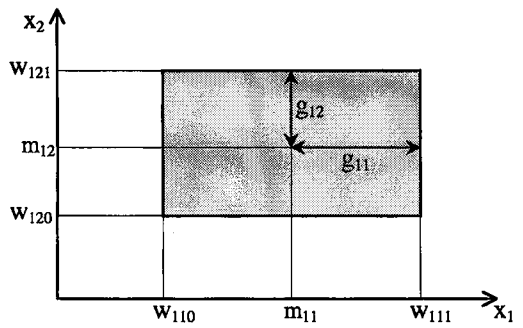
Yapının ilk katmanı, hiper prizmaları temsil eden düğümler ile hacimsel bölgeler oluşturmak; ikinci katmanı, minimum mesafe ölçüsünü hesaplamak ve üçüncü katmanı ise aynı sınıfa ait aktif düğümleri tek bir çıkışa yönlendirmek için kullanılır. Eğitim tamamlandıktan sonra birinci katman, gelen giriş vektörünün içine düştüğü

bölgenin kodunu üretir. İkinci katman, bu vektörün önceden bulunan kodlara en yakın olanı tespit eder ve bulunan koda ait çıkış düğümünü aktif yapar. Üçüncü katman, ikinci katmanda aktif olan düğümün sınıfını belirler.



$$f_1(x_1) = U(x_1 - w_{110}) - U(x_1 - w_{111}) \text{ , } \ldots\ldots \text{ , } f_K(x_n) = U(x_n - w_{Kn0}) - U(x_n - w_{Kn1})$$

Şekil 4.14 GetPrz ağının yapısı



Şekil 4.15 İki boyutta düğüm parametrelerinin gösterilimi

4.6.3 GetPrz Ağının Eğitimi

Genetik havuzdaki diziler tüm ağı temsil etmez. Genetik algoritmalar, birinci katmandaki düğümlerin ağırlık değerlerini (temsil edilen geometrik yapıların öznitelik uzayında pozisyonunu ve büyüklüğünü) araştırır; diğer katmandaki düğümlerin ağırlıklarını bulmak için kullanılmaz. Havuz içinde uyumluluk fonksiyonunu en iyi sağlayacak tek bir hiper prizmanın parametreleri araştırılır. Şekil 4.16'da prizma ağının eğitimi için kullanılacak genetik havuz gösterilmiştir. Havuz içinde n boyutlu uzayda bir dizi (düğüm), 2×n parametre ile temsil edilir. Her boyut için 2 parametreye ihtiyaç vardır: 1) Boyut merkezi, 2) Boyut aralığı.

Düğüm	1. boyut		2. boyut		.....	n. boyut	
1. aday →	w_{10}	w_{11}	w_{20}	w_{21}		w_{n0}	w_{n1}
i. aday →	.....		.....		.....	.....	
16. aday →	8	8	16 ikili		.....	8 ikili	8 ikili

Şekil 4.16 GetPrz ağının eğitiminde kullanılan genetik havuz

Eğitim sırasında ağın birinci katmanındaki düğümleri ihtiyaca göre otomatik olarak belirlenir. Birinci saklı katmandaki düğümlerin, giriş uzayında oluşan bölgeleri kodladıkları ifade edilmiştir. Bir bölge içindeki vektörler aynı sınıftan değil ise, eğitim sonrası bazı giriş değerleri için doğru sınıflama sonuçları üretilemeyecektir. Dolayısıyla uyumluluk fonksiyonu, eğitim sırasında seçilen hiper prizmalar ile oluşan bölgelerin içinde aynı sınıftan vektörlerin bulunmasını temin edecek şekilde tanımlanır. Amaç, yeni hiper prizmanın öznitelik uzayına ilavesi ile oluşan bölgelerde maksimum sayıda aynı sınıftan vektör elde etmektir. Uyumluluk fonksiyonunun matematiksel ifadesi GetÇKA ağının eğitiminde kullanılan fonksiyonla aynıdır.

Birinci katmandaki ilk düğümün parametrelerini bulmak için hiper prizma merkezi eğitim kümesinden olmak üzere aralıklar için rastgele değerler verilerek havuz oluşturulur. Her nesilde havuz içindeki tüm dizilere, 4 genetik işlem uygulanır. 10 nesil sonunda maksimum uyumluluk değerinin 0.95'in üstünde olup olmadığı

incelenir. Bu başarım sağlanmamışsa, en iyi uyumluluk değerini veren prizma, ağın ilk düğümü olarak havuzdan alınır ve ağın sabitleşmiş düğümü olarak atanır. Gelecek nesilde bu düğümün ağırlıkları araştırılmaz; ancak uyumluluk değeri hesabında kullanılır. Havuz ikinci düğümü bulmak için hazırlanır. Sabitlenen ağ düğümlerine havuzdaki 16 aday dizi ilave edilerek 16 adet yeni aday ağ elde edilir. Aday ağların içinde eski sabit ağ düğümleri ile genetik havuzda temsil edilen sadece bir yeni düğüm bulunmaktadır. Her aday ağ için bir uyumluluk değeri hesaplanır. Bu değer, bize aday ağların öznitelik uzayını bölmeleme yetenekleri hakkında bir bilgi vermektedir.

GetPrz ağının eğitim algoritması aşağıda verilmiştir.

Adım 1) Prizma merkezleri eğitim kümesinden olmak üzere aralıklara rastgele değerler vererek genetik havuzu oluştur. Nesil sayısını belirle.

Adım 2) Önceden bulunan ağ düğümleri ile havuz içinde temsil edilen düğümleri yukarda anlatılan şekilde birleştir. Her aday ağ için uyumluluk değeri hesapla.

Adım 3) Havuz içindeki tüm dizilere kopyalama, çaprazlama ve mutasyon işlemlerini uygula. Nesil sayacını azalt. Nesil sayacı sıfıra eşit değilse adım 2'ye git.

Adım 4) Havuz içinde en yüksek uyumluluk değeri veren diziyi bul.

Adım 5) Uyumluluk değeri 0.95 değerinden büyük ise eğitim algoritmasını sonlandır.

Adım 6) En iyi uyumluluk değerini veren diziyi havuzdan al ve bu diziyi ağın birinci katmanında sabitleşmiş yeni bir düğüm olarak ata. Adım 1'e git.

4.7 RCE Ağının Genetik Algoritmalar ile Eğitimi

RCE ağı iki katmandan meydana gelir. Birinci katmanındaki düğümler, öznitelik uzayında hiper küreleri temsil etmektedir. Hiper kürelerin merkezleri eğitim kümesinden direkt alınan vektörlerden meydana gelir ve eğitim sırasında değiştirilmez. Yarıçaplar ise, hiper kürelerin birbirleriyle kesişme yapmamasını temin edecek şekilde eğitim sırasında belirlenir. Bu durumda bir düğüm öznitelik uzayında sadece bir kapalı hacimsel bölge oluşturur.

Genetik olarak eğitilen RCE (GetRCE) ağının birinci katmanındaki düğümlerinin öznitelik uzayında kesiştirilmeleri, kapalı bölgelerin sayısının artmasına neden olur. GetRCE ağı, aynı boyutta ve düğüm sayısında RCE ağından daha fazla bölge üretebilme yeteneğindedir [19].

Bir önceki alt bölümde incelenen eğitim algoritması ve öznitelik uzayında oluşturulan anlamlı bölge sayısı bu yapı için de geçerlidir. Ancak birinci katmandaki düğümler, öznitelik uzayında hiper küreleri temsil etmektedir. Uyumluluk fonksiyonu aynıdır, genetik havuzdaki dizilerde 2×n yerine n+1 parametre bulunmaktadır. Ancak GetPrz ağı, karmaşık uzay dağılımlarını daha iyi temsil edebilme yeteneğine sahiptir. Hiper kürelerde hacim, sadece bir r yarıçap değeri ile kontrol edilmektedir. GetPrz ağında ise hacim, n adet aralık parametresi ile kontrol edilmektedir.

4.8 Nicemleme Ağının Genetik Algoritmalar ile Eğitimi

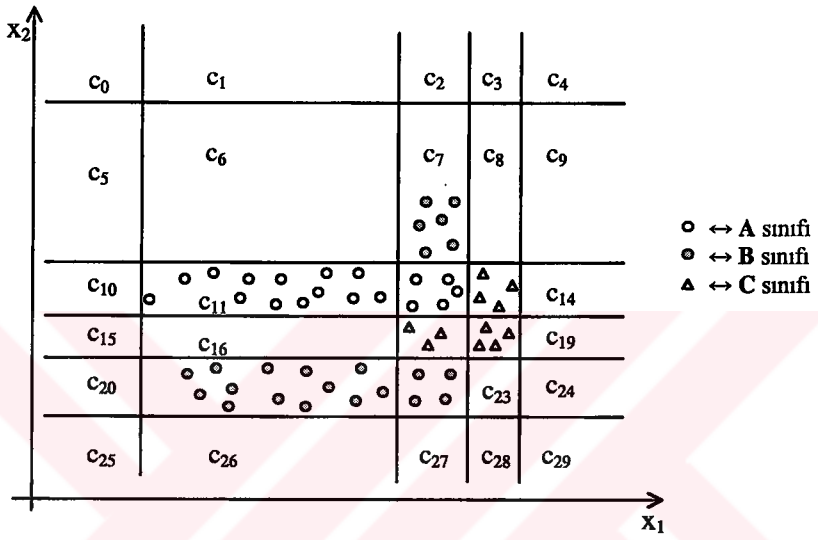
Bir önceki bölümde prizma ağının, öznitelik uzayını, çok katmanlı ağa göre aynı sayıda düğümle daha fazla bölmeleyebildiği gözlenmiştir. GetNic ağı ile öznitelik uzayı, daha az düğümle daha fazla bölmelenebilecektir. Aynı zamanda ağın fiziksel olarak gerçeklenmesi de son derece basittir.

Nicemleme ağı, öznitelik uzayını hiper düzlemlerle bölmeler; ancak hiper düzlemler n boyutlu uzayda n-1 eksene paralel, 1 eksene diktir. Düğümlerde kullanılan parametre sayısı daha az olmasına rağmen hiper düzlemlerin kesişimi ile oluşturulan bölge sayısı çok katmanlı ağ ve prizma ağına göre daha fazladır.

4.8.1 GetNic Ağı ile Öznitelik Uzayının Bölmelenmesi

Ağın birinci katmanındaki düğümler, iki boyutlu uzayda bir eksene paralel, diğer eksene dik bir doğruyu gösterir. Ağın birinci katmanındaki bir düğüm, n boyutlu öznitelik uzayında n-1 eksene paralel, 1 eksene dik bir hiper düzlem tanımlamaktadır. Dolayısıyla n boyutlu uzayda M adet düğüm, M adet parametre kullanılarak temsil edilebilir. Şekil 4.17'de nicemleme ağının birinci katmanındaki düğümlerin öznitelik uzayını bölmelemesi gösterilmiştir.

İki boyutlu örnek uzayda doğrularla kapalı bir alan oluşturabilmek için minimum 3 doğru kullanılması gerekir. Her doğru için 3 parametre gerekirse (özel durumlar hariç: doğruların eksenlere paralelliği), kapalı bir alan için toplam 9 parametre kullanılacaktır. Şekil 4.17'de bir boyut için 5, diğeri için 4 olmak üzere toplam 9 düğüm kullanılmıştır. Yeni ağ kullanılarak öznitelik uzayı, 9 parametre (düğüm) ile 12'si kapalı, toplam 30 bölgeye ayrılabilmiştir.



Şekil 4.17 GetNic ağı ile uzay bölmele

Bir anlamda yapılan işlem, öznitelik uzayı boyutlarının nicemlenmesi, ayrıklaştırılması olarak görülebilir. Tüm boyutlarda yapılan ayrıklaştırma, nicemleme işlemine benzemektedir. Çok boyutlu öznitelik uzayında ağın birinci katmanındaki düğümlerin kesişmeleri ile oluşturdukları şekil, prizma ağının birinci katmanındaki düğümlerin oluşturduğu şekillere benzemektedir. Ancak yeni ağda birinci katmandaki düğümler tek başlarına öznitelik uzayında kapalı hacimsel bölge oluşturamaz.

GetNic ağının birinci katmanındaki düğümlerin temsil ettiği kısıtlı hiper düzlemlerin kesiştirilmesi ile oluşan hacimsel bölgelerin sayısı, aynı sayıda parametre ile oluşturulan hiper düzlemlerin kesişimi sonucu üretilen hacimsel bölgelerin sayısından daha büyüktür. Eşitlik 4.14'te çok katmanlı ağ kullanıldığında p adet düzlem ile d boyutlu öznitelik uzayında oluşan bölge sayısının matematiksel ifadesi verilmiştir. ÇKA'nın oluşturduğu bölge sayısı ($C_{CKA}(p,d)$) aşağıdaki şekilde bulunmuştur:

$$C_{CKA}(p,d) = \begin{cases} 2^p & p \le d \\ \sum_{i=0}^{d} \binom{p}{i} & p > d \end{cases} \quad ; \quad \binom{p}{i} = \frac{p!}{i!(p-i)!} \tag{4.22}$$

GetNic ağının d boyutlu öznitelik uzayında p adet düzlem ile oluşturduğu bölge sayısı:

$$C_{NIC}(p,d) = \prod_{i=1}^{d} (p_i + 1) \qquad p = \sum_{i=1}^{d} p_i \tag{4.23}$$

şeklinde ifade edilmektedir. Burada p_i olarak belirtilen değerler, öznitelik uzayının i. boyutuna karşılık gelen eksenini nicemleyen düzlemlerin sayısıdır.

Maksimum sayıda bölge oluşturabilmesi, toplam düzlem sayısının herbir boyut eksenine olabildiğince eşit paylaştırılması ile mümkündür. d boyutlu öznitelik uzayında herbir boyut ekseni, toplam p adet düzlemin p/d tanesi ile nicemlenirse maksimum sayıda bölge elde edilir ($\forall$ i≠j için $|p_i - p_j| \le 1$ ise).

$$C_{NIC}(p,d) \ge \left(\text{int}\left(\frac{p}{d}\right) + 1 \right)^d \tag{4.24}$$

ÇKA'nın birinci saklı katmanındaki p adet düğüm, öznitelik uzayında p adet düzleme karşılık gelmektedir. Öznitelik uzayı d boyutlu ise, herbir düzlem (d+1) adet parametreyle temsil edileceğinden, p adet düğüm p·(d+1) adet parametreyle temsil edilecektir. GetNic ağında ise düğümlerin temsil ettiği düzlemlere kısıt getirildiği için bir düzleme (düğüm) karşılık tek bir parametre kullanılır (bu parametre, düzlemin dik olarak kestiği boyut eksenindeki pozisyonudur). ÇKA'da birinci saklı katmandaki düğümlerin tümü için p·(d+1) adet parametre gerekliyken, p·(d+1) adet parametre GetNic ağında p·(d+1) adet düzlemin kullanılmasına imkan vermektedir. Kullanılan parametre sayısı bakımından, d boyutlu uzayda ÇKA'da oluşturulan p adet düzlemin, GetNic ağında karşılığı p·(d+1) adet düzlemdir.

Bilindiği gibi düzlem (düğüm) sayısının artması uzayın daha çok bölmeleneceği ve böylece sınıf sınırlarında çözünürlüğün artacağı anlamına gelmektedir. Dolayısıyla

GetNic ağı daha fazla düzlem kullanarak öznitelik uzayını, ÇKA'dan daha fazla bölgeye ayırabilme yeteneğindedir.

Eşit sayıda parametre kullanılarak $C_{NIC} > C_{CKA}$ olduğunu göstermek mümkündür. Her iki ağdaki parametre sayılarının (PS) eşitliğinden;

$$PS_{CKA} = PS_{NIC} = p\cdot(d+1)$$

GetNic'deki düzlem sayısı = GetNic'deki parametre sayısı = $PS_{NIC} = p' = p\cdot(d+1)$

GetNic ağında 4.24 eşitliğinden;

$$C_{NIC}(p',d) \geq \left(\text{int}\left(\underbrace{\frac{p.(d+1)}{d}}_{m} \right) + 1 \right)^d$$

(d+1)/d çarpanı her zaman 1'den büyüktür. Bu çarpanı ihmal edip, $m \cong p$ alarak üstteki eşitliği tekrar düzenleyelim;

$$C_{NIC}(p',d) \geq (1+p)^d$$

şeklinde yazılabilir. $(1+p)^d$ 'nin açılımını yazalım;

$$(1+p)^d = \binom{d}{0}\cdot 1^d + \binom{d}{1}\cdot 1^{d-1}\cdot p^1 + \binom{d}{2}\cdot 1^{d-2}\cdot p^2 + \ldots\ldots\ldots + \binom{d}{d}\cdot 1^0\cdot p^d$$

$$C_{NIC}(p',d) \geq 1 + \binom{d}{1}\cdot p^1 + \binom{d}{2}\cdot p^2 + \ldots\ldots\ldots + p^d \qquad (4.25)$$

Eşitlik 4.22'den p>d için;

$$C_{CKA}(p,d) = \sum_{i=0}^{d}\binom{p}{i} \qquad p>d$$

olduğu belirtilmiştir. Yukardaki ifadeyi açık olarak yazalım;

$$C_{CKA}(p,d) = \binom{p}{0} + \binom{p}{1} + \ldots\ldots\ldots + \binom{p}{d} = 1 + \binom{p}{1} + \ldots\ldots\ldots + \binom{p}{d}$$

$$C_{CKA}(p,d) = 1 + p + \frac{p.(p-1)}{2.1} + \ldots\ldots\ldots + \frac{p.(p-1).(p-2)....(p-(d-1))}{d.(d-1).(d-2).....3.2.1} \qquad (4.26)$$

4.25'teki $\binom{d}{i}$'li terimler 1'den büyük sayılar olmasına rağmen bu terimleri ihmal edip C_{NIC}'i tekrar hesaplayalım;

$$C_{NIC}(p',d) \geq 1 + p^1 + p^2 + \ldots\ldots\ldots + p^d \qquad (4.27)$$

4.26 eşitliği ile 4.27'nin benzerliğinden de kolayca görülebileceği gibi her iki eşitlikte (d+1) adet terim vardır ve terimleri aynı sırada bire bir eşleştirdiğimizde GetNic'in ilk iki teriminden sonraki (ilk iki terim eşittir: 1 + p) tüm terimleri daima ÇKA'daki terimlerden büyüktür.

$$p^2 = p.p > \frac{p.(p-1)}{2.1}$$

$$p^3 = p.p.p > \frac{p.(p-1).(p-2)}{3.2.1}$$

$$\vdots$$

$$p^d > \frac{p.(p-1).(p-2)....(p-(d-1))}{d.(d-1).(d-2).....3.2.1}$$

4.27 eşitliği elde edilinceye kadar ihmal edilen terimler de göz önünde bulundurulacak olursa, C_{NIC}'in gerçek değeri 4.27'dekinin üzerindedir.

$p \leq d$ için de C_{NIC}'in C_{CKA}'dan büyük olduğunu gösterelim: 4.22 eşitliğinden, $p \leq d$ için;

$C_{CKA} = 2^p$

$$C_{NIC} \geq \left(\text{int}\left(\frac{p.(d+1)}{d} \right) + 1 \right)^d = (p+1)^d = h^d$$

Zaten $1 \leq p \leq d$ olduğu için her zaman $h^d \geq 2^p$ 'dir. Dolayısıyla; $p \leq d$ için

$C_{NIC} \geq C_{CKA}$ olduğu görülmektedir.

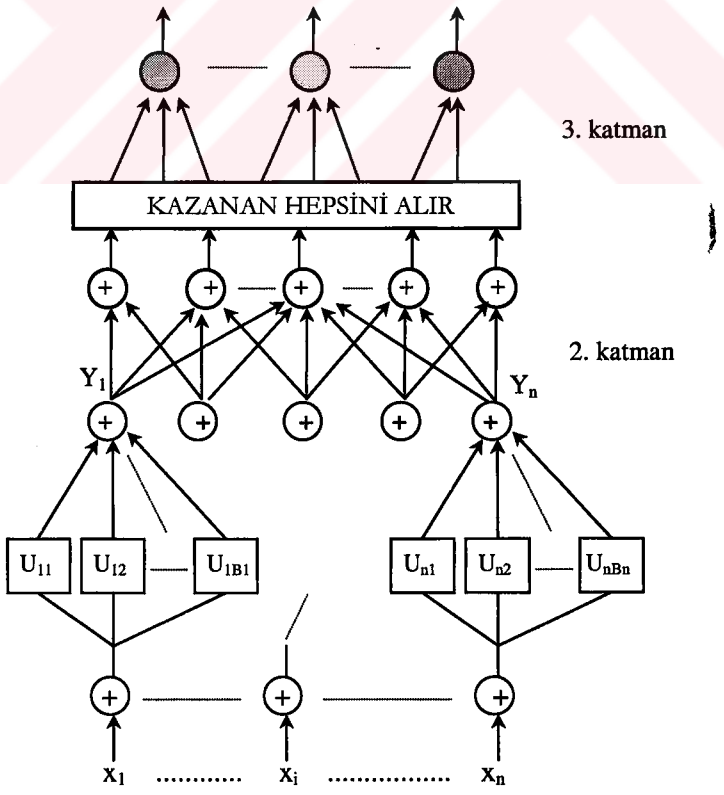
4.8.2 GetNic Ağının Yapısı

Sınıflar hakkındaki bilgi, GetNic'in düğümlerine dağıtılarak saklanmıştır. Dolayısıyla herbir düğüm tüm sınıf dağılımı hakkında global bir bilgiye sahiptir. Bilginin bu şekilde dağıtılarak saklanması, ağın fiziksel olarak 2'den fazla katman ile gerçeklenmesine neden olmaktadır.

Şekil 4.18'de GetNic ağının yapısı gösterilmiştir. Ağ, üç katmandan meydana gelir. Öznitelik uzayında ilk katmandaki düğümlerin temsil ettiği kısıtlı hiper düzlemlerin kesiştirilmesi ile hacimsel bölgeler oluşturulur. Birinci katmandaki düğümlerin çıkışları aşağıdaki şekilde hesaplanır.

$$U_{ij} = U(x_i - w_{ij}) \qquad Y_i = \sum_{j=1}^{B_i} U_{ij} \qquad i=1,2,...,n \qquad (4.28)$$

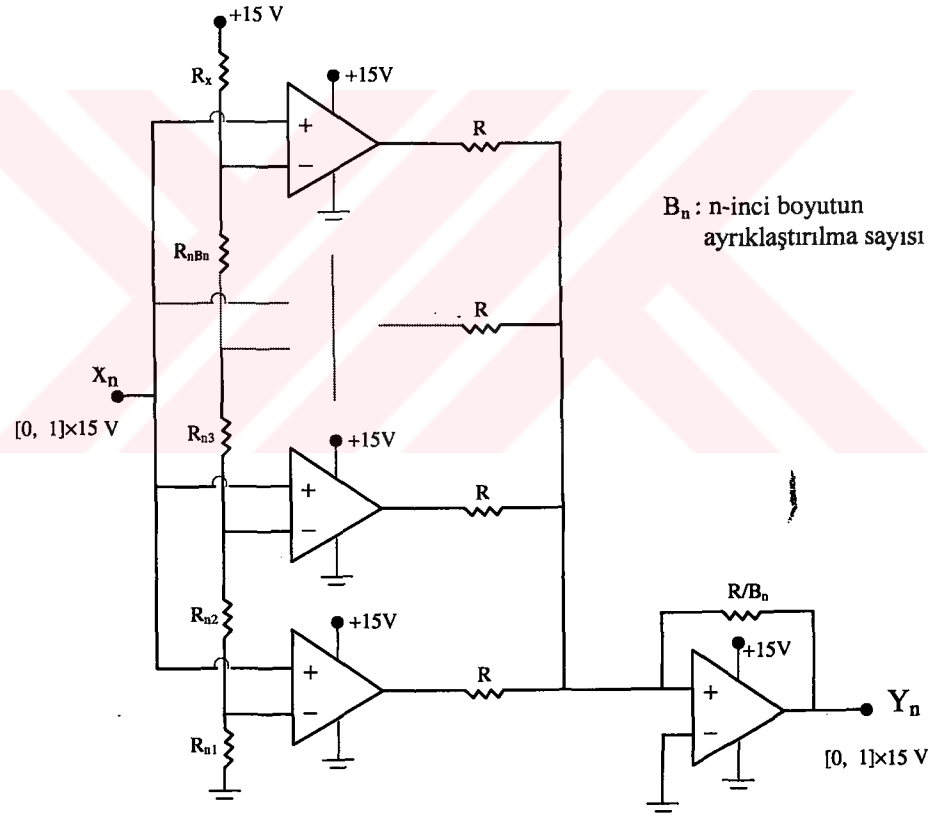
B_i , i-inci boyutun ayrıklaştırılma sayısı (i-inci boyuttaki düğüm sayısı), U(.) ise basamak fonksiyonudur.



Şekil 4.18 GetNic ağının yapısı

Birinci katman çıkışının boyutu diğer ağlara göre çok büyüktür. Boyut azaltılmadığı taktirde diğer ağlara göre çok daha fazla parametre kullanılacaktır. Ağın çıkış düğümleri, ilişkili oldukları giriş boyutuna (düzlemlerin dik oldukları eksenlere) göre gruplandırılır. Her grup içindeki düğümlerin çıkış değerleri (0 veya 1) toplanarak tek bir çıkış haline getirilir. Böylece ağın birinci katman giriş boyutu n ise çıkış boyutu da n olarak tanımlanır.

Sınıfların sınırlarının kabaca değiştiği uzay bölgeleri büyük aralıklarla, sınıf sınırlarının hızlı değişim gösterdiği uzay bölgeleri sık aralıklarla kodlanır. Birinci katmandaki bir grubun (x_n girişi ile Y_n çıkışı arasındaki yapı) fiziksel olarak gerçeklenmesi şekil 4.19'da gösterilmektedir.



Şekil 4.19 n-inci giriş düğümü ile ikinci katman girişi arasındaki bağlantı devresi

Birinci katman çıkışları, öznitelik uzayında oluşan bölgelerin kodlarını temsil etmektedir. GetNic ağı, GetÇKA, GetRCE ve GetPrz ağları gibi hibrit bir yapıya sahiptir. GetNic ağının ikinci ve üçüncü katmanı 4.5.2'de tanıtılan yapıya aynen benzemektedir.

Yapının ilk katmanı, hiper düzlemleri temsil eden düğümler ile hacimsel bölge oluşturmak; ikinci katmanı, minimum mesafe ölçüsünü hesaplamak ve üçüncü katman ise aynı sınıfa ait aktif düğümleri tek bir çıkışa yönlendirmek için kullanılır. Eğitim tamamlandıktan sonra birinci katman, gelen giriş vektörünün içine düştüğü hacimsel bölgenin kodunu üretir. İkinci katman, bu vektörün önceden bulunan kodlara en yakın olanı tespit eder ve bulunan koda ait çıkış düğümü aktif yapılır. Üçüncü katman, ikinci katmanda aktif olan düğümün sınıfını belirler.

4.8.3 GetNic Ağının Eğitimi

Genetik havuzdaki diziler tüm ağı temsil etmektedir. Genetik algoritmalar, birinci katmandaki düğümlerin öznitelik uzayında pozisyonunu (ağırlık değerleri) araştırır, diğer katmandaki düğümlerin ağırlıklarını bulmak için kullanılmaz. Havuz içinde uyumluluk fonksiyonunu en iyi sağlayacak sadece bir hiper düzlemin parametresi araştırılır; ancak ağın önceden bulunan düğümleri de havuz içindedir. Her yeni düğüm araştırmasında sadece bir eksen takımı seçilir. Dolayısıyla havuzdaki sütunlar, aynı eksene ait düğümleri göstermektedir. Şekil 4.20'de GetNic ağının eğitimi için kullanılacak genetik havuz gösterilmiştir.

Ağlar	1. boyut	2. boyut	.....	n. boyut	1.boyut	
1. ağ →	w_{11}	w_{21}		w_{n1}	w_{12}	
i. ağ →	.....	..... Boş	.....	..... Boş	.....	
16. ağ →	Boş	8 ikili	.....	.....	101...111	

Büyüme yönü →

Şekil 4.20 GetNic ağının eğitiminde kullanılan genetik havuz

Bu çalışmada uyumluluk fonksiyonu iki parçadan meydana gelmektedir. Yeni düğümler sadece bir eksen üzerinde araştırılır. Her 10 nesilde bu eksen sırayla değiştirilir. Bazı durumlarda eksenlerin ayrıklaştırılmasına ihtiyaç olmamasına rağmen, algoritmanın sırayla eksen seçmesi nedeni ile ağda anlamsız düğümler oluşur. Anlamsız düğümlerin ağdan çıkartılması için eğitim algoritmasına yeni bazı işlemler ve uyumluluk fonksiyonu eklenmiştir. Havuzdaki diziler, önceki alt

bölümlerde incelendiği gibi tek bir düğümden meydana gelmez. Bir düğüme bir parametre karşılık geldiği için diziler ağları (ağların tüm düğümlerini) temsil etmektedir.

Mutasyon işlemi, önceden öğrenilen dizilerdeki düğümlere uygulanır. Mutasyon işlemi, rastgele olarak bir dizideki bir düğümün kaldırılması veya kaldırılmış bir düğümün tekrar var edilmesi şekline dönüştürülmüştür. Bu işlem ve yeni seçilen ikinci uyumluluk fonksiyonu ile dizilerden anlamsız düğümler kaldırılır.

Birinci saklı katmandaki düğümlerin giriş uzayındaki hacimsel bölgeleri kodladıkları belirtilmişti. Bir bölge içindeki vektörler aynı sınıftan değil ise, eğitim sonrası bazı giriş değerleri için doğru sınıflama sonuçları üretilemeyecektir. Birinci uyumluluk fonksiyonu, eğitim sırasında seçilen hiper düzlemlerin kesişmesi ile oluşturulan bölgelerin içinde aynı sınıftan vektörlerin bulunmasını temin edecek şekilde tanımlanır. Amaç, yeni hiper düzlemin öznitelik uzayına ilavesi ile oluşan hacimsel bölgelerde maksimum sayıda aynı sınıftan vektör elde etmektir. Uyumluluk fonksiyonunun (UF) matematiksel ifadesi aşağıda verilmiştir.

$$\{E\}=\bigcup_{i=0}^{M-1}\{c_i\} \qquad \{c_i\}=\bigcup_{k=1}^{S}\{c_{ik}\}=\{c_{i1}\}\cup\{c_{i2}\}\cup.....\cup\{c_{iS}\}$$

$$MAX_i=\#\{c_{ip}\}=\text{Max}\,(\#\{c_{i1}\},\#\{c_{i2}\},.....,\#\{c_{iS}\})$$

$$UF_{j1}=\sum_{i=0}^{M-1}\frac{MAX_i}{\#\{E\}}$$

$$UF_{j2}=(DS_{max}-DS_j)/(DS_{max}-DS_{min})$$

$$UF_j=0.8\times UF_{j1}+0.2\times UF_{j2} \qquad (4.29)$$

Denklemlerde M hiper düzlemlerin kesişimi sonucu oluşan bölge sayısını, S sınıf sayısını, DS dizilerdeki düğüm sayısını, DS_{max} dizilerdeki maksimum düğüm sayısını, DS_{min} dizilerdeki minimum düğüm sayısını UF_{j1} ve UF_{j2} ise havuzdaki j. dizinin birinci ve ikinci uyumluluk fonksiyonlarını temsil etmektedir. Havuz içinde

dizi boyları sabittir; ancak mutasyon işlemi ile dizilerdeki bazı düzlemlerin kullanılması iptal edilmiştir (kullanılmaz etiketi ile).

Başlangıçta havuz, birinci katmanındaki ilk düğümün ağırlıklarını temsil eden dizilerden oluşturulur. Her nesilde havuz içindeki tüm dizilere 4 genetik işlem uygulanır. 10 nesil sonunda en yüksek uyumluluk değerinin 0.95'in üstünde olup olmadığı incelenir. Bu başarım sağlanmamışsa, en iyi uyumluluk değerini veren hiper düzlem ağın sabitleşmiş düğümü olarak atanır.

Gelecek nesillerde bu düğümün ağırlıkları yalnızca mutasyon işleminden etkilenmektedir. Havuz, ikinci düğümü temsil etmek üzere rastgele değerler ile tekrar oluşturulur. Bulunan dizinin boyu 1 artırılarak havuz içinde kopyaları oluşturulur. Sadece havuza ilave edilen bölgeler, rastgele değerler verilerek yeniden oluştururlur. Diziler ağların düğümlerini temsil etmektedir. Her dizinin uyumluluk değeri, o dizi içindeki hiper düzlemler birbirleri ile kesiştirilerek hesaplanır.

GetNic ağının eğitim algoritması aşağıda verilmiştir.

Adım 1) Havuza düşey bir sütun ilave et. Sırasıyla bir eksen takımı seç ve o eksen için ilave edilen sütuna rastgele değerler vererek genetik havuzu oluştur. Seçilen sütunda 4 düğümde bir, var olan düğümü boş olarak etiketle. Nesil sayısını belirle.

Adım 2) Havuzda diziler içinde temsil edilen hiper düzlemleri tek tek kesiştirerek, herbir dizi için birinci uyumluluk değerini hesapla. Havuzdaki diziler içinde minimum ve maksimum düğüm sayısını bul. Toplam uyumluluk fonksiyonunu hesapla.

Adım 3) Havuz içindeki tüm dizilere kopyalama, çaprazlama ve özel mutasyon işlemlerini uygula. Nesil sayacını azalt. Nesil sayacı sıfıra eşit değilse adım 2'ye git.

Adım 4) Havuz içinde en yüksek uyumluluk değeri veren diziyi bul.

Adım 5) Uyumluluk değeri 0.95 değerinden büyük ise eğitim algoritmasını sonlandır.

Adım 6) En iyi uyumluluk değerini veren diziyi tüm havuza kopyala ve adım 1'e git.

5. YAPAY SİNİR AĞLARININ SINIFLAMA SONUÇLARI

5.1 Öznitelik vektörleri

Bu bölüm içinde iki farklı sınıflama sonucu incelenecektir.

1) Örnek öznitelik uzayı
2) EKG vuruları

Her iki analizde de öğrenme işleminde kullanmak üzere ortak bir eğitim kümesi oluşturulur. Oluşturulan bu eğitim kümesi incelenecek tüm ağların eğitiminde kullanılır.

Birinci araştırma, ağların genelleme yeteneklerini, eğitim algoritmasının stratejisini ve öznitelik uzayında düğüm dağılımını görsel olarak incelemek için yapılmaktadır. İkinci araştırma, EKG işaretlerindeki vuru tiplerinin tespit edilmesine yöneliktir. Birinci ve ikinci inceleme arasında bir bağlantı yoktur, her ikisi de yukarda ifade edilen konulara yönelik bağımsız çalışmalardır.

Genetik olarak ağların eğitiminde üniform çaprazlama işlemi kullanılır. Genetik havuzun derinliği 16 olarak seçilmiştir. Mutasyon işlemi 32 ikilide bir gerçeklenmektedir.

5.1.1 Örnek Öznitelik Uzayı

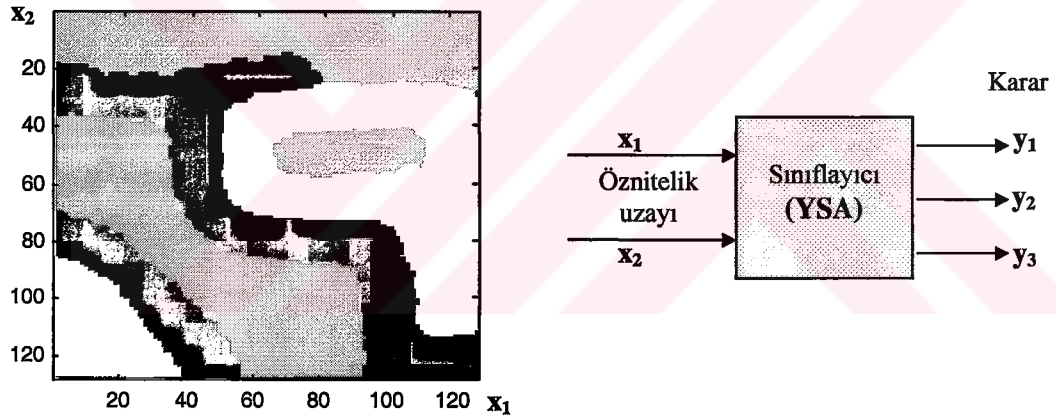
Örnek öznitelik uzayında, iki elemanlı 16384 (128×128) adet vektör bulunmaktadır. Eğitim kümesi bu uzaydan, sınıf dağılımları eşit olacak şekilde toplam 300 adet vektör alınarak oluşturulur. Test kümesi, 3 sınıfa ait tüm vektörlerden (siyah bölgeler hariç, 11691 adet vektör) meydana getirilmiştir.

Alt bölümler içinde, örnek öznitelik uzayının sınıflandırılması ve yeni geliştirilen ağlar ile benzeşen eski yapılar karşılaştırmalı olarak incelenecektir. Karşılaştırmada

üç parametre kullanılmaktadır:1) Test vektörlerinin doğru sınıflanma oranı, 2) Bağıl eğitim zamanı, 3) Düğüm sayısı.

Birinci analiz, ağın genelleme özelliği hakkında nicel bir değer vermektedir. İkinci analiz, ağların birbirlerine göre bağıl eğitim hızları hakkında bilgi vermektedir. Üçüncü analiz, eğitim sonrası ağda kullanılacak düğüm sayısı hakkında bilgi vermektedir.

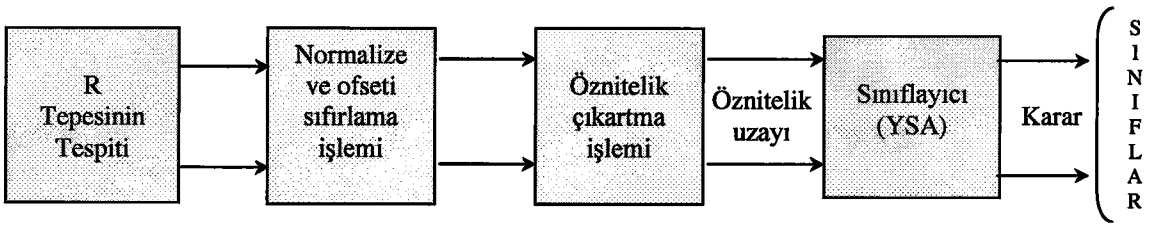
Şekil 5.1'de iki boyutlu örnek öznitelik uzayı ve bu uzayı sınıflamak için kullanılan blok gösterilmiştir. Test uzayında vektörlerin ait oldukları sınıflar beyaz, açık-gri ve koyu-gri olmak üzere üç tanedir. Siyah olarak görülen bölgeler herhangi bir sınıfa ait olmayan bölgelerdir. Tüm ağların 2 girişi ve sınıfları temsil eden 3 çıkışı bulunmaktadır. Öznitelik uzayı eksenlerindeki 1-128 değerleri ağa giriş olarak verilmeden önce 0-1 değerleri arasına normalize edilirler.



Şekil 5.1 Örnek öznitelik uzayı ve bu uzayın sınıflandırılması işlemi

5.1.2 EKG Vuruları

Şekil 5.2'de EKG vurularını belirlemek için kullanılan bloklar gösterilmiştir.



Şekil 5.2 EKG vurularının belirlenmesi

Birinci blok R tepesinin belirlenmesi için kullanılır. İkinci blok, 256 örneklik pencere içerisindeki EKG işaretinin tepeden tepeye genliğini 1 mV'a normalize etmek ve pencere içersindeki işaretin ofsetini yok etmek için kullanılır. Üçüncü blok, R tepesi dikkate alınarak EKG işaretinden özniteliklerin çıkartılması için kullanılır. Bu blok içinde iki farklı öznitelik çıkartma yönteminin başarıma etkisi incelenecektir: 1) Modül frekans spektrum değerleri, 2) Dalgacık katsayıları ve bu katsayıların oto-korelasyonları. Dalgacık temelli uygulamalar biyomedikal alanında hızlı bir gelişim göstermektedir. Bu nedenle modül frekans spektrum değerleri ile dalgacık katsayıları test aşamasında karşılaştırmalı olarak incelenecektir. Dördüncü blok, öznitelik vektörlerinin sınıflandırılması için kullanılır. Bu blok içinde sınıflayıcı olarak çalışma içinde geliştirilen yeni ağlar ve karşılaştırmak amacıyla bu ağlar ile benzeşen eski yapılar kullanılacaktır.

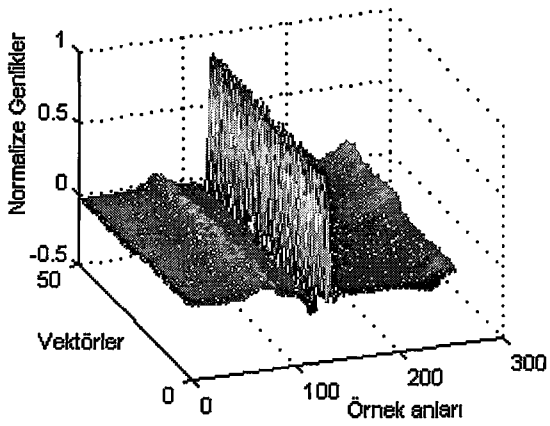
Test ve eğitim kümesi her sınıftan 50 adet vektör alınarak oluşturulmuştur. Şekil 5.3.a-j'de her sınıftan 50 adet vektör olmak üzere 10 farklı EKG dalga şekli gösterilmektedir. Çalışma içerisinde genlik değişimini yoketmek için EKG işaretinin tepeden tepeye değeri 1 mV'a normalize edilir ve işaretin ortalaması kaldırılır. EKG işaretlerinin yaşa, cinsiyete ve kişiden kişiye farklılıklar gösterdiği bilinmektedir. Şekil 5.3'te bu farklılıklar bazı vurularda daha belirgin olarak gözlenmektedir.

5.1.2.1 Modül Frekans Spektrum Değerleri

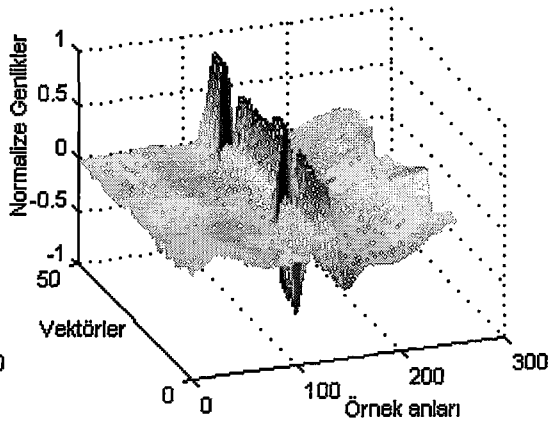
EKG işaretinden QRS kompleksi belirlenir ve bu bölgeden R tepesi bulunur. R tepesi etrafında sola doğru 128, sağa doğru 128 örnek alınarak bir pencere oluşturulur. Bu işlem ile pencere içerisinde sadece bir QRS kompleksi olması sağlanır. Pencere içerisinde kalan işarete, aşağıda ifadesi verilen ayrık Fourier dönüşümü (AFD) uygulanır [13, 14, 17, 18].

$$X(k) = \sum_{n=0}^{255} x(n) \cdot e^{\frac{-j2\pi kn}{256}}$$

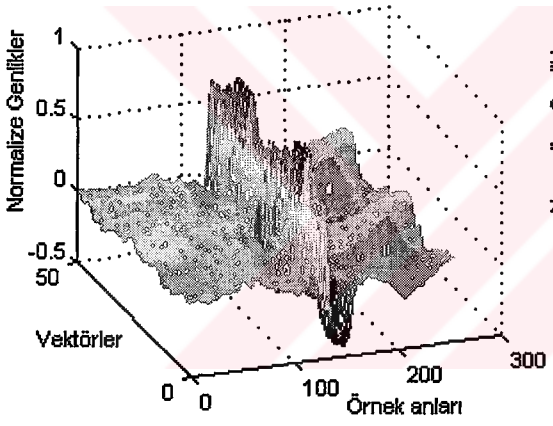
İşaretten elde edilen 128 frekans bileşeni içinde bazı değerlerin gürültü içerdiği ve 10 faklı EKG dalga şekli için anlamlı bilgi taşımadığı gözlenmiştir. Bu nedenle çalışma içersinde, bölüm 2.5'te anlatılan diverjans ve dinamik programlama kullanılarak 128 frekans spektrum bileşeni içinde anlamlı olarak gözlenen 15 ayrık bileşenin modül



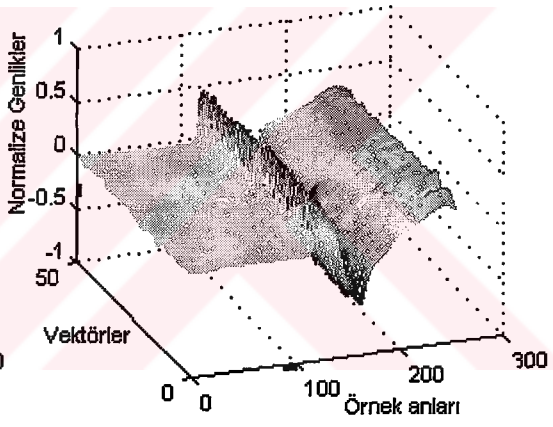
Şekil 5.3.a) Normal EKG vurusu



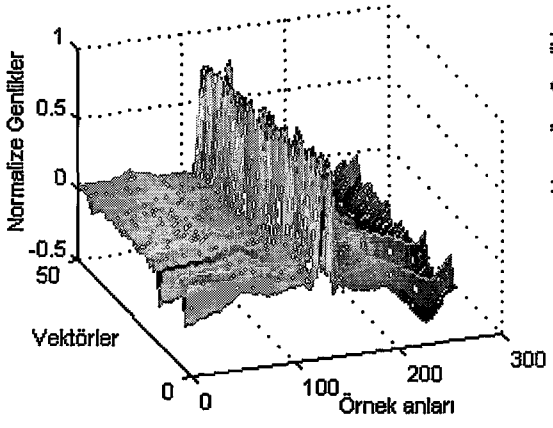
b) L tipi EKG vurusu



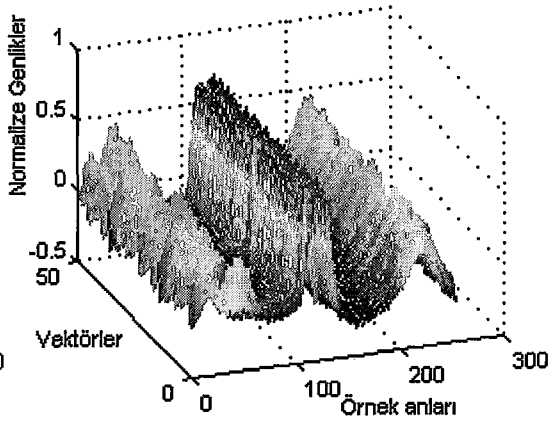
Şekil 5.3.c) R tipi EKG vurusu



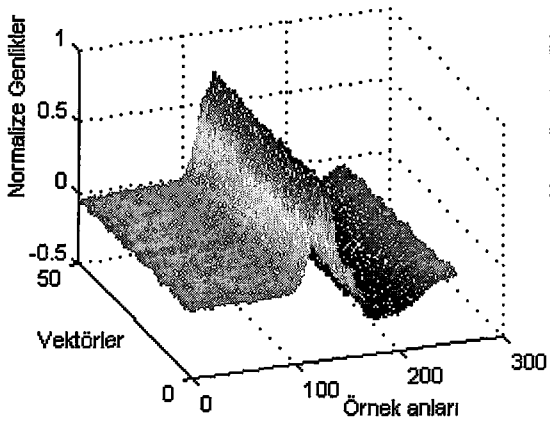
d) P tipi EKG vurusu



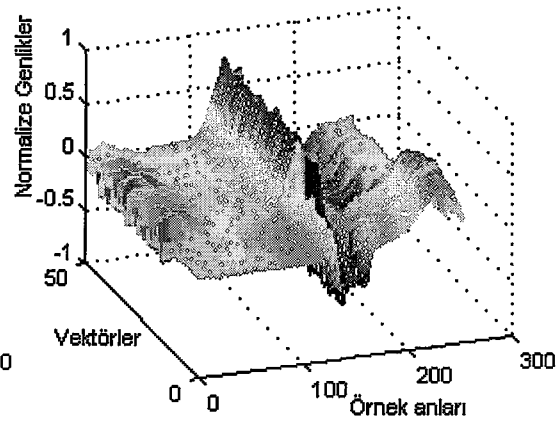
Şekil 5.3.e) p tipi EKG vurusu



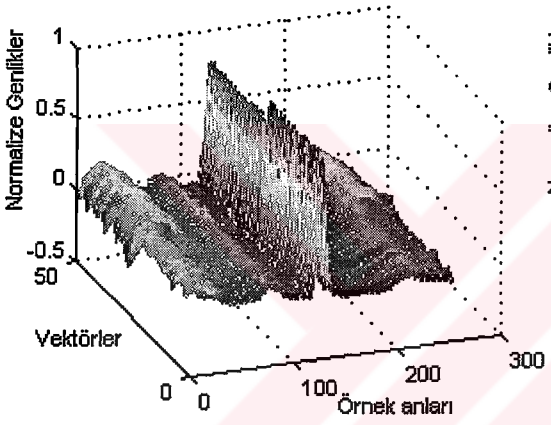
f) a tipi EKG vurusu



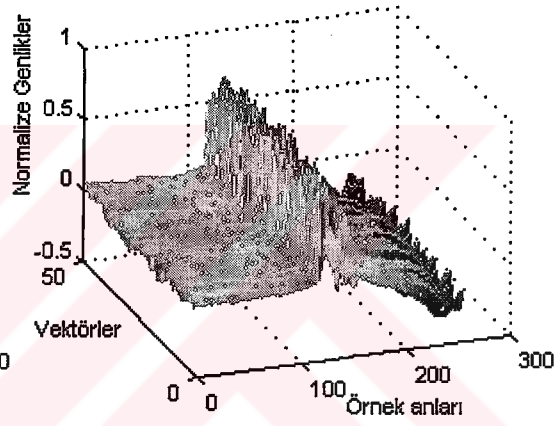
Şekil 5.3.g) E tipi EKG vurusu



h) V tipi EKG vurusu



Şekil 5.3.i) F tipi EKG vurusu



j) f tipi EKG vurusu

değerleri, öznitelik vektörünü oluşturmak üzere araştırılır. Bu bileşenler, diverjansı 2.5225 değerinde maksimum yapacak şekilde aşağıdaki sırada bulunmuştur. Diverjans analizi ilk 40 bileşen üzerine uygulanmıştır. 1'den 40'a kadar olan genlik bileşenleri içinden en yüksek diverjans değerini veren ilk 15 bileşen tespit edilir ve bütün öznitelik vektörlerinde bu bileşenler kullanılır.

$K_i^T = [k_{i1}, k_{i2}, k_{i3}, ..., k_{i15}]$

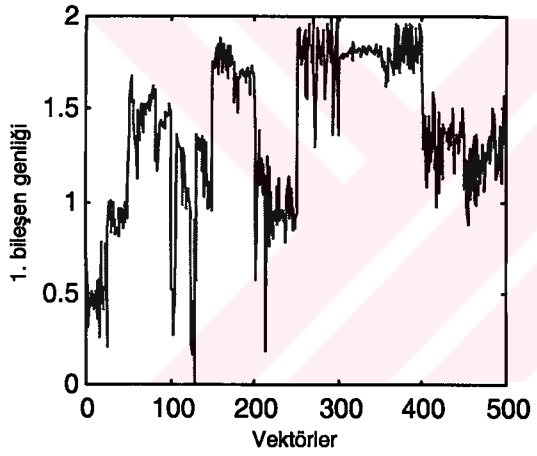
X(j) : j. frekans spektrumu bileşeni

K_i : yeni oluşturulan *i.* öznitelik vektörü

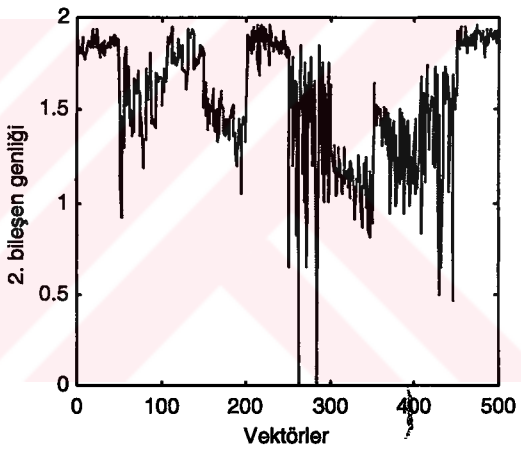
k_{i1}	k_{i2}	k_{i3}	k_{i4}	k_{i5}	k_{i6}	k_{i7}	k_{i8}	k_{i9}	k_{i10}	k_{i11}	k_{i12}	k_{i13}	k_{i14}	k_{i15}
X(3)	X(11)	X(6)	X(7)	X(2)	X(35)	X(10)	X(19)	X(4)	X(8)	X(9)	X(14)	X(5)	X(22)	X(39)

Şekil 5.4'te öznitelik vektörünün tek bir bileşeninin 10 sınıftaki genlik dağılımı gösterilmiştir. Şekilde x ekseni eğitim kümesindeki 500 (herbir sınıf için 50 vektör

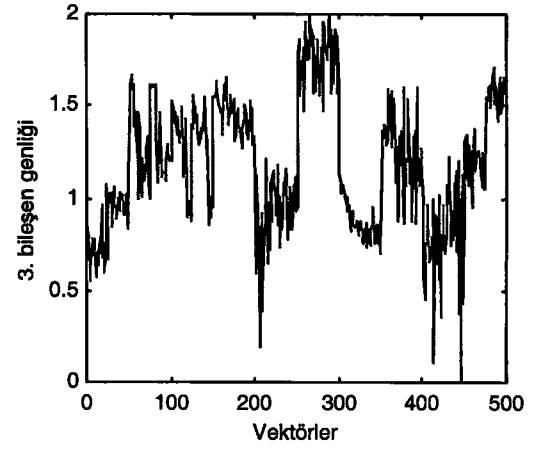
olmak üzere 10x50 vektör) adet vektörü göstermektedir. Sınıflar x ekseni boyunca 50şerlik gruplar halinde sırayla dizilidirler. y ekseni ise bu vektörlerin incelenen bileşeninin 10 sınıftaki genlik dağılımını vermektedir. Öznitelik vektörlerinin tüm bileşenleri 0-2 aralığına normalize edilmiştir. Öznitelik vektörünün bileşenlerinin normalize edilmesi vektörlerin dağılımını bozmadığı için diverjans değerini değiştirmez. Bu işlem, genetik havuzdaki temsili vektörlerin bileşenlerinde nicemleme hatasını azaltmak için gerçeklenmiştir. Şekiller incelendiğinde sınıf içi saçılımın büyük olduğu gözlenmektedir. Sınıf dağılımının en iyi durumda olması gereken halini ifade etmek ve bulunan diverjans değerlerini anlamlandırmak için 15. bileşen, şekil 5.4.p'deki grafik ile değiştirilmiş ve diverjans değeri 68.5245 olarak bulunmuştur. Son grafikte basamaklar, sınıflar arası ayrılığı; basamaklar üzerindeki küçük (gürültü şeklindeki) salınımlar, sınıf içi saçılımı göstermektedir.



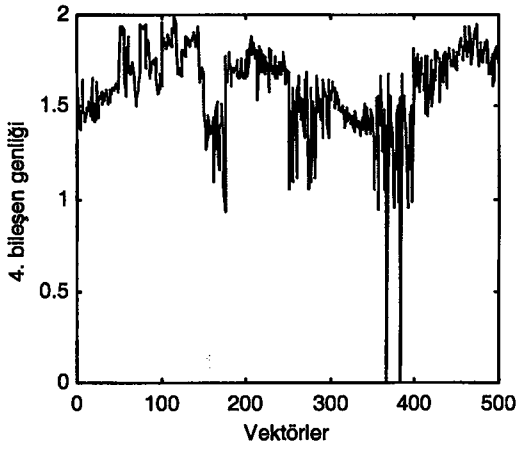
Şekil 5.4.a) 1. bileşenin genlik dağılımı



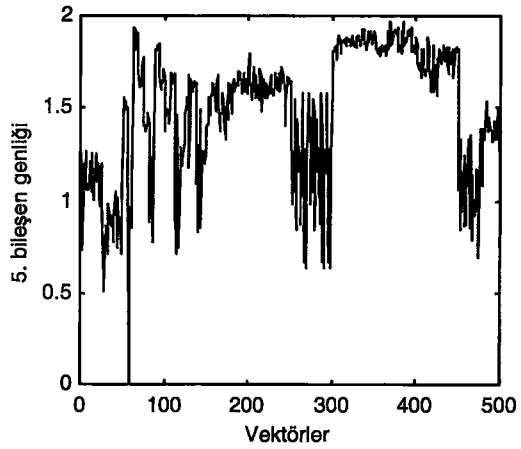
b) 2. bileşenin genlik dağılımı



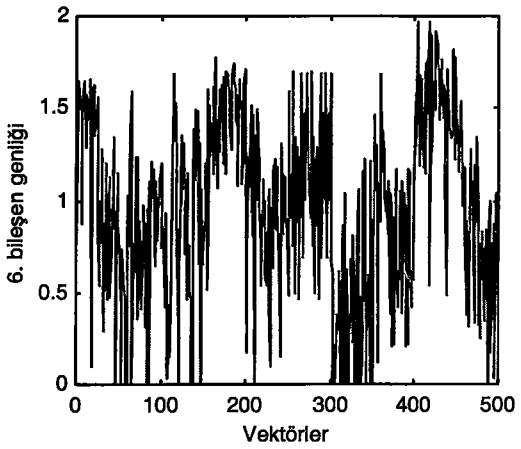
Şekil 5.4.c) 3. bileşenin genlik dağılımı



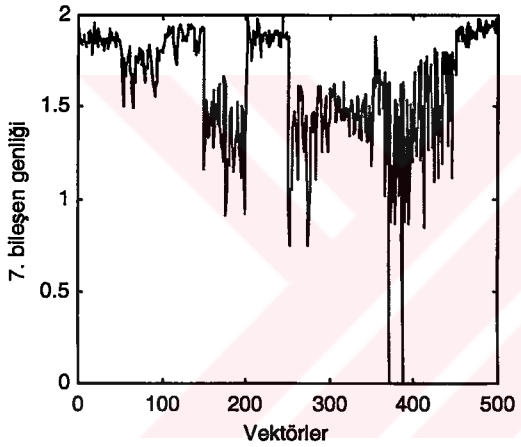
d) 4. bileşenin genlik dağılımı



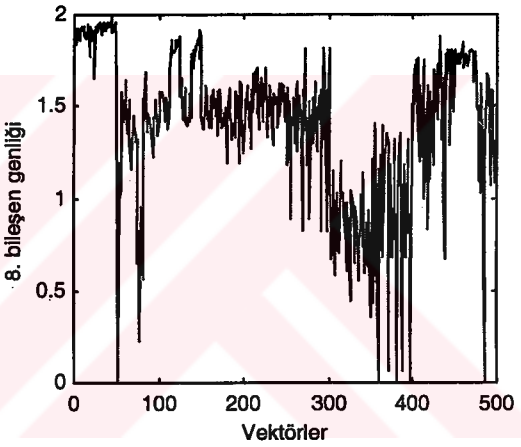
Şekil 5.4.e) 5. bileşenin genlik dağılımı



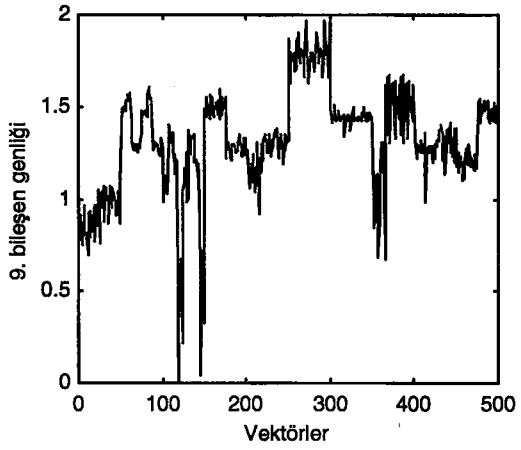
f) 6. bileşenin genlik dağılımı



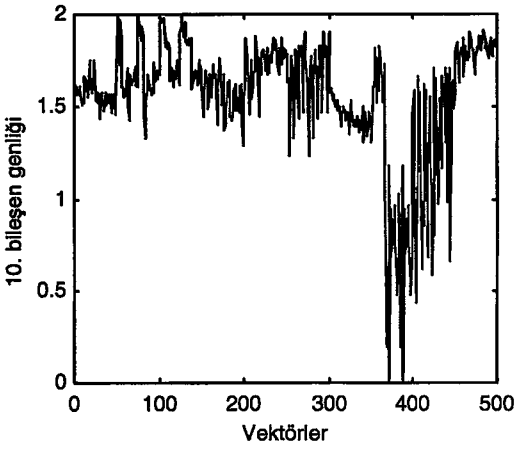
Şekil 5.4.g) 7. bileşenin genlik dağılımı



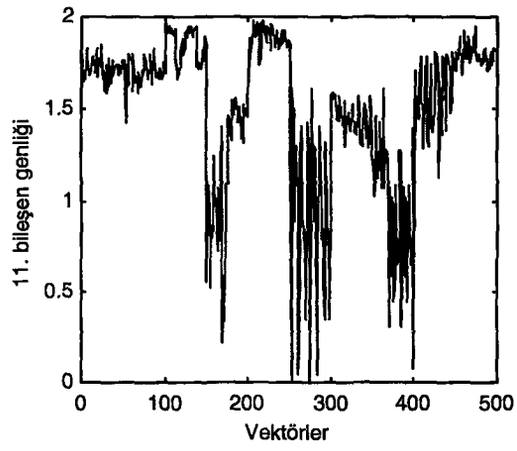
h) 8. bileşenin genlik dağılımı



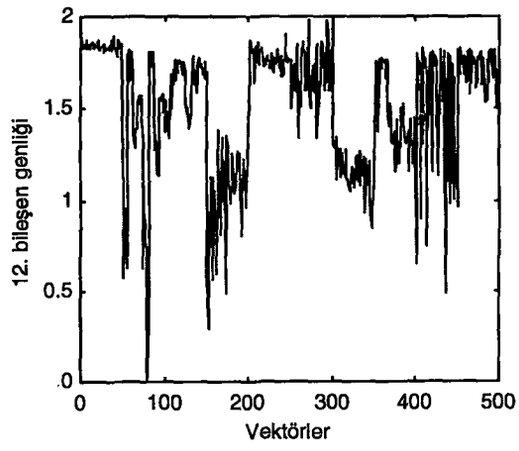
Şekil 5.4.i) 9. bileşenin genlik dağılımı



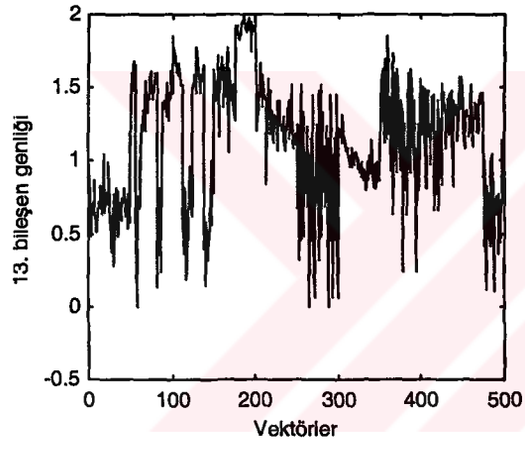
j) 10. bileşenin genlik dağılımı



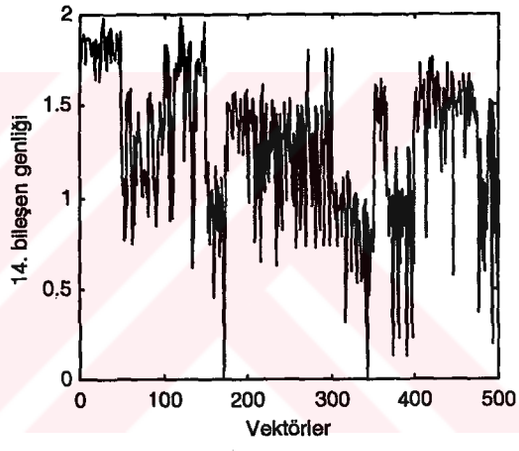
Şekil 5.4.k) 11. bileşenin genlik dağılımı



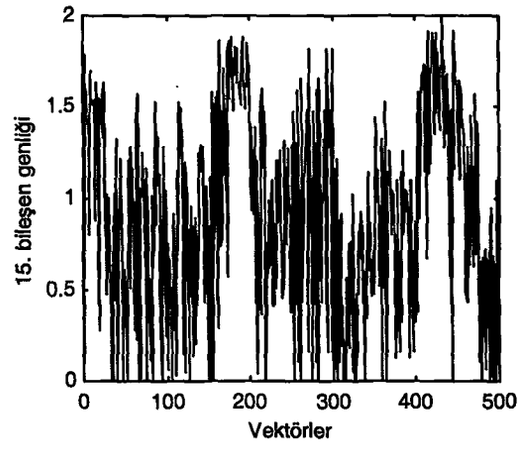
l) 12. bileşenin genlik dağılımı



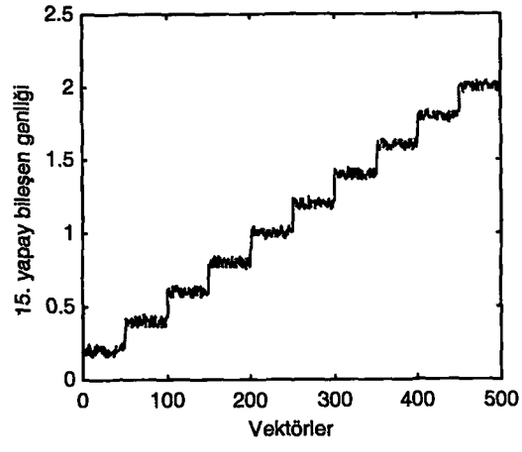
Şekil 5.4.m) 13. bileşenin genlik dağılımı



n) 14. bileşenin genlik dağılımı



Şekil 5.4.o) 15. bileşenin genlik dağılımı



p) 15. yapay bileşenin genlik dağılımı

5.1.2.2 Dalgacık Katsayıları ve Otokorelasyonları

EKG işaretinden QRS kompleksi belirlenir ve bu bölgeden R tepesi bulunur. R tepesi etrafında sola doğru 128, sağa doğru 128 örnek alınarak bir pencere oluşturulur.

Frekans spektrumunda olduğu gibi dalgacık düzlemi üzerinde 10 farkı EKG işareti için anlamlı katsayılar araştırılır ve bu katsayılar kullanılarak öznitelik vektörleri oluşturulur. Şekil 5.5'te 10 farklı EKG vurusu için 'Daubechies-2' ye göre dalgacık düzlemleri gösterilmektedir. Öznitelik vektörlerine yeni bileşenler katmak için 4. ayrıştırma seviyesindeki 18 dalgacık yaklaşıklık katsayısı üzerinde otokorelasyon işlemi gerçeklenerek r(0)–r(10) otokorelasyon değerleri hesaplanır. 2., 3. ve 4. seviyelerdeki dalgacık ayrıntı katsayıları, 4. seviye dalgacık yaklaşıklık katsayıları ve 11 adet otokorelasyon değeri birleştirilerek 66+34+18+18+11=147 boyutlu 500 adet vektör oluşturulur. Dinamik programlama kullanılarak vektörün bileşenleri diverjansı maksimum yapacak şekilde sıralanır. Bu sıra, diverjansı 7.2076 değerinde maksimum yapacak şekilde belirlenmiştir. 147 vektör bileşeni içinden en iyi 15 bileşen seçilmiş ve bütün öznitelik vektörlerinde bu bileşenler kullanılmıştır.

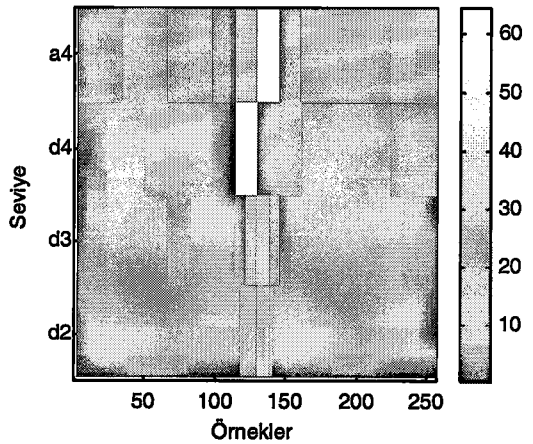
$K_i^T = [k_{i1}, k_{i2}, k_{i3}, ..., k_{i15}]$

K_i : Yeni oluşturulan *i.* öznitelik vektörü

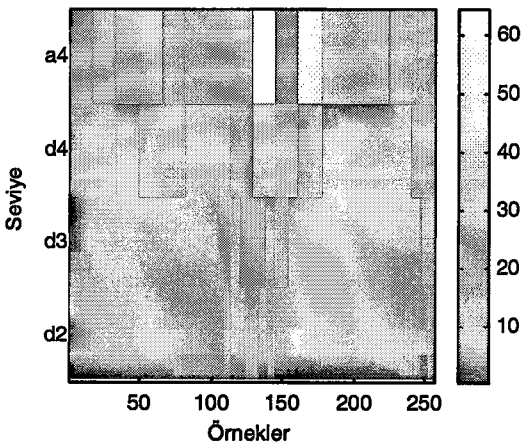
r(n) : n. otokorelasyon bileşeni, *n=0,1,...,10*

$a_4(j)$: 4. seviyedeki j. dalgacık yaklaşıklık katsayısı, *j=1,2,...,18*

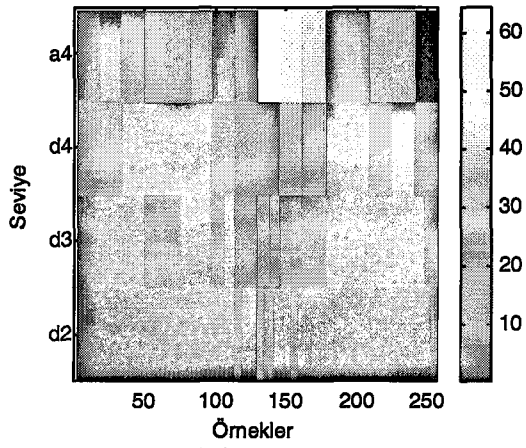
k_{i1}	k_{i2}	k_{i3}	k_{i4}	k_{i5}	k_{i6}	k_{i7}	k_{i8}	k_{i9}	k_{i10}	k_{i11}	k_{i12}	k_{i13}	k_{i14}	k_{i15}
r(1)	r(2)	$a_4(17)$	$a_4(9)$	$a_4(18)$	$a_4(7)$	$a_4(14)$	r(8)	$a_4(6)$	$a_4(10)$	$a_4(13)$	r(7)	$a_4(12)$	r(5)	r(3)



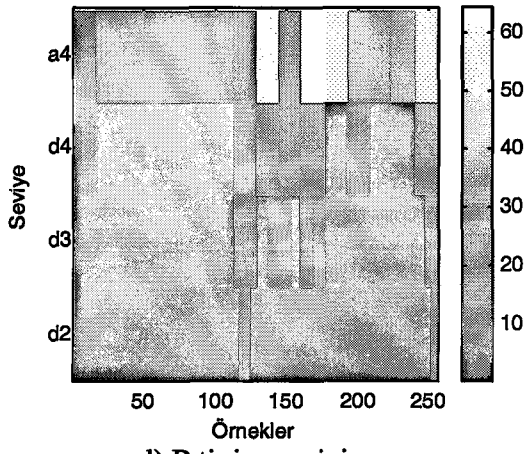
Şekil 5.5.a) N tipi vuru için dalgacık düzlemi



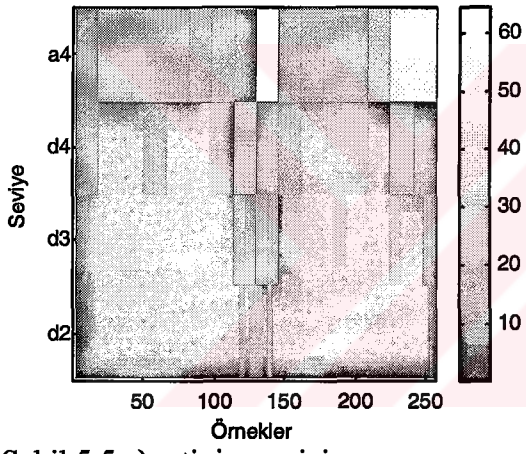
b) L tipi vuru için dalgacık düzlemi



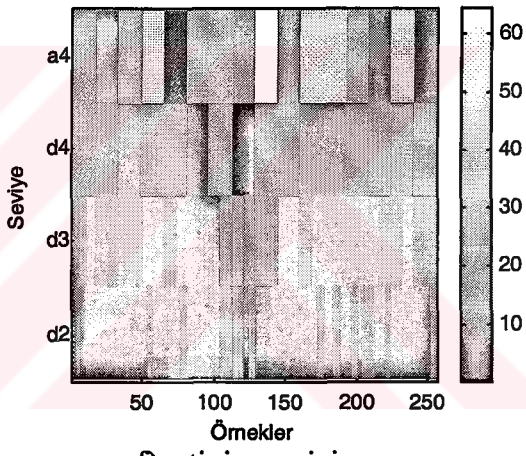
Şekil 5.5.c) R tipi vuru için dalgacık düzlemi



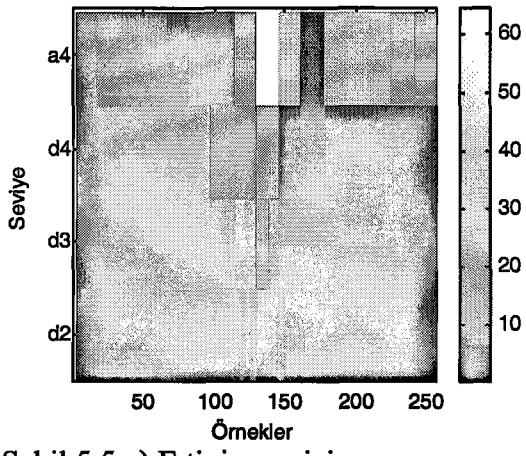
d) P tipi vuru için dalgacık düzlemi



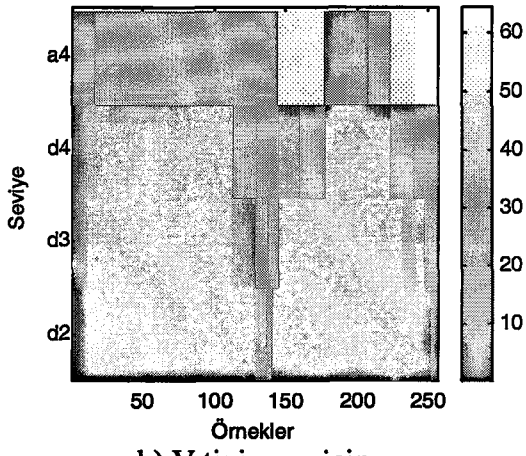
Şekil 5.5.e) p tipi vuru için dalgacık düzlemi



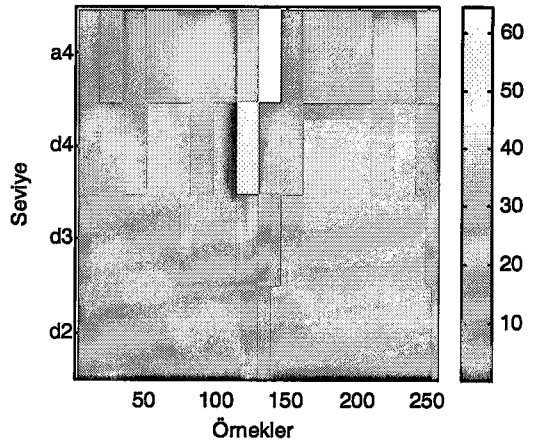
f) a tipi vuru için dalgacık düzlemi



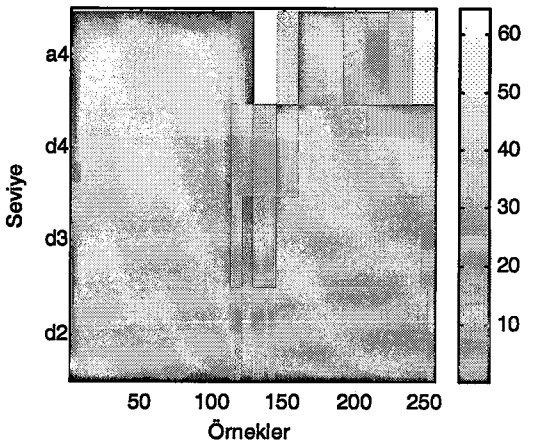
Şekil 5.5.g) E tipi vuru için dalgacık düzlemi



h) V tipi vuru için dalgacık düzlemi



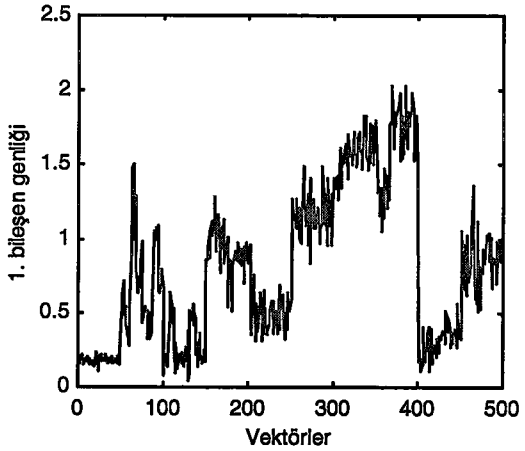
Şekil 5.5.i) F tipi vuru için dalgacık düzlemi



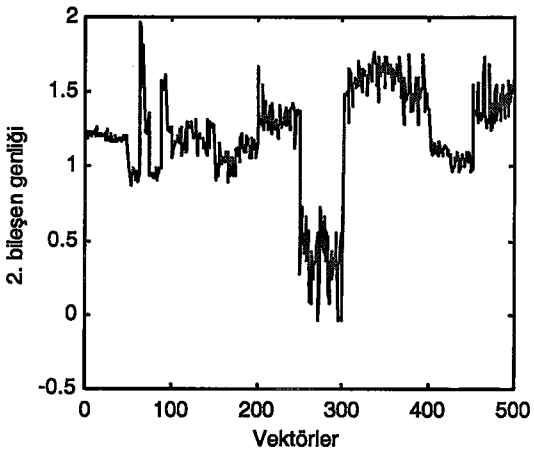
j) f tipi vuru için dalgacık düzlemi

Şekil 5.6'da öznitelik vektörünün tek bir bileşeninin 10 sınıftaki genlik dağılımı gösterilmiştir. Şekilde x ekseni eğitim kümesindeki 500 adet vektörü göstermektedir. Sınıflar x ekseni boyunca 50şerlik gruplar halinde sırayla dizilidirler. y ekseni ise bu vektörlerin incelenen bileşeninin 10 sınıftaki genlik dağılımını vermektedir.

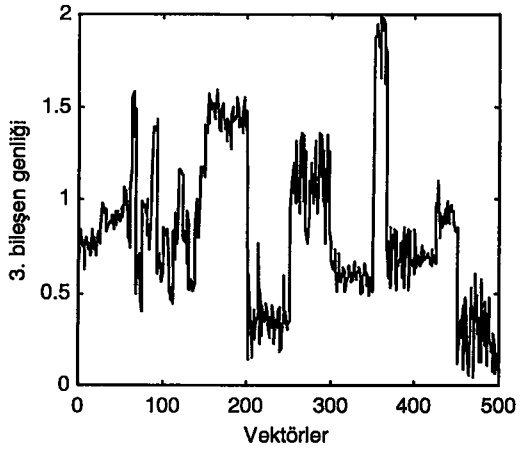
Dalgacık temelli öznitelik çıkartma yönteminde R tepesinin yerinin belirlenmesi son derece önemlidir. R tepesinin yanlış belirlenmesi durumunda dalgacık düzleminde elde edilen şeklin ötelendiği gözlemlenir. Öteleme sonucu öznitelik vektörleri tamamen farklılaşacak ve yanlış sınıflama kararı oluşumuna sebep olacaktır. Bu durumun etkisi güçlü bir R tepesi belirleme algoritması ile giderilir [67]. Çalışma içersinde otokorelasyon değerlerinin ve dalgacık yaklaşık katsayılarının (zamanda çözünürlüğü düşük) kullanılması öteleme etkisini bir ölçüde kaldırmaktadır.



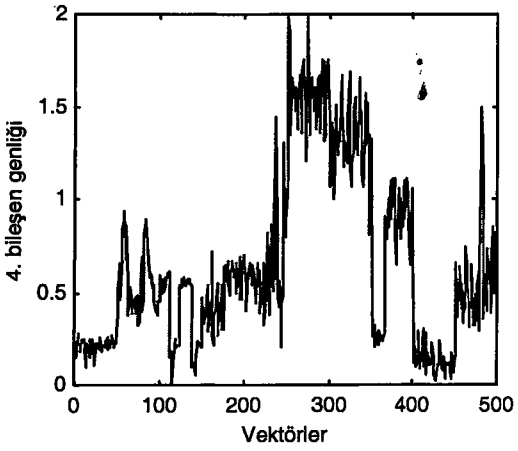
Şekil 5.6.a) 1. bileşenin genlik dağılımı



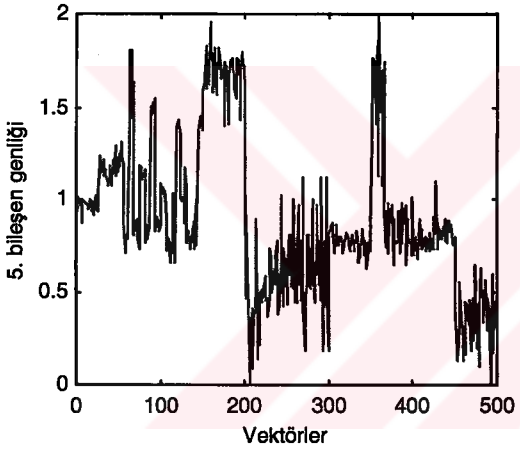
b) 2. bileşenin genlik dağılımı



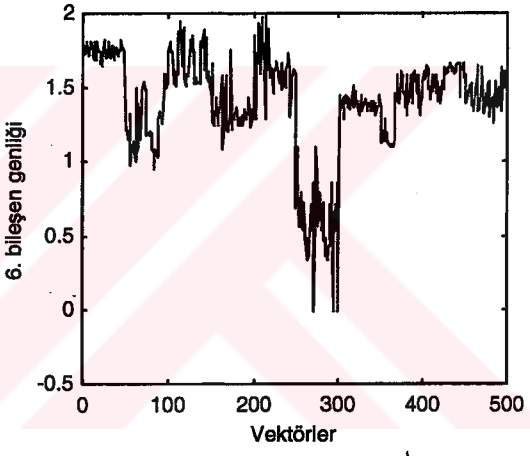
Şekil 5.6.c) 3. bileşenin genlik dağılımı



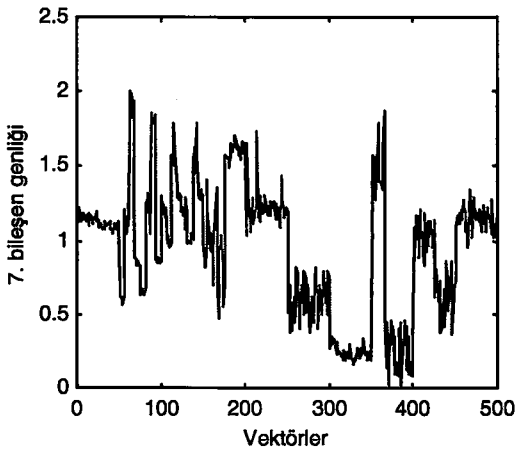
d) 4. bileşenin genlik dağılımı



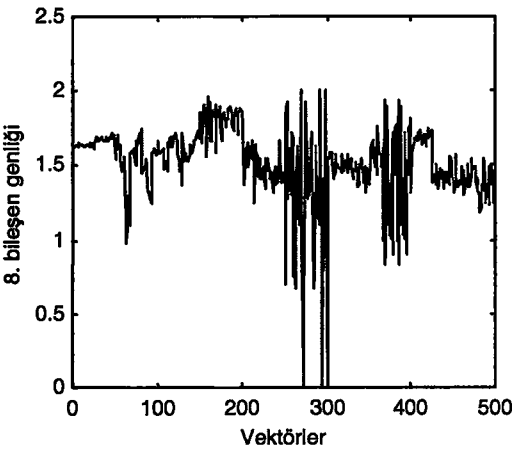
Şekil 5.6.e) 5. bileşenin genlik dağılımı



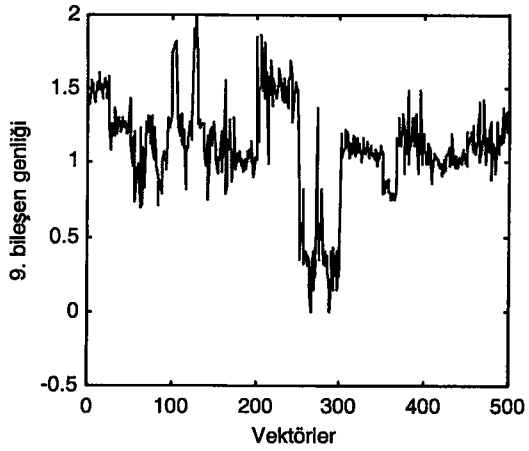
f) 6. bileşenin genlik dağılımı



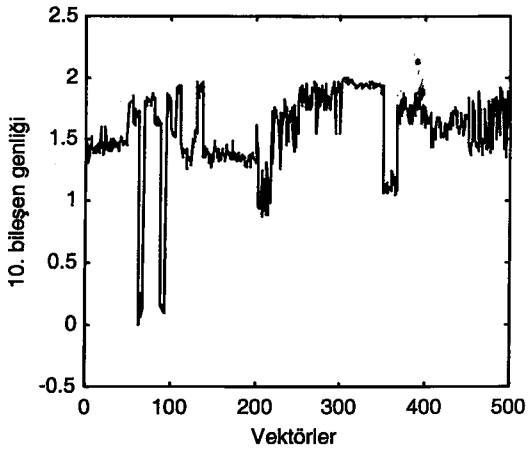
Şekil 5.6.g) 7. bileşenin genlik dağılımı



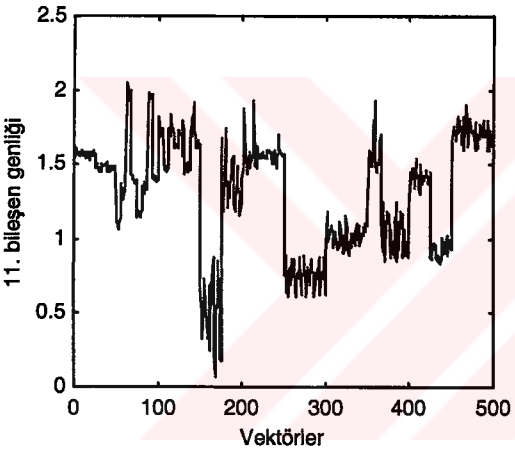
h) 8. bileşenin genlik dağılımı



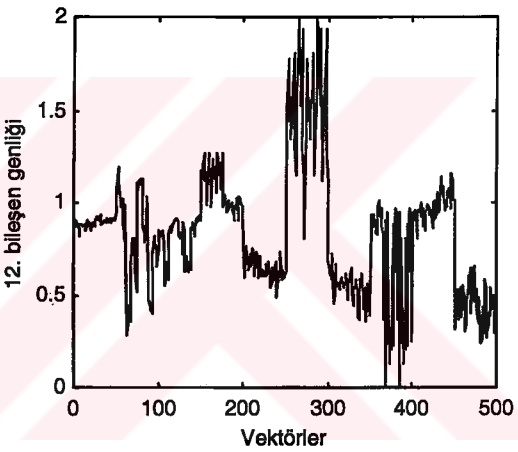
Şekil 5.6.i) 9. bileşenin genlik dağılımı



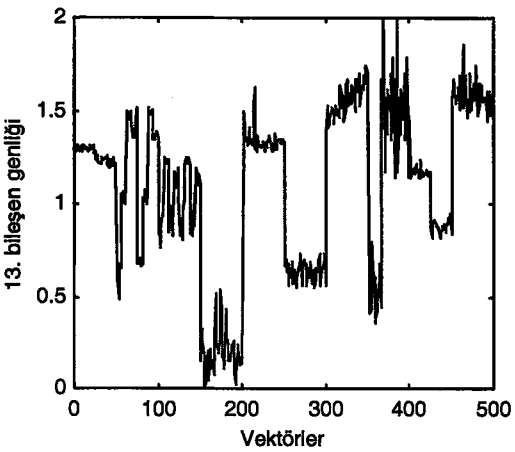
j) 10. bileşenin genlik dağılımı



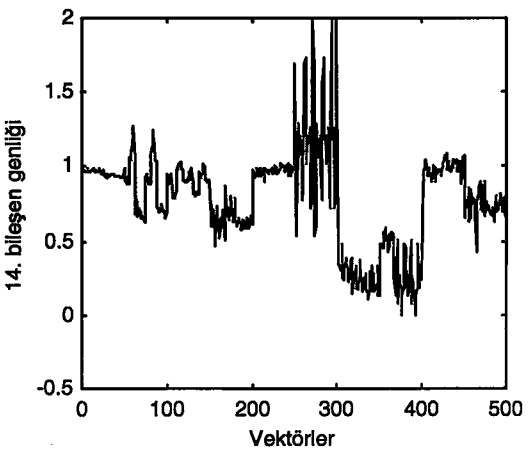
Şekil 5.6.k) 11. bileşenin genlik dağılımı



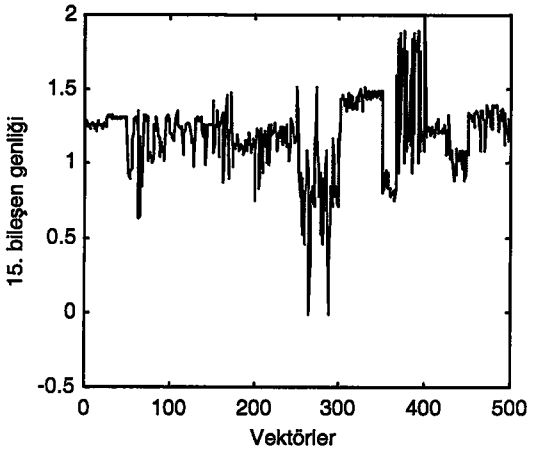
l) 12. bileşenin genlik dağılımı



Şekil 5.6.m) 13. bileşenin genlik dağılımı



n) 14. bileşenin genlik dağılımı

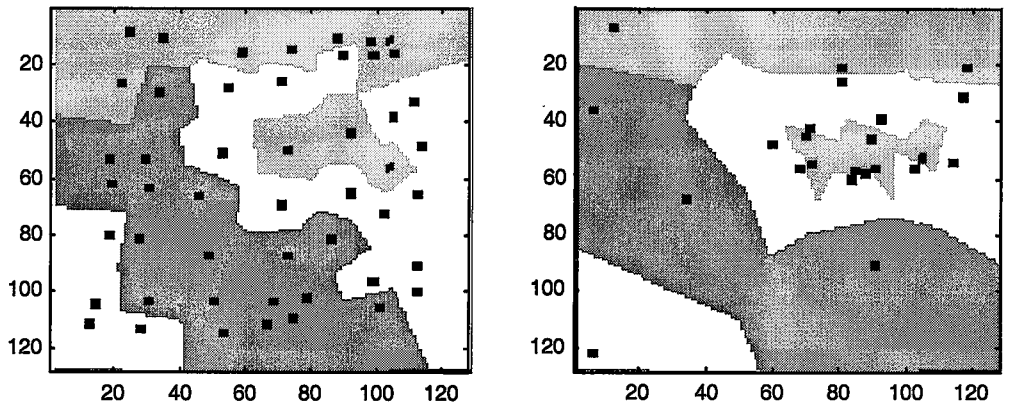


Şekil 5.6.o) 15. bileşenin genlik dağılımı

5.2 GetKoh Ağının Sınıflama Sonuçları

5.2.1 Örnek Öznitelik Uzayı İçin Sınıflama Sonuçları

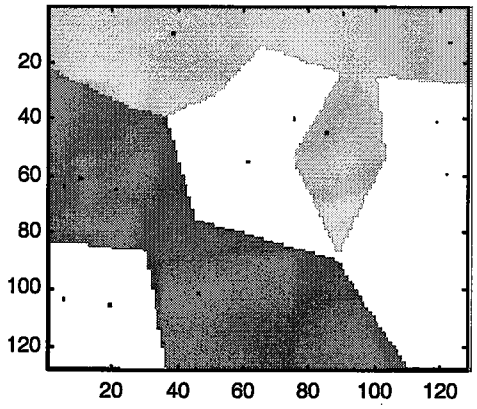
Bölüm 3'te Kohonen ve GAL ağının problemleri ayrıntılı olarak incelenmiştir. Kohonen ağının başlıca problemleri; eğitim öncesinde komşuluk sabiti, kazanç sabiti ve çıkış düğüm sayısının belirlenmesi, eğitim algoritmasının çıkış düğümlerini sınıfların içlerine doğru hareket ettirmesi ve eğitim algoritması içinde düğüm sayısını ihtiyaca göre belirleyecek bir mekanizmanın olmayışı şeklinde özetlenebilir. Şekil 5.7.a'da Kohonen ağının sınıf içlerini de temsil etmek için düğüm kullandığı gözlenmektedir. Sınıfların birbirlerine yakın olduğu bölgelerde sınırların düzgün belirlenemediği gözlenmektedir.



Şekil 5.7.a Kohonen ağının sınıflama sonucu b. GAL ağının sınıflama sonucu

GAL ağının problemleri; eğitim sonrası bulunan düğümlerin ağırlıklarının eğitim kümesindeki vektörlerin ağırlıklarından farklı olmayışı (ezberci bir eğitim yöntemi), öznitelik uzayında sınıfların biribirine yaklaşması durumunda eğitim sonrası aşırı sayıda düğüm elde edilmesi şeklinde özetlenebilir. Şekil 5.7.b'de GAL ağının sınıf içlerini temsil etmek için düğüm ataması yapmadığı ve sınıf sınırlarını daha iyi temsil etmek için düğümlerin sınırlarda yoğunlaştığı gözlenmektedir. Ancak ezberci bir eğitim algoritması kullanıldığı için ağın düğümleri, eğitim kümesindeki vektörlerden oluşur; düğüm ağırlıkları eğitim sırasında değiştirilmez. Bu nedenle sınıflama sonucu eğitim kümesine çok bağlıdır ve GAL'ın öğrenme algoritması, sınıflama başarımını artırmak için yeni düğüm pozisyonları oluşturamaz. Diğer bir problem de düğüm sayısının, sınıf sınırları birbirlerine yaklaştığında aşırı büyümesidir.

Şekil 5.7.c'de genetik algoritmalar ile eğitilen ağın sınıflama sonucu görülmektedir. Yeni geliştirilen ağ ile düğümler, sınıf sınırlarına veya sınıfların içlerine doğru bir hareket oluşturmazlar; genetik algoritmalar ile en iyi sınıflama sonucu üretecek pozisyona doğru yönlenirler. Böylece sınıf sınırları birbirine yaklaştığı durumda bile düğüm sayısında aşırı bir artış gözlenmez. Eğitim algoritması sayesinde düğüm sayısı ihtiyaca göre kendiliğinden belirlenir. Tablo 5.1, genetik algoritmalar ile eğitilen ağın, daha az düğüm ile daha iyi sınıflama başarımı verdiğini göstermektedir.



Şekil 5.7.c GetKoh ağının sınıflama sonucu

Tablodan, GAL ağının eğitiminin daha hızlı olduğu gözlenmektedir; ancak aradaki fark aşırı derecede büyük değildir.

Tablo 5.1 Örnek öznitelik uzayında GAL, Kohonen ve GetKoh'un başarım sonuçları

Testler	GAL	Kohonen	GetKoh
Düğüm sayısı	24	49	14
Sınıflama başarımı	0.91	0.90	0.93
Eğitim zamanı	1.5 sn	1 sn	14 sn

5.2.2 EKG Vurularının Belirlenmesi

Tablo 5.2'de frekans temelli öznitelik vektörleri (FTÖV) kullanılarak 10 farklı EKG vurusu için başarım sonuçları gösterilmiştir. Tablo 5.3'te ise dalgacık temelli öznitelik vektörleri (DTÖV) kullanılarak elde edilen başarım sonuçları gösterilmiştir.

Tablo 5.2 FTÖV kullanılarak GAL, Kohonen ve GetKoh'un başarım sonuçları

Testler	GAL ile başarım	Kohonen ile başarım	GetKoh ile başarım
N tipi vuru	50/50	45/50	47/50
L tipi vuru	46/50	29/50	38/50
R tipi vuru	44/50	39/50	46/50
P tipi vuru	45/50	44/50	50/50
p tipi vuru	46/50	30/50	50/50
a tipi vuru	50/50	46/50	50/50
E tipi vuru	48/50	49/50	49/50
V tipi vuru	44/50	42/50	43/50
F tipi vuru	49/50	39/50	44/50
f tipi vuru	48/50	40/50	44/50
Genel başarım	0,94	0,806	0,922
Düğüm sayısı	41	49	38
Eğitim zamanı	2 sn	1 sn	490 sn

Tablo 5.3 DTÖV kullanılarak GAL, Kohonen ve GetKoh'un başarım sonuçları

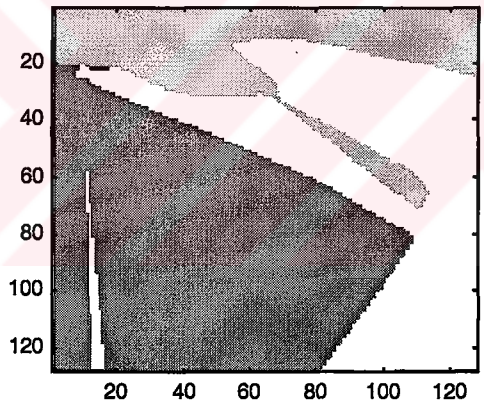
Testler	GAL ile başarım	Kohonen ile başarım	GetKoh ile başarım
N tipi vuru	50/50	41/50	50/50
L tipi vuru	46/50	48/50	50/50
R tipi vuru	50/50	44/50	50/50
P tipi vuru	50/50	50/50	50/50
p tipi vuru	42/50	48/50	50/50
a tipi vuru	50/50	50/50	50/50
E tipi vuru	49/50	48/50	49/50
V tipi vuru	40/50	43/50	48/50
F tipi vuru	49/50	49/50	50/50
f tipi vuru	50/50	48/50	50/50
Genel başarım	0,952	0,938	0,994
Düğüm sayısı	22	49	28
Eğitim zamanı	2 sn	1 sn	254 sn

Giriş uzayı boyutunun büyümesi durumunda, ağların eğitim sürelerinde de bir artış olduğu gözlenmiştir.

5.3 GetÇKA'nın Sınıflama Sonuçları

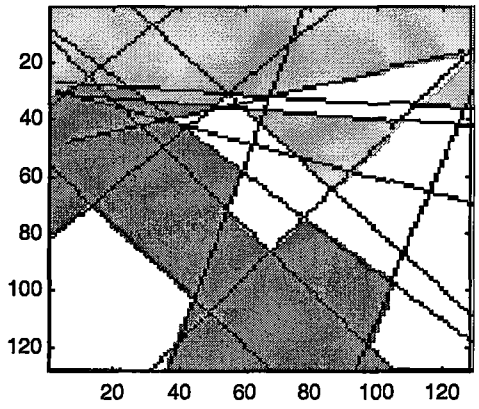
5.3.1 Örnek Öznitelik Uzayı İçin Sınıflama Sonuçları

Bölüm 3'te çok katmanlı ağın problemleri ayrıntılı olarak incelenmiştir. Çok katmanlı ağın problemleri; saklı katman sayısının ve bu katmandaki düğümlerin sayısının eğitim öncesinde belirlenmesi, ağın eğitiminde kullanılan algoritmaların yerel optimumlara yakalanması ve optimum çözüme ulaşmak için uzun süre eğitilmesi şeklinde özetlenebilir. Şekil 5.8.a'da çok katmanlı ağın örnek öznitelik uzayını sınıflaması gösterilmektedir.



Şekil 5.8.a Çok katmanlı ağın sınıflama sonucu

Şekil 5.8.b'de genetik algoritmalar ile eğitilen çok katmanlı ağın sınıflama sonucu ve birinci katmandaki düğümlerin iki boyutlu uzayda temsil ettiği doğrular görülmektedir. Genetik algoritmalar, en iyi sınıflama sonucu üretecek düğümlerin temsil ettiği doğruları araştırır. Böylece sınıf sınırları birbirine yaklaşsa bile düğüm sayısında aşırı bir artış gözlenmez. Eğitim sırasında düğüm sayısı ihtiyaca göre kendiliğinden belirlenir. Tablo 5.4, genetik algoritmalar ile eğitilen çok katmanlı ağın daha az düğüm ile daha iyi sınıflama başarımı verdiğini göstermektedir. Tablodan, genetik algoritmalar ile ağın eğitiminin daha hızlı olduğu gözlenmektedir.



Şekil 5.8.b GetÇKA'nın sınıflama sonucu

Tablo 5.4 Örnek öznitelik uzayında ÇKA ve GetÇKA'nın başarım sonuçları

Testler	ÇKA	GetÇKA
Düğüm sayısı	2-40-48-3	13
Sınıflama başarımı	0.76	0.93
Eğitim zamanı	7×700 sn	44 sn

5.3.2 EKG Vurularının Belirlenmesi

Tablo 5.5'te frekans spektrumu kullanılarak elde edilen başarım sonuçları gösterilmiştir. Tablo 5.6'da ise dalgacık temelli öznitelik vektörleri kullanılarak elde edilen başarım sonuçları gösterilmiştir.

Tablo 5.5 FTÖV kullanılarak ÇKA ve GetÇKA'nın başarım sonuçları

Testler	ÇKA ile başarım	GetÇKA ile başarım
N tipi vuru	50/50	46/50
L tipi vuru	50/50	37/50
R tipi vuru	50/50	46/50
P tipi vuru	43/50	42/50
p tipi vuru	50/50	50/50
a tipi vuru	50/50	50/50
E tipi vuru	50/50	50/50
V tipi vuru	10/50	32/50
F tipi vuru	50/50	50/50
f tipi vuru	48/50	50/50
Genel başarım	0,902	0,906
Düğüm sayısı	15-30-20-10	18
Eğitim zamanı	7×960 sn	310 sn

Tablo 5.6 DTÖV kullanılarak ÇKA ve GetÇKA'nın başarım sonuçları

Testler	ÇKA ile başarım	GetÇKA ile başarım
N tipi vuru	50/50	50/50
L tipi vuru	50/50	47/50
R tipi vuru	37/50	49/50
P tipi vuru	50/50	50/50
p tipi vuru	50/50	50/50
a tipi vuru	50/50	50/50
E tipi vuru	50/50	49/50
V tipi vuru	33/50	46/50
F tipi vuru	48/50	40/50
f tipi vuru	50/50	50/50
Genel başarım	0,936	0,962
Düğüm sayısı	15-20-10-10	22
Eğitim zamanı	8×840 sn	396 sn

Giriş uzayının boyutu büyüdüğünde ağların eğitim sürelerinde de bir artış olduğu gözlenmektedir.

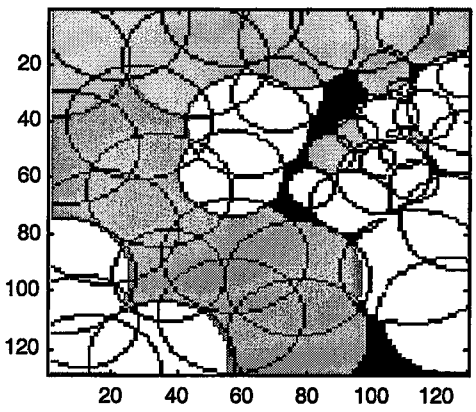
5.4 GetPrz ve GetRCE Ağlarının Sınıflama Sonuçları

5.4.1 Örnek Öznitelik Uzayı İçin Sınıflama Sonuçları

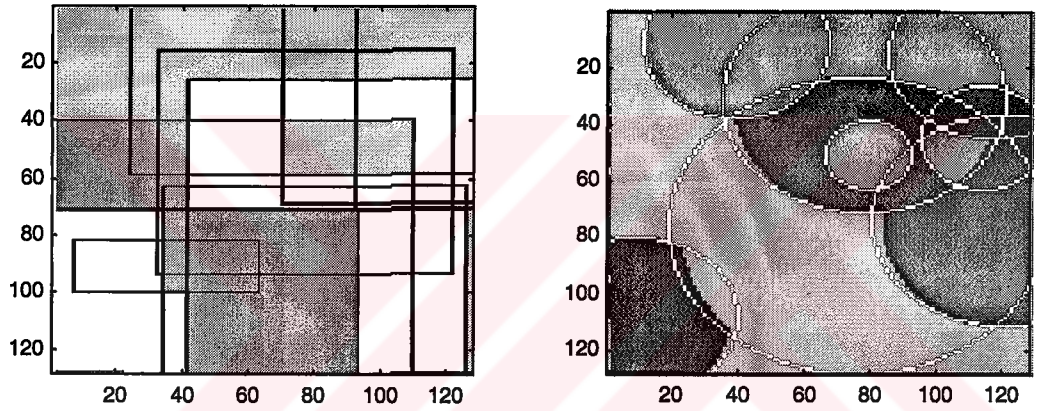
GetPrz ağı öznitelik uzayında kapalı bölgeler oluşturmaktadır. RCE ağı, GetPrz ağı gibi öznitelik uzayında kapalı bölgeler oluşturur; ancak düğümler öznitelik uzayında hiper küreleri temsil etmektedir. Her üç ağda da bir düğüm tek başına kapalı bir bölge oluşturabildiği için, aşağıda üç ağın sınıflama sonuçları karşılaştırmalı olarak incelenecektir.

RCE ağının problemleri; eğitim algoritmasında hiper küre merkezlerini optimum belirleyecek bir mekanizma olmaması, ağın düğümlerinin temsil ettiği hiper kürelerin birbirleri ile çakışma oluşturmaması (az sayıda düğüm ile çok sayıda kapalı bölge elde edilemeyişi) şeklinde özetlenebilir. Şekil 5.9.a'da RCE ağının öznitelik uzayını sınıflama sonuçları ile birinci katmandaki düğümlerin iki boyutlu uzayda temsil ettiği daireler (n boyutta hiper küreler) gösterilmektedir.

Şekil 5.9.b,c'de GetPrz ve GetRCE ağlarının sınıflama sonuçları ve birinci katmandaki düğümlerin temsil ettiği kapalı bölgeler görülmektedir. Eğitim sırasında düğüm sayısı ihtiyaca göre kendiliğinden belirlenir.



Şekil 5.9.a RCE ağının sınıflama sonucu



(b) (c)

Şekil 5.9.b) GetPrz ve c) GetRCE ağlarının sınıflama sonucu

Tablo 5.7, genetik algoritmalar ile eğitilen ağın, daha az düğüm ile daha iyi sınıflama başarımı verdiğini göstermektedir. Tablodan, genetik algoritmalar ile ağların eğitiminin daha uzun sürdüğü gözlenmektedir; ancak aradaki fark aşırı derecede büyük değildir.

Tablo 5.7 Örnek öznitelik uzayında RCE, GetRCE ve GetPrz'nin başarım sonuçları

Testler	RCE	GetRCE	GetPrz
Düğüm sayısı	41	8	9
Sınıflama başarımı	0.92	0.93	0.95
Eğitim zamanı	3 sn	40 sn	18 sn

5.4.2 EKG Vurularının Belirlenmesi

Tablo 5.8'de frekans spektrumu kullanılarak elde edilen başarım sonuçları gösterilmiştir. Tablo 5.9'da ise dalgacık temelli öznitelik vektörleri kullanılarak elde edilen başarım sonuçları gösterilmiştir.

Tablo 5.8 FTÖV kullanılarak RCE, GetRCE ve GetPrz'nin başarım sonuçları

Testler	RCE ile başarım	GetRCE ile başarım	GetPrz ile başarım
N tipi vuru	43/50	44/50	48/50
L tipi vuru	45/50	39/50	46/50
R tipi vuru	38/50	45/50	37/50
P tipi vuru	41/50	43/50	46/50
p tipi vuru	43/50	50/50	50/50
a tipi vuru	50/50	50/50	50/50
E tipi vuru	40/50	50/50	50/50
V tipi vuru	34/50	38/50	30/50
F tipi vuru	46/50	49/50	50/50
f tipi vuru	46/50	39/50	39/50
Genel başarım	0,852	0,894	0,892
Düğüm sayısı	91	18	12
Eğitim zamanı	8 sn	130 sn	90 sn

Tablo 5.9 DTÖV kullanılarak RCE, GetRCE ve GetPrz'nin başarım sonuçları

Testler	RCE ile başarım	GetRCE ile başarım	GetPrz ile başarım
N tipi vuru	47/50	48/50	50/50
L tipi vuru	38/50	49/50	46/50
R tipi vuru	41/50	49/50	48/50
P tipi vuru	50/50	50/50	50/50
p tipi vuru	19/50	50/50	50/50
a tipi vuru	50/50	50/50	50/50
E tipi vuru	45/50	50/50	50/50
V tipi vuru	50/50	45/50	46/50
F tipi vuru	34/50	49/50	44/50
f tipi vuru	40/50	50/50	50/50
Genel başarım	0,828	0,98	0,968
Düğüm sayısı	40	16	10
Eğitim zamanı	4 sn	80 sn	60 sn

Tablolardan giriş uzayının boyutu büyüdüğünde ağ eğitim sürelerinde de bir artış olduğu gözlenmektedir.

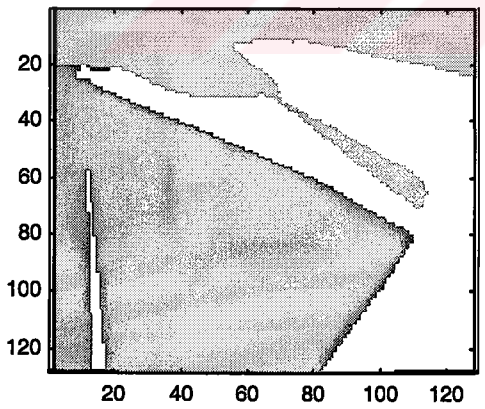
5.5 GetNic Ağının Sınıflama Sonuçları

5.5.1 Örnek Öznitelik Uzayı İçin Sınıflama Sonuçları

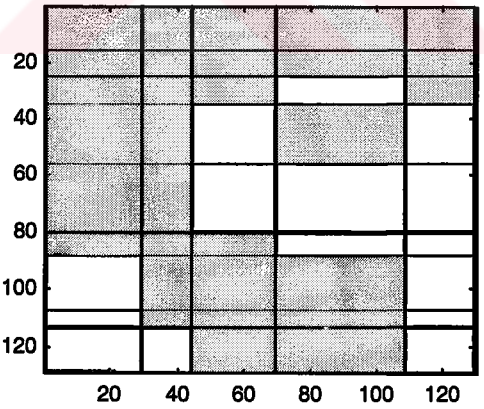
GetNic ağının düğümleri öznitelik uzayında hiper düzlemleri temsil etmektedir. Çok katmanlı ağın birinci katmanındaki düğümler de öznitelik uzayında hiper düzlemleri temsil etmektedir. Her iki ağda da bir düğüm tek başına bir hiper düzlemi temsil ettiği için, aşağıda iki ağın sınıflama sonuçları karşılaştırmalı olarak incelenmiştir.

Bölüm 4.5'te çok katmanlı ağın problemleri ayrıntılı olarak incelenmiştir. Öznitelik uzayında bu ağ ile oluşturulan bölge sayısının, ağın birinci katmanındaki düğüm sayısıyla ilişkisi incelenmiştir. Kullanılan parametre sayısı yönünden incelemeye bakıldığında, GetNic ağının daha az parametre ile daha fazla bölmeleme yapabileceği gözlemlenmiştir. Şekil 5.10.a'da çok katmanlı ağın öznitelik uzayını sınıflaması gösterilmektedir.

Şekil 5.10.b'de genetik algoritmalar ile eğitilen GetNic ağının sınıflama sonucu ve birinci katmandaki düğümlerin iki boyutlu uzayda temsil ettiği doğrular görülmektedir. Eğitim sırasında düğüm sayısı ihtiyaca göre kendiliğinden belirlenir.



Şekil 5.10.a ÇKA'nın sınıflama sonucu



b. GetNic ağının sınıflama sonucu

Tablo 5.10, genetik algoritmalar ile eğitilen ağın, daha az düğüm ile daha iyi sınıflama başarımı verdiğini göstermektedir. Tablodan, genetik algoritmalar ile ağın eğitiminin daha hızlı olduğu gözlenmektedir.

Tablo 5.10 Örnek öznitelik uzayında ÇKA ve GetNic'in başarım sonuçları

Testler	ÇKA	GetNic
Düğüm sayısı	2-40-48-3	12/3=4 düğüm
Sınıflama başarımı	0.76	0.95
Eğitim zamanı	7×700 sn	11 sn

5.5.2 EKG Vurularının Belirlenmesi

Tablo 5.11'de GetNic ağında, frekans spektrumu ve dalgacık temelli öznitelik vektörleri kullanılarak elde edilen başarım sonuçları gösterilmiştir.

Tablodan giriş uzayının boyutu büyüdüğünde ağ eğitim sürelerinde de bir artış olduğu gözlenmektedir.

Tablo 5.11 GetNic ağının FTÖV ve DTÖV ile başarım sonuçları

Testler	Frekans	Dalgacık
N tipi vuru	47/50	48/50
L tipi vuru	49/50	48/50
R tipi vuru	48/50	48/50
P tipi vuru	50/50	49/50
p tipi vuru	50/50	50/50
a tipi vuru	50/50	50/50
E tipi vuru	50/50	50/50
V tipi vuru	35/50	50/50
F tipi vuru	50/50	45/50
f tipi vuru	39/50	50/50
Genel başarım	0,936	0,98
Düğüm sayısı	18/16	16/16
Eğitim zamanı	86 sn	60 sn

6. SONUÇLAR

Çalışma içerisinde yapılan incelemeleri iki grup altında toplayabiliriz: 1) Öznitelik çıkartma işlemi, 2) Sınıflayıcı olarak yeni YSA yapısı geliştirilmesi. Öznitelik vektörlerinin hastanın yaşından, cinsiyetinden ve ölçü cihazının kazanç ve ofset gibi ayarlarından etkilenmemesini sağlamak için giriş işaretinin tepeden tepeye genliği 1 mV'a ve ofseti sıfıra getirilecek şekilde ayarlanır. Analizler, gerek frekans ve gerekse dalgacık analizi ile oluşturulan vektörlerin işaretin genlik ve ofset değişimlerinden etkilendiğini göstermektedir. Ofset ayarı ve tepeden tepeye işaret genliğini sabitleme işlemi ile bu etki yok edilmiştir. Çalışma içersinde her EKG vurusu için 256 veriden oluşan ve sadece bir QRS kompleksini içeren bir dikdörtgensel pencere oluşturulur. Pencere içindeki veriler, QRS içindeki R tepesi pencerenin ortasına gelecek şekilde ayarlanır.

Öznitelik vektörlerinin seçimi sınıflama performansını doğrudan etkilemektedir. Çalışmanın ilk aşamalarında öznitelik vektörlerinin oluşturulması için sadece frekans spektrumu kullanılmaktaydı. Ancak kategori sayısını artırmanın sınıflama performansını düşürdüğü gözlenmiştir. Dinamik programlama ile adım adım boyut artırılarak oluşturulan dağılıma ait diverjans değerleri hesaplandığında sınıfların saçıldığı, bir öbekleşme yapmadığı tespit edilmiştir.

Öznitelik vektörlerinin daha iyi tanımlanması için literatürde sıkça gözlenen dalgacık analizinin kullanılması planlanmıştır. Her 256 veriden oluşan EKG vurusu için bir dalgacık düzlemi tanımlanır. Düzlemde sınıf dağılımını en iyi yapacak katsayıları bulmak için diverjans analizi kullanılmaktadır. 1., 2., 3. ve 4. ayrıştırma seviyelerindeki ayrıntı katsayıları kullanılarak analiz yapılmış, ancak sınıf dağılımının iyileşmediği gözlenmiştir. Özniteliklerin daha iyi tanımlanması için 4. seviyedeki yaklaşıklık katsayıları ve bu katsayıların otokorelasyonları, 2-4. seviyelerdeki ayrıntı katsayıları ile birleştirilmiştir. Dinamik programlama ile 15 adet en iyi vektör bileşeni araştırılmış ve 15 boyutlu uzayda dalgacık yönteminin frekans spekturumuna göre sınıf dağılımında 3 kat bir iyileşme oluşturduğu tespit edilmiştir.

Ancak yeni bileşenlere göre sınıf dağılımına ait diverjans değerleri de öznitelik uzayında sınıfların öbekleşmediğini göstermektedir.

Öznitelik vektörlerinin iyi belirlenememesi sınıflama başarımını düşürmektedir. Analizler, başarımı artırmak için karmaşık sınıf dağılımlarını temsil edecek yeni sınıflayıcı yapılarının araştırılması gerektiğini göstermektedir. Yeni ağ yapıları araştırılırken test ve gözlem amaçlı iki boyutlu örnek bir uzay tanımlanmıştır. Örnek uzayın iki boyutlu olması, sınıf sınırlarının ağ düğümleri tarafından temsil edilişinin görsel incelenmesine imkan vermiştir. Örnek uzay, aynı zamanda klasik ağ eğitim algoritmalarının düğümleri yönlendirişi hakkında bilgi de vermektedir.

Çalışmada sınıfları temsil etmek üzere 5 farklı yapay sinir ağı yapısı incelenmiştir. Bu yapılar, bilginin temsil edilişine göre iki kısma ayrılırlar: 1) Her ağ düğümü, bütüne ait bilginin sadece belirli bir kısmını (yerel bilgi) saklamaktadır, 2) Her ağ düğümü, bütüne ait genel bilgiyi saklamaktadır. Öznitelik uzayında sınıflara ait vektörler saçılmış durumda bulunuyorsa, birinci kısma giren ağlarda eğitim sonrasında düğüm sayısının aşırı arttığı gözlenmektedir. İkinci kısımdaki ağların da düğüm sayısı bu dağılımdan etkilenmekte; ancak bu etki birinciye göre daha az olmaktadır.

Çok katmanlı ağın sınıf bölgelerini temsil edişi örnek uzayda incelenmiş ve bu bölgeleri *daha az parametre* ile oluşturacak yeni yapılar araştırılmıştır. Analizler bölgeleri oluşturacak yeni yapıda hiper düzlemler yerine hiper prizmalar ve kürelerin kullanılmasının avantajlı olacağını göstermiştir. Çok katmanlı ağın birinci saklı katmanındaki düğümleri, öznitelik uzayında hiper düzlemleri temsil etmektedir. Sınıf bölgeleri, bu düzlemlerin kesiştirilmesi sonucu oluşturulmaktadır. Yeni ağ yapılarındaki düğümler de öznitelik uzayında geometrik şekiller oluşturmaktadır. Çalışmada daha az düğüm ile daha fazla ayrık hacimsel bölge oluşturabilmek için hiper prizma ve kürelerin hacimlerini çakıştırma yoluna gidilmiştir. Gerek örnek uzayın gerek EKG işaretlerinin sınıflandırılmasında, yeni yapılar ile daha az parametre kullanılarak daha iyi sınıflama başarımı elde edilmiştir. Hiper küre aynı sayıda ayrık bölgeyi daha az parametre ile oluşturabilmektedir. Hiper prizma ağında, hiper kürelere göre daha fazla parametre kullanılmaktadır. Ancak hiper prizma ağı çoklu bölge kodu ile sınıf dağılımını daha iyi temsil edebilme yeteneğine sahiptir.

GetNic ağı çok katmanlı ağa benzemektedir; ancak çok katmanlı ağın temsil ettiği düzlemler üzerinde bir sınırlama getirilmiştir. Düzlemler bir boyut eksenine dik, diğerlerine paralel olacak şekilde seçilir. Bu koşul, bir düğümün sadece bir parametre ile temsil edilmesine imkan sağlar. Sınıflara ait dağılımın birbirine yaklaştığı uzay bölgeleri daha fazla sayıda düğüm, diğer bölgeler daha az düğüm ile temsil edilmektedir. Bu yaklaşım, incelenen boyutun nicemlenmesi işlemine benzemektedir.

Literatür incelendiğinde, her yapı için farklı bir eğitim algoritması kullanıldığı gözlenmektedir. Çalışmada bu probleme genetik algoritmalar kullanılarak çözüm getirilmiştir. Geliştirilen her yeni yapı için eğitim algoritması fazlaca değişmemiş, böylece yeni yapıların tasarımında ve onların eğitiminde bir zorlukla karşılaşılmamıştır.

Kategori sayısını artırmak sınıflama performansını düşürmektedir [11]. Tablo 6.1'de incelenen tüm ağlar için örnek uzayı ve EKG işaretlerini toplam sınıflama başarımı ve ağların kullandıkları parametre sayıları (PS) verilmektedir.

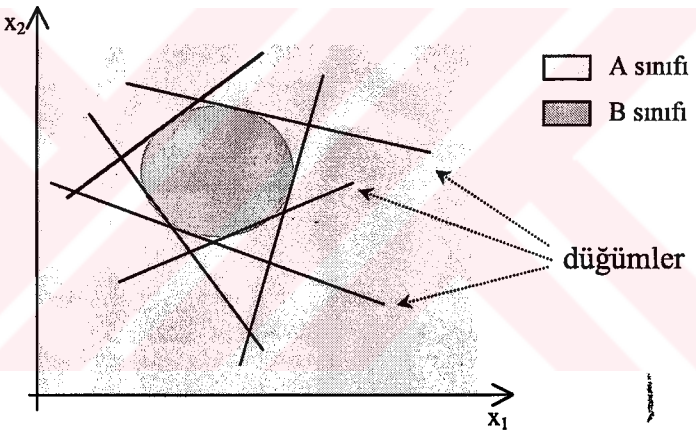
Tablo 6.1 YSA'ların örnek uzayı ve EKG işaretlerini sınıflama başarımları

Yapay Sinir Ağları	Örnek Uzay		Frekans Analizi		Dalgacık Analizi	
	PS	Başarım	PS	Başarım	PS	Başarım
GAL	2×24	0.91	15×41	0.94	15×22	0.95
Kohonen	2×49	0.90	15×49	0.80	15×49	0.93
GetKoh	2×14	0.93	15×38	0.93	15×28	0.99
ÇKA	3×40	0.76	16×30	0.90	16×20	0.93
GetÇKA	3×13	0.93	16×18	0.90	16×22	0.96
RCE	3×41	0.92	16×91	0.85	16×40	0.82
GetPrz	4×9	0.95	30×12	0.89	30×10	0.97
GetRCE	3×8	0.93	16×18	0.89	16×16	0.98
GetNic	1×12	0.95	1×18	0.94	1×16	0.98

n boyutlu uzayda hiper düzlem n+1 parametre, hiper küre n+1 parametre, nokta n parametre, hiper prizma 2×n parametre, nicemleme ağı düğümü ise sadece 1 parametre ile tanımlanmaktadır. 15 boyutlu uzayda hiper düzlem 16 parametre, hiper küre 16 parametre, nokta 15 parametre, hiper prizma 30 parametre, nicemleme ağı düğümü ise 1 parametre ile tanımlanmaktadır. Tez içinde 10 farklı EKG vurusu GetNic ağı kullanılarak %98 başarı ile sınıflanabilmiştir. Gerek örnek uzayın gerekse

EKG vurularının sınıflanmasında GetNic ağının en yüksek başarımları verdiği gözlenmektedir. Genetik algoritmalar ile eğitilen ağların karşılaştırıldıkları ağlara göre düğüm sayısını azaltırken sınıflama performanslarını artırdıkları gözlenmektedir.

Sınıflama başarımlarının yaklaşık olarak %95'ten büyük olması sınıfların öznitelik uzayında dağılımlarına bağlıdır. Bu değerden büyük bir başarım, ağın sınıflama yeteneği ile ilgili değildir. Şekil 6.1'deki gibi iki sınıftan oluşan bir dağılım düşünelim. Daire şeklindeki koyu-gri renkli alan bir sınıfa, açık-gri renkli alan ise diğer bir sınıfa ait olsun. Bu durumda %95'in üzerinde bir başarı elde etmek, düğümleri hiper kürelerden oluşan ağ için basittir; ancak düğümleri hiper düzlemlerden oluşan ağın aynı başarımı verebilmesi için çok sayıda düğüm kullanması gerekir.



Şekil 6.1. İki sınıflı örnek uzay

Literatür incelendiğinde, yapılan çalışmalarda %98 başarım ile en fazla 7 tip EKG vurusunun sınıflandırıldığı gözlenmiştir [35]. Kategori sayısını artırmak sınıflama performansını düşürmektedir [11]. Literatürde, hibrit yapay sinir ağı yapıları kullanılarak başarımın artırılmaya çalışıldığı gözlenmektedir [10,13,14]. Tez çalışmasında, kategori sayısı artırılırken geliştirilen ileri beslemeli hibrit ağ yapıları ile başarımın %97'nin üzerinde tutulması sağlanmıştır. Yeni sınıflayıcı yapıları daha fazla kategorinin yüksek başarımla sınıflanabilmesine olanak sağlamaktadır.

Literatürde EKG işaretlerinin yapay sinir ağları ile sınıflandırılmasına yönelik çalışmalarda, düğüm sayısını indirgeme yöntemlerinin sıkça kullanıldığı

gözlenmektedir [9,27,29]. Tablo 1 incelendiğinde GetNic ağının parametre sayısının diğer ağlara nazaran 16 kat daha az olduğu görülmektedir.

Bazı çalışmalarda sınıflama başarımını artırmak için sistem içinde farklı derivasyonların bir arada kullanıldığı gözlenmektedir [5,13,14]. Tez içinde incelenen tüm EKG vuruları 2 nolu standart bipolar derivasyonla alınmıştır, farklı derivasyonların kullanılmasına gerek kalmamıştır. Tek bir derivasyon kullanımı sistem karmaşıklığını da azaltmaktadır. Pentium III-450 MHz işlemcili bir bilgisayarda MATLAB paket programı ile süreler incelenmiş; normalizasyon işleminin 0.03 sn, dalgacık analizinin 0.05 sn, frekans analizinin 0.06 sn ve GetNic ağı için sınıflama işleminin 0.07 sn sürdüğü gözlenmiştir. Bu değerler Windows98 işletim sistemi çalışırken ölçüldüğü için süreler, 100 vuru ortalaması alınarak hesaplanmıştır. Ölçümler, çalışmanın gerçek zamanda vuru sınıflama işleminde kullanılabileceğini göstermektedir.

Hasta üzerinden gerçek zamanda alınan EKG kayıtlarının saklanması problemi değişik sıkıştırma teknikleri kullanılarak giderilmektedir. Tez çalışmasında öznitelik çıkartma yöntemi olarak önerilen Daubechies-2 dalgacık dönüşümü aynı zamanda veri sıkıştırma çalışmalarında da kullanılmaktadır. Özniteliklerin dalgacık analizi ile elde edilmesi sıkıştırma ve sınıflama işlemlerinin paralel gerçekleşmesine imkan vermektedir. EKG işareti sıkıştırma işlemi için dalgacık katsayılarına dönüştürülürken paralelinde sınıflama işlemi için de kullanılmaktadır. EKG'nin sıkıştırılırken sınıflanarak etiketlenmesi veri analizinin daha hızlı gerçeklenmesine olanak sağlayacaktır.

Bilginin daha az parametre ile daha iyi temsil edilmesi genel bir problemdir. Çalışmada özellikle yeni sınıflayıcı yapıları araştırmakla bu genel probleme değişik bir noktadan bakılması amaçlanmıştır. Hiper düzlemlerin, prizmaların ve kürelerin kesişimi ile bilginin saklanması, parametre sayısına bağlı olarak incelenmiştir. Gerçeklenen yapay sinir ağlarının EKG işaretlerinin sınıflanması konusu dışında kullanılabilmesi için sadece özniteliklerin, probleme göre tekrar belirlenmesi yeterli olacaktır, sınıflayıcılar hiçbir değişiklik yapılmadan yeni eğitim ve test kümeleriyle kullanılabilecektir.

KAYNAKLAR

[1] **Cohen, A.,** 1986. Biomedical signal processing volume II, CRC Press Inc., 75-79.

[2] **Yazgan, E. ve Korürek, M.,** 1996. Tıp elektroniği, İ.T.Ü., Elektrik-Elektronik Fakültesi, Ofset Baskı Atölyesi.

[3] **MIT-BIH Arrhythmia Database.** 1992. Available from Massachusetts Institute of Technology, 77 Massachusetts Avenue, Room 20A-113, Cambridge, MA 02139, USA.

[4] **Goldman, M. J.,** 1979. Principles of clinical electrocardiography, Lange Medical Publications, California.

[5] **Reinhardt, L., Vesanto, R., Montonen, J., Fetsch, T., Makijarvi, M., Sierra, G. and Breithardt, G.,** 1996. Application of learning vector quantization for localization of myocardial infarction, *18th Annual International Conference of the IEEE-EMBS*, paper no. 211.

[6] **Yonghong, K., Jie, J., Yecho, H. and Zhicheng, L.,** 1998. Paced ECG analysis based on neural networks, *20th Annual International Conference of the IEEE-EMBS*, 206-209.

[7] **Risk, M. R., Sobh, J. F. and Saul, J. P.,** 1997. Beat detection and classification of ECG using self organizing maps, *19th Annual International Conference of the IEEE-EMBS*, 89-91.

[8] **Palreddy, S., Tompkins, W. J. and Hu, Y. H.,** 1995. Customization of ECG beat classifiers developed using SOM and LVQ, *17th Annual International Conference of the IEEE-EMBS*, paper no. 778.

[9] **Chazal, P. and Celler, B. G.,** 1998. Selecting a neural network structure for ECG diagnosis, *20th Annual International Conference of the IEEE-EMBS*, 1422-1425.

[10] **Silipo, R. and Marchesi, C.,** 1998. Artificial neural networks for automatic ECG analysis, *IEEE Transactions on Signal Processing*, May, vol. 46, no. 5, 1417-1425.

[11] **Silipo, R., Gori, M., Taddei, A., Varanini, M. and Marchesi, C.,** 1995. Classification of arrhythmic events in ambulatory electrocardiogram, using artificial neural networks, *Computers and Biomedical Research*, vol. 28, no. 4, 305-318.

[12] **Hu, Y. H., Palreddy, S. and Tompkins, W. J.,** 1997. A patient-adaptable ECG beat classifier using a mixture of experts approach, *IEEE Transactions on Biomedical Engineering*, September, vol. 44, no. 9, 891-900.

[13] **Bortolan, G. and Fusaro, S.,** 1996. Feature reduction and RBF in classifiers based on ANN, *18th Annual International Conference of the IEEE-EMBS*, paper no. 819.

[14] **Silipo, R., Bortolan, G. and Marchesi, C.,** 1996. Supervised and unsupervised learning for diagnostic ECG classification, *18th Annual International Conference of the IEEE-EMBS*, paper no 1054.

[15] **Dokur Z., Ölmez, T. and Yazgan, E.,** 1999. ECG waveform classification using the neural network and wavelet transform, *First Joint Conference of the BMES-IEEE EMBS*, Atlanta USA.

[16] **Dokur, Z., Ölmez, T. and Yazgan, E.,** 1999. Comparison of discrete wavelet and Fourier transforms for ECG beat classification, *Electronics Letters*, vol. 35, no. 18, 1502-1504.

[17] **Dokur, Z., Ölmez, T., Yazgan, E. and Ersoy, O. K.,** 1997. Detection of ECG waveforms by neural networks, *Medical Engineering and Physics*, December, vol. 19, no. 8, 738-741.

[18] **Ölmez, T., Dokur, Z. and Yazgan, E.,** 1998. Classification of ECG waveforms using a novel neural network, *20th Annual International Conference of the IEEE-EMBS*, Hong Kong, vol. 20, no.3, 1616-1619.

[19] **Ölmez, T., Dokur, Z. and Yazgan, E.,** 1997. Classification of ECG waveforms by using genetic algorithms, *19th Annual International Conference of the IEEE-EMBS*, Chicago, USA, 92-94.

[20] **Dokur, Z., Ölmez, T., Korürek, M. and Yazgan, E.,** 1996. Detection of ECG waveforms by using artificial neural networks, *18th Annual International Conference of the IEEE-EMBS*, Amsterdam, paper no.1038.

[21] **Dokur, Z., Ölmez, T. ve Yazgan, E.,** 1996. Genetik algoritmalar kullanılarak EKG şekil bozukluklarının belirlenmesi, *Biyomedikal Mühendisliği Ulusal Toplantısı (BİYOMUT'96)*, Ekim, 43-46.

[22] **Dokur, Z., Ölmez, T., Yazgan, E. ve Korürek, M.,** 1995. ECG şekil bozukluklarının yapay sinir ağları ile belirlenmesi, *Biyomedikal Mühendisliği Ulusal Toplantısı (BİYOMUT'95)*, 159-163.

[23] **Dokur, Z., Ölmez, T., Korürek, M. ve Yazgan, E.,** 1995. Bulanık sınıflayıcı ile EKG şekil bozukluklarının tespit edilmesi, *3. Sinyal İşleme ve Uygulamaları Kurultayı*, B: İşaret İşleme, Nisan, 208-213.

[24] **Minami, K., Nakajima, H. and Toyosima, T.,** 1999. Real-Time discrimination of ventricular tachyarrhythmia with Fourier-Transform neural network, *IEEE Transactions on Biomedical Engineering,* February, vol. 46, no. 2, 179-185.

[25] **Yao, J., Gan, Q., Zhang, X. and Li, J.,** 1998. Pruning algorithm in wavelet neural network for ECG signal classification, *20th Annual International Conference of the IEEE-EMBS*, 1482-1485.

[26] **Afonso, V. X., Wieben, O., Tompkins, W. J., Nguyen, T. Q. and Luo, S.,** 1997. Filter bank-based ECG beat classification, *19th Annual International Conference of the IEEE-EMBS*, 97-100.

[27] **Paul, J. S., Reddy, M. R. S. and Kumar, V. J.,** 1997. Automatic detection of PVC's using autoregressive models, *19th Annual International Conference of the IEEE-EMBS*, 68-71.

[28] **Jouny, I., Hamilton, P. and Kanapathipillai, M.,** 1994. Adaptive wavelet representation and classification of ECG signals, *16th Annual International Conference of the IEEE-EMBS*, paper no. 48.

[29] **Fredric, M. H. and Soowhan, H.,** 1996. Classification of Cardiac Arrhythmias, *IEEE Transactions on Biomedical Engineering*, April, vol. 43, no. 4, 425-430.

[30] **Celler, B. G. and Chazal, P.,** 1998. Low computational cost classifiers for ECG diagnosis using neural networks, *20th Annual International Conference of the IEEE-EMBS*, 1337-1340.

[31] **Wieben, O., Tompkins, W. J. and Afonso, V. X.,** 1997. Classification of PVCs with a fuzzy logic system, *19th Annual International Conference of the IEEE-EMBS*, 65-67.

[32] **Szilágyi, S. M.,** 1998. Event recognition, seperation and classification from ECG recordings, *20th Annual International Conference of the IEEE-EMBS*, 236-239.

[33] **Hosseini, H. G. and Nazeran, H.,** 1998. Detection and extraction of the ECG signal parameters, *20th Annual International Conference of the IEEE-EMBS*, 127-130.

[34] **Elias, A., Leija, L., Alvarado, C., Hernández, P. and Gutiérrez, A.,** 1997. Personal computer system for ECG recognition in myocardial infarction diagnosis based on an artificial neural network, *19th Annual International Conference of the IEEE-EMBS*, 1095-1096.

[35] **Yang, M., Hu, W. and Shyu, L.,** 1997. ECG events detection and classification using wavelet and neural networks, *19th Annual International Conference of the IEEE-EMBS*, 280-281.

[36] **Rioul, O. and Vetterli, M.,** 1991. Wavelets and signal Processing, *IEEE Signal Processing Magazine*, October, vol. 8, no. 4, 14-38.

[37] **Daubechies I,** 1994. Ten Lectures on Wavelets, Capital City Press, Vermont.

[38] **Sahambi, J. S., Tandon, S. N. and Bhatt, R. K. P,** 1997. Using wavelet transforms for ECG characterization, *IEEE Engineering in Medicine and Biology*, January/February, vol. 16, no. 1, 77-83.

[39] **Senhadji, L., Carrault, G., Bellanger, J. J. and Passariello, G.,** 1995. Comparing wavelet transforms for recognizing cardiac patterns, *IEEE Engineering in Medicine and Biology*, March/April, vol. 14, no. 2, 167-173.

[40] **Hecht-Nielsen, R.,** 1990. Neurocomputing, HNC, Inc. and University of California, San Diego.

[41] **Güzeliş, C.,** 1993. Yapay Sinir Ağları, Yüksek Lisans Ders Notu, İstanbul Teknik Üniversitesi, İstanbul.

[42] **Kröse, B. J. A. and Patrick, P. S.,** 1991. An introduction to neural networks, University of Amsterdam.

[43] **Reilly, D. L., Cooper, L. N. and Elboum C.,** 1982. Neural model for category learning, *Biological Cybernetics*, 45, 35-41.

[44] **Alpaydın, E.,** 1990. Neural models of incremental supervised and unsupervised learning, *Ds. Thesis*, Ecole Polytechnique Federale De Lausanne, Switzerland.

[45] **Lippmann, R. P.,** 1987. An introduction to computing with neural nets, *IEEE ASSP Magazine,* April, pp. 4-22.

[46] **Miller, A. S., Blott, B. H. and Hames T. K.,** 1992. Review of neural network applications in medical imaging and signal processing, *Medical & Biological Engineering & Computing*, 30, 449-464.

[47] **Dokur, Z., Ölmez, T. and Yazgan, E.,** 1998. Classification of MR and CT images using genetic algorithms, *20th Annual International Conference of the IEEE-EMBS,* Hong Kong, vol. 20, no.3, 1418-1421.

[48] **Dokur, Z., Ölmez, T. and Yazgan, E.,** 1997. Classification of magnetic resonance images by using genetic algorithms, *19th Annual International Conference of the IEEE-EMBS*, Chicago, USA, 1391-1393.

[49] **Dokur, Z., Ölmez, T. and Yazgan, E.,** 1996. Biomedical image classification by using artificial neural networks, *8th Mediterranean Electrotechnical Conf. (MELECON'96),* Italy, May, vol. 3, 1469-1471.

[50] **Ölmez, T., Dokur, Z. and Yazgan, E.,** 1996. MR image classification by the neural network and the genetic algorithms, *18th Annual International Conference of the IEEE-EMBS,* paper no.1039.

[51] **Ölmez, T., Dokur, Z. and Yazgan, E.,** 1996. Genetik algoritmalar ve çok tabakalı ağ kullanılarak biyomedikal görüntülerin sınıflandırılması, *4. Sinyal İşleme ve Uygulamaları Kurultayı,* Nisan, 433-437.

[52] **Ölmez, T.,** 1997. Classification of ECG waveforms by using RCE neural network and genetic algorithms, *Electronics Letters,* 28th August, vol. 33, no. 18, 1561-1562.

[53] **Holland, J. H.,** 1992. Genetic Algorithms, *Scientific American*, July, 44-50.

[54] **Goldberg, D. E.,** 1989. Genetic algorithms in search optimization and machine learning, Addison Wesley Pub. Comp., Inc.

[55] **Schaffer, J. D., Whitley, D. and Eshelman, L. J.,** 1992. Combination of genetic algorithms and neural networks: A survey of the state of the art, *Proc. Int. Wks. on Combination of Genetic Algorithms and Neural Networks (COGANN-92)*, 1-37.

[56] **Frank, Z. B., Donald, E. B. and Worthy, N. M.,** 1992. Fast genetic selection of features for neural network classifiers, *IEEE Transactions on Neural Networks*, vol.3, no. 2, 324-339.

[57] **Harp, S. A., Samad, T. and Guha, A.,** 1990. Designing application specification of neural networks using the genetic algorithms, *Advances in Neural Information Processing Systems,* 2, 447-454.

[58] **Harp, S. A. and Samad, T.,** 1991. Genetic optimization of self-organizing feature maps, *IEEE International Joint Conference on Neural Networks (IJCNN-91 Seattle)*, July, pp. I341-I346.

[59] **Dominic, S., Das, R., Whitley, D. and Anderson C.,** 1991. Genetic reinforcement learning for neural networks, *IEEE International Joint Conference on Neural Networks (IJCNN-91 Seattle)*, July, pp. II397-II404.

[60] **Ölmez, T., Dokur, Z. ve Yazgan, E.,** 1997. Genetik algoritmalar kullanılarak rakamların tanınması, *5. Sinyal İşleme ve Uygulamaları Kurultayı,* Mayıs, 2.cilt, 713-716.

[61] **Dokur, Z., Ölmez, T. ve Yazgan, E.,** 1997. RCE ağı ve genetik algoritmalar kullanılarak cisim tanıma, *5. Sinyal İşleme ve Uygulamaları Kurultayı,* Mayıs, 1.cilt, 250-253.

[62] **Dokur, Z., Ölmez, T., Yazgan, E. ve Korürek, M.,** 1996. Yapay sinir ağları yardımıyla rakamların tanınması, *4. Sinyal İşleme ve Uygulamaları Kurultayı,* Nisan, 146-150.

[63] **Ölmez, T.,** 1995. Yapay sinir ağları yardımıyla biyomedikal dokuların sınıflandırılması, *Doktora Tezi*, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

[64] **Vittorio, M.,** 1994. Genetic evolution of the topology and weight distribution of neural networks, *IEEE Transaction on Neural Networks*, vol. 5, no. 1, 39-53.

[65] **Bruce, A. W. and Timothy, D. C.,** 1994. Evolving space-filling curves to distribute radial basis function over an input space, *IEEE Transactions on Neural Networks*, vol. 5, no. 1, 15-23.

[66] **Makhoul, J., Jaraudi, A. E. and Schwartz, R.,** 1989. Formation of disconnected regions with a single hidden layer, *IJCNN International Joint Conference on Neural Networks*, 1, 455-460.

[67] **Afonso, V. X., Tompkins, W. J., Nguyen, T. Q. and Luo, S.,** 1999. ECG beat detection using filter banks, *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 2, 192-202.