KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

İNSAN YÜZLERİ ARASINDAKİ BENZERLİĞİN TEMEL BİLEŞENLER ARACILIĞI İLE ARAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisi Yasemin BEKİROĞLU

TEMMUZ 2007
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

İNSAN YÜZLERİ ARASINDAKİ BENZERLİĞİN TEMEL BİLEŞENLER ARACILIĞI İLE ARAŞTIRILMASI

Bilgisayar Mühendisi Yasemin BEKİROĞLU

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"Bilgisayar Yüksek Mühendisi"
Ünvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 19.06.2007
Tezin Savunma Tarihi : 05.07.2007

Tez Danışmanı : Prof. Dr. Vasif V. NABİYEV
Jüri Üyesi : Yrd. Doç. Dr. Hüseyin PEHLİVAN
Jüri Üyesi : Yrd. Doç. Dr. Ali GANGAL

Enstitü Müdürü :Prof. Dr. Emin Zeki BAŞKENT

Trabzon 2007

ÖNSÖZ

Yüksek Lisans eğitimim ve özellikle de tez çalışması süresince yol gösteren, zaman ayıran Sayın Hocam Prof. Dr. Vasıf NABİYEV'e teşekkürlerimi sunarım.

Tüm eğitim hayatım boyunca bana en iyi fırsatları sunan annem ve babama teşekkür ederim.

Yüksek Lisans eğitimime destek olan TÜBİTAK' a teşekkürlerimi sunarım.

Yasemin BEKİROĞLU

Trabzon 2007

İÇİNDEKİLER

ÖZET

Tezde, yüzler arasındaki benzerliğin değerlendirilmesi ele alınmaktadır. Benzerlik tespiti, sorulan yüze benzer olanların bulunmasıdır ve bir sistemin bunu gerçekleştirmesi için, insanın birçok etkene rağmen yüzler arasındaki benzerliği farkedebilme yeteneğini taklit edebilmesi gerekir. Yüzler için mümkün değişimleri dikkate alacak şekilde verimli bir temsil şekli belirlemek ve seçilen temsil şeklini kullanarak yeni bir yüzü sınıflamak gerçekleştirilmesi gereken işlemlerdir.

İncelenecek yüzlerin üzerinde çalışılan resimlerden elde edilebilmesi için yüz tespiti işlemi uygulanmaktadır. Çalışmada yüz tespiti bölümünde yoğun olarak çok katmanlı yapay sinir ağı kullanılmıştır. Ağ, yüz ve yüz olmayan resimler arasında seçim yapabilmesi amacıyla eğitilmektedir. Yüz tespiti işleminde, ağın eğitiminden sonra ikinci aşama olarak, olası tüm yüzlerin bulunabilmesi amacıyla görüntünün çeşitli boyutlara küçültülerek taranması gerçekleştirilmektedir.

Yüz tespiti kısmında, incelenen giriş resmindeki olası yüzler ten bölgelerinde içerileceğinden, ten bölgelerine odaklanılarak üzerinde çalışılan resim bölümünün azaltılması sağlanmıştır. Bu amaçla, ten bölgelerinin belirlenmesinde giriş resmi YCbCr renk uzayına taşınmaktadır. Ten işleme bölümünden elde edilen sonuç yapay sinir ağına dayalı sorgulama kısmında işlenmektedir.

Benzerlik incelemesi bölümünde ele alınan temel yöntem Temel Bileşen Analizi'dir. Temel Bileşen Analizi, orjinal resimleri, daha az boyutlu uzayda ifade şekillerini bularak, aralarındaki benzerlikler ve farklılıklar açısından yorumlama imkanı sunmaktadır. Temel Bileşen Analizi' nden elde edilen sonuçların değerlendirilmesinde farklı uzaklık ölçme yöntemleri, bu yöntemlere dayalı oylama ve yapay sinir ağı kullanılmaktadır. Ayrıca, Temel Bileşen Analizi ile elde edilen sonuçlarla karşılaştırmak amacıyla görüntülerin çıkarma işlemiyle kıyaslanması da gerçekleştirilmiştir.

Benzerlik incelemesi için belirlenen veritabanı üzerinde yapılan sorgulamalarda benzer yüzlerin bulunması başarıyla gerçekleştirilmiştir ve veritabanında olmayan bir yüzle sorgulama yapıldığında da veritabanındaki en yakın yüz bulunabilmektedir.

Anahtar Kelimeler: Yüz Tespiti, Yapay Sinir Ağları, Temel Bileşen Analizi, Görüntü İşleme

SUMMARY

Evaluation of Similarity Between Human Faces with Principal Component Analysis

The thesis contains the examination of determining similarity between human faces. Detection of similarity can be defined as listing faces similar to a given face and to be able to achieve it a system should imitate the talent with which humans recognize faces despite a lot of factors. Defining a new way of representing faces taking possible changes into account and classifying a given face using the chosen way are the required operations.

To obtain faces to decide how similar they were, firstly a face detection on a given image was performed. In the face detection part of the work, a multilayer neural network was implemented. The network was trained to distinguish faces from non-faces. After training the network, to detect all faces image containing faces was scanned while it was scaled down.

In face detection, since possible faces on input image are included on the skin part, skin parts were focused to decrease the image part to be searched for faces. With this aim, to detect skin parts input image was converted into YCbCr color space. The result obtained in the skin detection was processed in examination based on the neural network in the face detection part.

Principal Component Analysis is the method preferred in similarity evaluation part of the work. Principal Component Analysis provides an efficient way of comparing faces according to the similarities and differences between them representing faces on a lower dimensional space. Four distance measures, voting based on them and the neural network deisgned were utilized in evaluation of the results obtained in Principal Component Analysis. Additionally, comparing face images by subtraction was performed to compare with the result obtained using Principal Component Analysis.

Listing similar faces in images included in a database designed to be used in similarity evaluation was performed succesfully and most similar faces to a given face which was not incuded in the database could also be listed.

Keywords: Face Detection, Artificial Neural Networks, Principal Component Analysis, Image processing.

ŞEKİLLER DİZİNİ

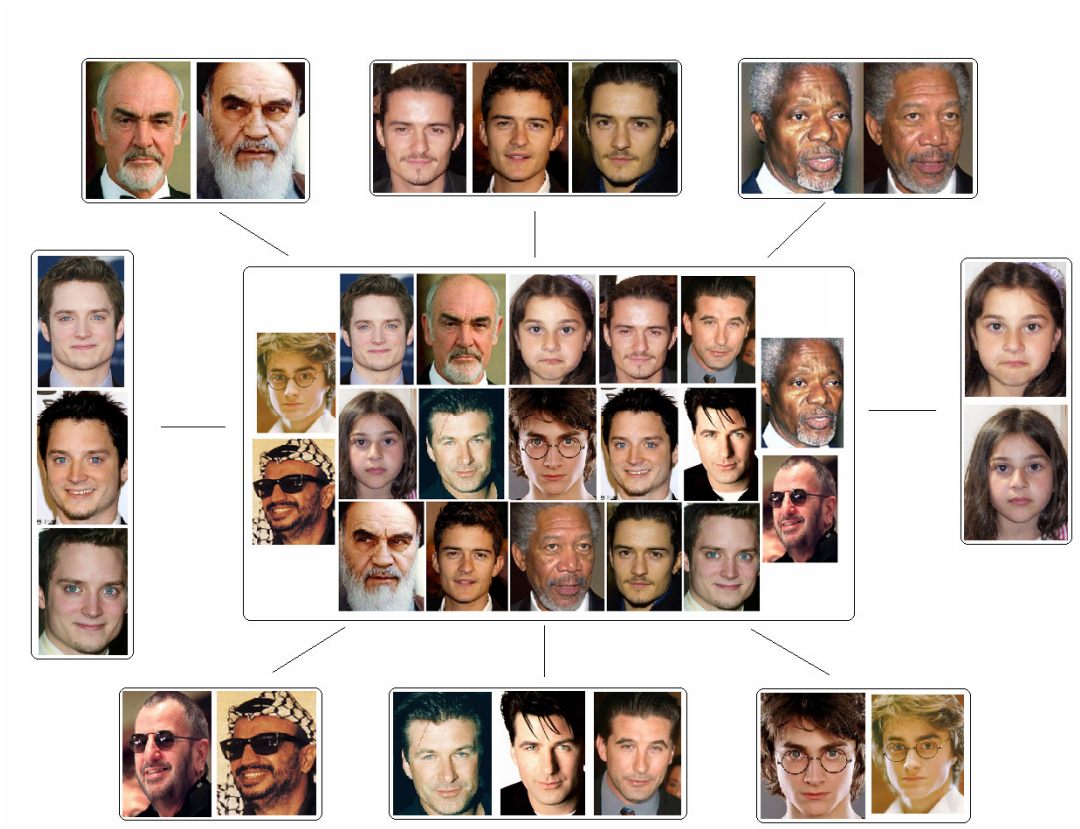
TABLOLAR DİZİNİ

Sayfa No

1. GENEL BİLGİLER

1.1. Giriş

Benzerlik tespitinde aynı kişiye ait değişik etkenler altındaki farklı resimlerin bulunabilmesinin yanısıra farklı kişiler arasındaki yakınlığın elde edilebilmesi de kapsanmaktadır. Şekil 1'de bu durum örneklendirilmektedir.



Şekil 1. Kişilerin benzerlik durumları

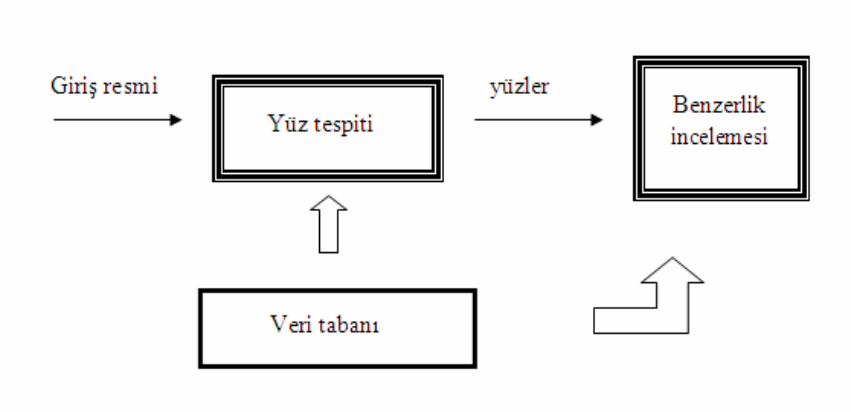
Yüz tanıma; kimlik tespiti, insan bilgisayar etkileşimi, otomatik güvenlik sistemleri gibi birçok alanda uygulanmaktadır. Yüz tanıma sistemlerindeki verimliliğin arttırılmasında benzer yüzlerin doğru tespiti önem taşımaktadır. Işıklandırma, duruş şekli, kişinin duygusal durumu, yaşlanma etkisi gibi farklı etkenlerden ötürü yüz tanıma hataya açık ve zor bir problemdir. Yüzler arasındaki benzerliğin ölçümü farklı etkenlerin sonucu etkilemedeki rolüne ışık tutar.

Benzerlik tespiti, sorulan yüze benzer olanların bulunmasıdır ve bir sistemin bunu gerçekleştirmesi için, insanın birçok etkene rağmen yüzler arasındaki benzerliği farkedebilme yeteneğini taklit edebilmesi gerekir. Yüzler için mümkün değişimleri dikkate alacak şekilde verimli bir temsil şekli belirlemek ve seçilen temsil şeklini kullanarak yeni bir yüzü sınıflamak gerçekleştirilmesi gereken işlemlerdir.

1.2. Çalışmanın Genel Yapısı ve Amacı

Çalışmada, kişilere ait farklı resimlerden oluşan veritabanı üzerinde yapılan sorgulamalarla benzer yüzlerin bulunması gerçekleştirilmektedir. Ayrıca veritabanından olmayan bir yüzle sorgulama yapıldığında da veritabanındaki en yakın yüz bulunabilmektedir.

Sistemin yüzü otomatik olarak inceleyebilmesi için öncelikle arka plan bilgisini eleyip yüzü elde etmesi gerekir. Bu amaçla çalışma yüz tespiti ve benzerlik incelemesi olmak üzere temel olarak iki kısımdan oluşmaktadır. Şekil 2'de bu çalışma düzeni verilmektedir.



Şekil 2. Yüz tespiti ve benzerlik incelemesi çalışma düzeni

1.3. Biyometri

Her teknolojinin temelinde canlı varlıklara benzeştirme, onların karakteristiklerinden yararlanarak yöntemler, cihazlar, makineler geliştirme amacı vardır. Yani her şeyin özünde anlaşılabilen karakteristik özellik bir şekilde insanların yaşam tarzlarını ve işlerini kolaylaştırmak için kullanılmaktadır. Bu fikirden hareketle insanları birbirinden ayırt

edebilme, dolayısıyla onları tanımlayabilme başta güvenlik konuları olmak üzere başka birçok alan açısından önemini git gide arttırmaktadır. İnsanlar birbirlerini, yüz yapısı; gülüş şekli; saçlarının rengi veya uzunluğu kısalığı; boyunun kısalığı uzunluğu gibi birçok şekilde tanımlanabilecek dış görünüş özellikleri ve/veya yürüyüş şekli, konuşma tarzı, sesi gibi davranışsal özellikleri yanı sıra daha birçok özellikleri ile zihinlerinde resmederler. Böylece daha sonra tekrar karşılaştıklarında onlar hakkında hatırlarında tuttukları özellikleri ile onları tanırlar. Dolayısıyla günümüzün teknolojik ihtiyaçları için insanların hafızaya resmetme, tanımlama becerilerinin benzeştirilmesi gerekmektedir.

Bu anlamda, biyometri, fiziksel ve davranışsal özeliklere dayalı olarak insanları tanımlamak için kullanılan yöntemler bütünü olup bu bütünü modelleyen sistemlere de "biyometrik sistem" denir [1]. Günümüzde kullanılan başlıca biometrik özellikler; parmak izi, retina, iris, avuç içi bilgisi, el ve yüz yapısı, ses, dişler ve DNA olarak sıralanabilir. Biometik sistemlerin en büyük üstünlüğü, bu gibi ayırt ediciliği çok yüksek özelliklerin tahmin dilmesi, kaydedilmesi, benzeştirilmesi imkânsıza yakın zor olmasıdır. Öyle ki, insandan insana değişen ve bu nedenle aktarılması zor olan bu özellikler biyometriyi güvenlik sistemlerinde en çok tercih edilen alan haline dönüştürmüştür. Diğer biyometri uygulama alanları bilgisayarlı kimlik tespiti ve kişinin tanınması, yapay zeka araştırmaları, on-line alışveriş, elektronik ticaret, internet erişimidir [2, 3].

Biyometrik tanıma sistemlerinde kullanılan yöntemler benzerlik göstermektedir. Önce kayıtlar alınmakta ve kendilerine has biçimlerde kodlanıp kaydedilmektedirler. Daha sonra örnekler alınıp kodlandıktan sonra mevcut kayıtlarla karşılaştırılıp eşleme yapılmaktadır [3].

Biyometrik sistemlerde amaç, en kısa zaman içinde en doğru biçimde yanılma payı en düşük olacak şekilde eşleme yapmaktır. Pratikte bu hedeflere ulaşma konusunda birçok zorluk bulunmaktadır. Örneğin en kısa zamanda belirlenen birçok özelliği kıyaslamak başlı başına zor bir görevdir. Ayrıca el ve parmak izi tanımada yara veya silinmeler; ses tanımada içinde bulunan ortamdaki gürültü kirliliği, hastalıklardan dolayı ses değişimleri; yüz tanımada zaman içindeki fiziksel değişimler ve yüzdeki kırışıklıklar, yüzdeki ifade değişiklikleri tanıma işini zorlaştırmaktadır. Bu zorluklara rağmen uygulamalardaki amaç, en doğru sonuçları üretecek sistemleri geliştirmektir [3].

Geliştirilen biyometrik tanıma sistemleri uygulamada genellikle bir biyometrik özelliğe göre karşılaştırma şeklinde özelleşmiştir. Bu anlamda retina, parmak izi gibi bazı özelliklerin karşılaştırmasını temel alan yöntemlerin başarı oranları oldukça yüksek

olmasına rağmen her uygulamada uygulanabilirliği ve kullanım kolaylığı istenildiği gibi olamamaktadır. Öte yandan, yüz ve ses tanıma sistemlerinin her ikisinde de kullanım kolaylığı ve uygulanabilirlik istenilen düzeyde olmasına karşın doğruluklarındaki yanılma oranları yüksektir [3].

1.4. Antropometri ve Kanon (Orantılar Kuralı)

Her vücudun birbirinden farklı olduğunu gösteren oranlar bütünü vardır. Bu bütünü inceleyen bilim dalına "Antropometri" denmektedir. Bu bilim dalı, insan vücudunun ya da vücudun bir bölümünün ölçülerini ve orantılarını inceler. Öyle ki, insan vücutları veya vücutlarının bölümleri her ne kadar çeşitlik gösterseler de özde belli, değişmeyen oranlara sahiptirler. Örneğin, kulaklarda farklı olabilecek kulak çeşitleri incelendiğinde, yirmi adet kulak çeşidi bulunduğu ve bunları beş burun tipi, yedi çeşit göz rengi vb. değişik düzenlemelerde birleştirildiğinde birbirlerine hiç benzemeyen son derece farklı yüzlerin ortaya çıktığı görülür [4].

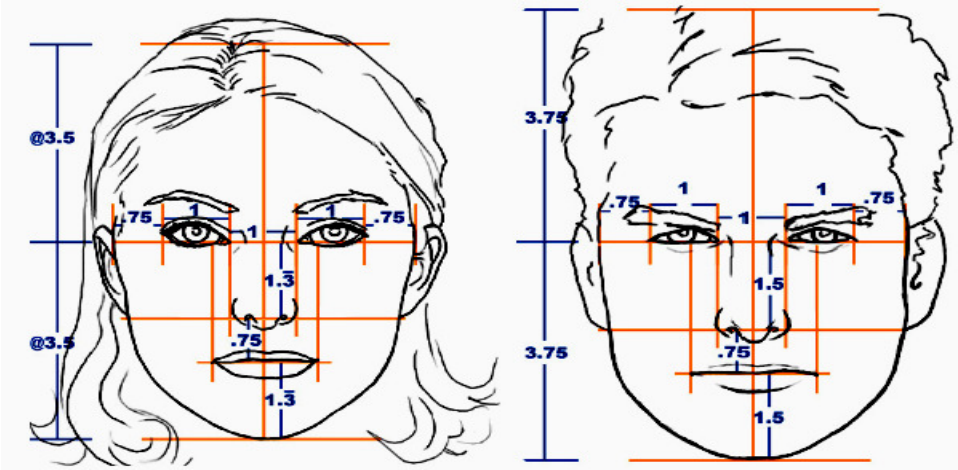
Antropometri, ayrıca, morfoloji (biçimbilim) ve anatomiyi tamamlayan bir bilim dalı olarak; ırk, cinsellik ve yaş açısından binlerce vücudun orantılarını kıyaslayarak araştırmış; yüz yıllardan bu yana kullanılan ölçüleri incelemiş; günümüze, vücut ölçüleri ve orantıları ile ilgili daha doğru bilgiler sağlamıştır. Örneğin insan boyları ölçülmüş ve ortalama olarak boy değişimleri üç grup olarak gözlemlenmiştir. Bunlar 165 cm, 175 cm ve 175 cm üstü olarak bulunmuştur. Benzer olarak baş yükseklikleri üzerine yapılan bir çalışmada baş yüksekliğinin 21,5 cm ve 24 cm arasında değiştiği ve vücut ölçüleri içinde en az değişen ölçü olduğu fark edilmiş ve başın yüksekliği 22,5 cm olarak kabul edilmiştir [4].

Her ne kadar yukarıdaki örneklerde verildiği gibi verilen ölçülerde ortalama değerler kullanılabilirse de oransal (modüler) ölçüler kullanmak daha sağlıklı görünmektedir. Bu anlamda, ölçü ve orantıları saptamak için de "Kanon" veya "Orantılar kuralı" olarak bilinen bir sistem kullanılmaktadır. Kanon, oran (modül) olarak ifade edilen bir ölçü biriminden yararlanarak insan vücudunun oran ve boyutlarını saptayan bir ölçme sistemidir. Bu sistem, yukarıda da ifade edilen kıyaslamalı çalışmalarla elde edilmiş olup orantıların belirlenmesi ve bu orantılardan yola çıkarak ölçülendirmenin yapılabilmesinde önemli bir yol göstericidir [4].

1.5. Yetişkin İnsan Yüzüne ait Kanon Değerleri

İnsan vücudunun tamamında olduğu gibi her bir kısmı (örneğin, insan yüzü) içinde kanon değerleri belirlenmiştir. İnsandan insana oranların en az değişim gösteren bölümü de yüz olduğu düşünülürse yüze ait kanon değerlerini kullanmak daha doğru sonuçlar sunacaktır. Böylelikle yüz tanımlanmasında bu değerler kullanılarak yüz ayrıklaştırma işlemi gerçekleştirilebilir. Yüze ait kanon değerleri ayrıca insanların çocuk, yetişkin ve yaşlı olmalarına göre değişebilmektedir. Bu bölümde ortalama olarak yetişkin insan yüzü düşünülüp, yetişkin insan yüzüne ait oranlar verilecektir.

Yüze ait oranlar, bu oranların belirlenmesinde temel alınan birim boyun tanımlanmasıyla elde edilir. Birim boy olarak, gözler arası mesafe dikkate alınmaktadır. Öyle ki, birim boy, bir göz boyu olarak tanımlanmaktadır. Buna göre yetişkin bir kadın ve erkeğe ait kanon değerleri Şekil 3'te gösterilmektedir.



Şekil 3. Yetişkin kadın ve erkeklerde temel yüz oranları [3].

Şekil 3'te görünen yüz geometrisine göre göz, ağız, burun ve bunun gibi yüz bileşenlerinin göreceli konumları ve genişlik veya uzunlukları elde edilebilmektedir. Örneğin, gözlerin konumunu, erkek yüzü üzerinde ifade etmek gerekirse; gözler, düşeyde yüzün ortasındadır. Yatayda ise kendilerinden düşey olarak indirilen doğruların yüzün düşey simetri eksenine göre geometrik ortasındadır.

1.6. Yüz Tespiti ve Kıyaslamada Kullanılan Yöntemler

Görüntülerden otomatik yüz tespitini gerçekleştirmek için birçok yöntem uygulanmaktadır. Örneğin, giriş resminin standart bir örnek yüzle olan ilişkisini belirleyen yöntem şablon eşlemeye dayalıdır (template matching) [5, 6]. Kullanılan yöntemlerden görüntü tabanlılar (image based) sinir ağları ve istatistiksel analiz içerir [7] ve metodlar arasında en iyi sonuç verenlerdendir. Yapay sinir ağları ile yüz tespiti konusunda en önemli çalışmalardan biri Rowley ve diğerleri [8,9] tarafından yapılmıştır. Geliştirdikleri sistemde görüntüler ön işleme tabi tutularak eğitilmiş çok katmanlı sinir ağına verilir ve ağ çıkışına göre sınıflandırmaları yapılır. Herbir pikseli ten rengine yakınlığına göre etiketlemek ve ortaya çıkan bölgeleri büyüklüklerine göre yüz olarak sınıflandırmak renk tabanlı bir yöntemdir. Yüzlerin farklı duruşlarında da başarılı sonuç vermektedir fakat belirlenen ten rengi hassasiyetinde çalışmaktadır[10]. En çok tercih edilen renk uzayları RGB, normalize edilmiş RGB, YCbCr, HSI (Hue, Saturation, Intensity) dir. Ten renginin ayırd edilebilmesi seçilen renk uzayına bağlıdır[11].

Benzerlik ilişkisi göz önüne alınarak uygulanan değerlendirmelerden; geometrik özelliklere dayalı yöntemlerde [12, 13, 14 ve 15], yüz bileşenleri (göz, burun, ağız ve çene) belirlenir ve bu bileşenler arasındaki alan, uzaklık, açı gibi özellik ve ilişkiler yüzleri tanımlada kullanılır. Veri azaltmada etkili ve ekonomik olmalarına ve ışıklandırma ile bakış açısındaki değişimden etkilenmemelerine rağmen bu yöntemler yüz bileşenlerinin ölçümüne ve özellik çıkarmaya karşı çok hassastırlar. Öte yandan, özellik çıkarma ve ölçüm yöntemleri ile bugüne kadar geliştirilen algoritmalar bu ihtiyacı giderecek kadar yeterli güvenilirlikte değildir [15,16]. Şablon eşleme (template matching) ve sinir ağlarına dayalı yöntemler [17, 18] genelde yüzlerin görüntü tabanlı temsilini (örneğin, piksel değerleri dizisi) doğrudan yapmaktadırlar. Geometrik yüz bileşenlerinin ölçümüne ve belirlenmesine gerek olmadığı için bu yöntemler geometrik yüz tabanlı yöntemlerle karşılaştırıldıklarında uygulanmaları daha pratik ve kolaydır [15]. En başarılı şablon eşleme yöntemlerinden biri, yüzün ifade edilebilmesi ve saptanabilmesi için kullanılan Karhunen Loeve dönüşümü veya temel bileşen analizine dayanan özyüz yöntemidir [19, 15]. Veri tabanındaki her yüz resmi, özyüz uzayında karakteristik eksenlere verilen bileşenlerden oluşan bir vektör ile temsil edilirler. Genelde yüz saptanmasında en yakın uzaklık kriteri (the nearest distance criterion) kullanılmaktadır [15]. Birçok yüz tanıma yaklaşımı içerisinde, görüntü tabanlı alt uzay analizi en iyi sonuçları verir. Alt uzay analizi

görüntü daha düşük boyutlu uzaya (alt uzaya) izdüşüm yapılarak sağlanır ve sonra tanıma bilinen görüntüyle tanınacak görüntü arasındaki uzaklık ölçülerek gerçekleştirilir. Böyle bir sistemin en zor tarafı yeterli bir alt uzay bulmaktır. En çok tercih edilen izdüşüm metodları Temel Bileşen Analizi (Principal Component Analysis, TBA), Bağımsız Bileşen Analizi (Independent Component Analysis, BBA) ve Doğrusal Ayrıklaştırma Analizi (Linear Discriminant Analysis, DAA)'dir [20].

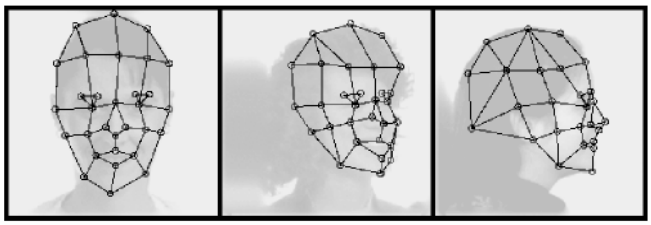
TBA [21, 19 ve 22], yüz tanıma için yaygın olarak kullanılan karakteristik dönüşüm yöntemlerinden biridir. TBA [19], en iyi temsili sağlayan izdüşüm vektörlerini (Şekil 4) bulur ve izdüşümle dönüştürülen örnekler orjinal örnekler hakkında en çok bilgiyi kapsamaktadırlar[20]. TBA yaklaşımı [23], veri boyutunu veriyi sıkıştırarak [24] azaltır ve yüzün en verimli düşük boyutlu yapısını ortaya çıkarır. Bu boyut azaltma yararlı olmayan bilgiyi ortadan kaldırır [25] ve yüz yapısını özyüz olarak bilinen ortogonal (ilişkisiz) bileşenlere tam olarak ayrıştırır. Herbir yüz görüntüsü özyüzlerin ağırlıklandırılmış toplamı (özellik vektörü) olarak ifade edilebilir. Kıyaslamalar özyüzlerden hesaplanan özellik vektörleri arasındaki uzaklık ölçülerek gerçekleştirilir. TBA [26], varyans gibi ikinci derece istatistik bilgisi ile ilgilenirken; BBA, [26,27] ikinci ve daha yüksek dereceden istatistikleri yakalar ve giriş verisini istatistiksel olarak mümkün olan en bağımsız vektör tabanına (Şekil 4) izdüşürür[20]. BBA [26], giriş verisindeki ikinci ve daha yüksek dereceden bağımlılıkları azaltır ve dönüştürülen veriyi kendisi boyunca istatistiksel olarak bağımsız yapan tabanı bulmaya çalışır. BBA, lineer bir ortogonal olmayan koordinat sistemi belirler. Bu koordinat sisteminin eksenlerinin doğrultusu orjinal verideki ikinci ve daha yüksek dereceden istatistiklerle belirlenir. Amaç istatistiksel olarak bağımsızlık sağlayan lineer bir dönüşüm sağlamaktır [28]. DAA [29, 30], TBA ve BBA'nın aksine sınıflandırma bilgisi kullanmakta ve Fisher Ayrıklaştırma Kriteri'nin en büyük olduğu vektörler kümesini elde etmektedir. DAA, [29, 31] sınıf bilgisini kullanır ve sınıf içi dağılımı en küçük yaparken sınıf arası dağılımı en büyük yapan vektörleri (Şekil 4) bulur[20]. DAA, bilinmeyen sınıf örneklerini sınıflandırmak için bilinen sınıflı eğitim örneklerine dayalı istatistiksel bir yaklaşımdır [25]. Bu teknik sınıf arası varyansı maksimum, sınıf içi varyansı minimum yapmayı amaçlar (sınıflar arasında en iyi ayrımı sağlayan uzayın vektörlerini bulur). Büyük boyutlu yüz verisi ele alındığında, bu teknik; örnek uzayı boyutuna göre az sayıda eğitim örneği varken, küçük örnek boyutu problemi ile karşılaşır[32][23]. TBA ve BBA, danışmansız öğrenme yöntemleri olarak

adlandırılabilirken; DAA, eğitim sürecinde her bir resim için sınıf bilgisine ihtiyaç duyduğundan danışmanlı öğrenme yöntemi olarak adlandırılmaktadır [28].

Bu üç algoritmada sınıflandırma önce giriş resimlerini bir izdüşüm matrisi ile bir alt uzaya taşıyarak ve sonra girişin izdüşümle elde edilen katsayı vektörünü diğer görüntülere veya sınıflara ait vektörlerle kıyaslayarak yapılır. Çeşitli gruplar bu algoritmalar için çeşitli sonuçlar elde etmiştir. Zhao [31] TBA ve DAA' nın birlikte DAA dan daha iyi sonuç verdiğini rapor etmiştir. Beveridge [33], farklı uzaklık ölçümleri kullanarak yaptıkları testlerde TBA algoritmasının DAA' dan iyi sonuç verdiğini belirtmektedirler. Martinez [30], küçük bir eğitim seti için (her sınıf için 2 görüntü) TBA' nın , DAA' dan iyi sonuç verdiğini ileri sürmüştür. Beak [34], TBA' nın BBA' dan başarılı sonuçlar ürettiğini belirtmiştir[28].

Model tabanlı yöntemlerde insan yüzünün, yüz değişimlerini yakalayacak modelini inşa etmek amaçlanır [35]. Yüz bileşenlerinin yerinden bağıl konumları ve uzaklıkları çıkarılır. İnsan yüzleri benzer topolojik yapıya sahiptir. Yüz belli noktalarda bulunan düğüm ve uzaklık vektörleri ile etiketlenmiş kenarlarla yapılandırılabilir yani yüz grafı çıkarılabilir. Yüz deste grafı örnek yüz görüntülerinden çıkarılır ve genel bir temsil ifadesi olarak kullanılabilir. Düğümler yeni yüz üzerinde belirlendikten sonra yüz tanıma oluşturulan bu graf ve diğerleri arasında benzerlik kıyaslanarak gerçekleştirilir. Model tabanlı yöntemlerden olan Elastik Graf Eşleme (Elastic Bunch Graph Matching, EGE), yüz görüntülerinin ışıklandırmadaki değişimler, duruş şekli, psikolojik yüz ifadesi gibi lineer olmayan özelliklerini dikkate alır [23]. Bir Gabor dalgacık dönüşümü (Gabor wavelet transform) yüzü bir esnek ızgara üzerine izdüşüren dinamik bağlantı mimarisi sağlar [25]. Esnek ızgara üzerindeki düğümler (Şekil 5'teki dairelerle ifade edilen), piksel etrafındaki görüntü davranışını ifade eder ve görüntünün bir Gabor filtresi ile konvolusyonunun sonucudur. Tanıma, herbir Gabor düğümündeki Gabor filtresi tepkisinin benzerliğine dayalı olarak yapılır [25]. Bu metoddaki zorluk, doğru sınır işareti belirlemelerinin sağlanma zorunluluğudur ve bu durum bazen TBA ve DAA metodlarının birleştirilmesiyle başarılır [25]. PCA ve LDA birlikte kullanıldığında EGE' den daha hızlı ve iyi sonuç vermektedir [36,37]. EGE, sınıf başına bir eğitim görüntüsü olduğunda tek seçenek olarak görülmektedir [36]. Sınıf başına fazla eğitim örneği olduğunda da, eğer görüntüler çok küçükse ve gözler ideal bir şekilde bulunamıyorsa EGE kullanılmalıdır [36]. Örneğin farklı bir çalışmada da, gözleri kapalı kişilerin TBA kullanılarak daha kolay tanındığı fakat EGE ile tanımanın daha zor olduğu belirtilmiştir[38].

Şekil 4. Şekil ORL veritabanından [39] elde edilen TBA, BBA ve DAA taban vektörlerinden bazıları [35].



Şekil 5. Esnek deste harita graf (Elastic Bunch Map Graphing) [40, 23].

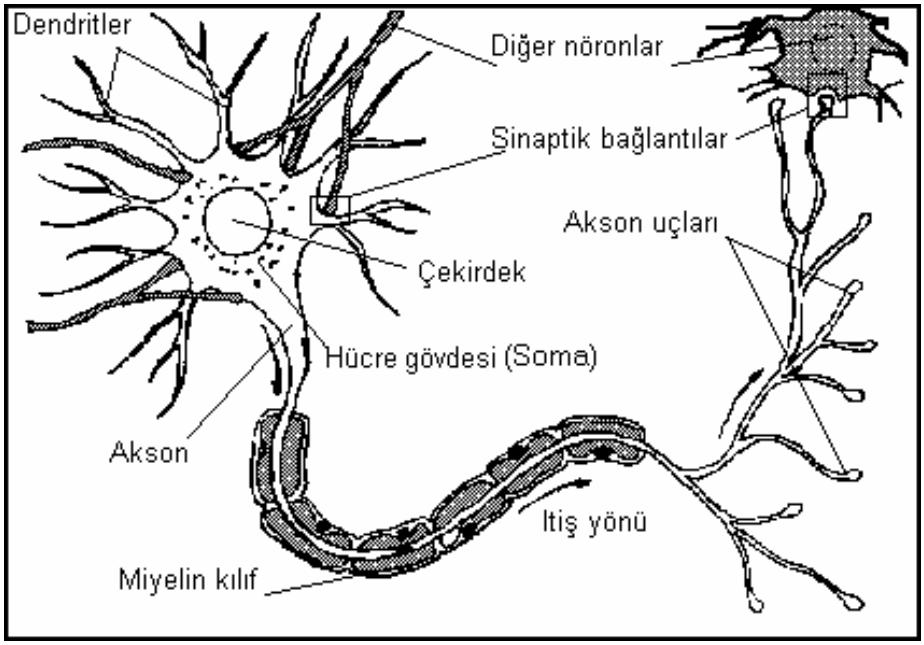
1.7. Yapay Sinir Ağları

Bu bölümde kullanılan tekniklerden biri olan Yapay Sinir Ağları'nın yapısı anlatılmaktadır [41, 42]. Bölüm sonunda kullanılan model üzerinde yapılan uygulamalardan bahsedilmektedir.

1.7.1. Sinir Ağı Nedir

Bir sinir ağı insan beynine dayalı muhakeme modeli olarak tanımlanabilir. Beyin yoğun olarak birbirine bağlı sinir hücrelerinden (temel bilgi işleme birimleri) yani *nöron*lardan oluşur. İnsan beyninde yaklaşık olarak 10 milyar nöron ve nöronlar arasında 60 trilyon bağlantı yani *sinaps* bulunmaktadır. Beyin birçok nöronu aynı anda kullanarak işlevlerini çok hızlı gerçekleştirmektedir.

Bir nöron, *soma* adı verilen hücre gövdesi, *dendrit* adı verilen uzantıları ve *akson* adı verilen uzun tek bir uzantıdan oluşmaktadır. Dendritler soma etrafında ağ şeklinde dallanırken, akson ise diğer nöronların soma ve dendritlerine doğru uzanır. Şekil 6, bir sinir hücresinin yapısını göstermektedir.



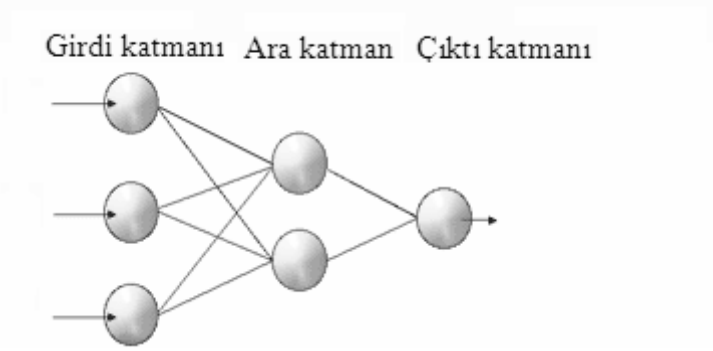
Şekil 6. Biyolojik sinir hücresi

Sinyaller karmaşık elektro-kimyasal reaksiyonlarla bir nörondan diğerine yayılır. Sinapslardan salınan kimyasal bileşenler hücre gövdesinin elektriksel potansiyelinde değişime neden olur. Potansiyel eşik değere ulaşınca elektriksel bir darbe akson boyunca gönderilir. Darbe potansiyellerinin artmasına ya da azalmasına neden olarak sinapslara ulaşarak yayılır. Uyarı modeline cevapta, nöronlar bağlantılarının gücünde uzun süreli değişiklikler gösterirler. Nöronlar diğer nöronlarla yeni bağlantılar oluşturabilirler. Bu mekanizmalar beyindeki öğrenmenin temelini oluştururlar.

Beynimiz oldukça karmaşık, lineer olmayan ve paralel bilgi işleme sistemi olarak düşünülebilir. Bilgi bir sinir ağında tüm ağ boyunca aynı anda işlenir ve saklanır. Nöronlar arasında yanlış cevaba giden bağlantılar zayıflatılırken, doğru cevaba giden bağlantılar güçlendirilir. Sonuç olarak, sinir ağları deneyimle öğrenme yeteneğine sahiptir. Öğrenmek biyolojik sinir ağlarının temel ve gerekli bir özelliğidir. Öğrenmelerindeki kolaylık ve doğallık, biyolojik sinir ağlarının bilgisayarla gerçekleştirilmesine olanak sağladı.

Yapay Sinir Ağları (YSA), öğrenme yeteneğine sahiptir yani başarımlarını arttırmak için deneyimden faydalanırlar. Yeterli miktarda örnek verildiğinde YSA karşılaşmadığı diğer örnekler için genelleme yapabilir.

Bir YSA, *nöron* adı verilen ve beyindeki biyolojik nöronlara benzer olan birbirlerine bağlı işlem birimlerinden oluşur. Nöronlar, bir nörondan diğerine sinyaller geçiren ağırlıklandırılmış bağlantılarla bağlıdır. Herbir nöron bağlantılarıyla giriş sinyalleri alır ve bir çıkış sinyali üretir. Çıkış sinyali nörondan çıkan (biyolojik aksona karşılık gelen) bağlantıyla gönderilir. Bu bağlantı aynı işareti gönderen dallara ayrılır. Dallar ağdaki diğer nöronlara gelen bağlantılarda sonlanır. Şekil 7, YSA' daki bağlantıları göstermektedir ve Tablo 1'de biyolojik sinir ağı ve YSA arasındaki benzerliği ifade etmektedir. Şekil 7'de görüldüğü gibi YSA katmanlardan oluşur ve ağdaki nöronlar bu katmanlara yerleşmiştir. Dış çevreye bağlı nöronlar giriş ve çıkış katmanlarını oluşturur.



Şekil 7. YSA' nın yapısı

Tablo 1. Biyolojik sinir ağı ve YSA arasındaki benzerlik

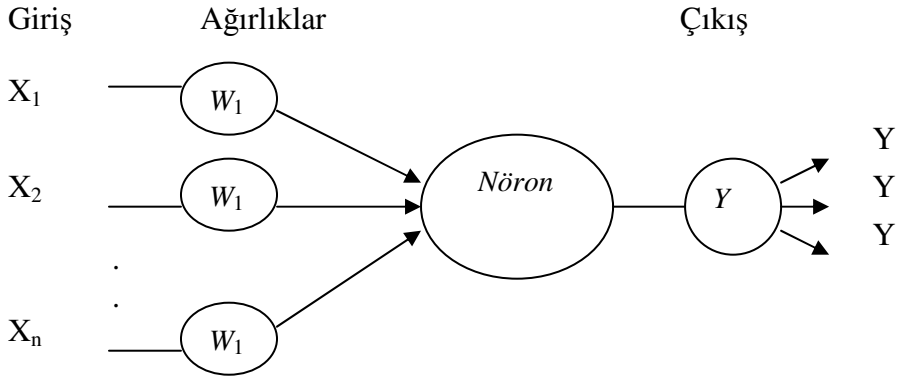
Biyolojik sinir ağı	Yapay sinir ağı
Soma	Nöron
Dendrit	Giriş
Akson	Çıkış
Sinaps	Ağırlık

Nöronları ilişkilendiren bağlantılar sayısal ağırlıklara sahiptir. Ağırlıklar ANN' deki temel bellek mantığını oluşturur ve herbir nöron girişinin gücünü yani önemini ifade eder. Bir YSA, bu ağırlıkların tekrarlanan güncellemeleriyle öğrenir.

Herbir nöron bilgi işleme birimidir ve giriş ile ağırlıklar verildiğinde aktiavsyon seviyesini hesaplar. Bir YSA inşa etmek için kullanılacak nöron sayısı ve nöronların bağlantı şekli belirlenmelidir. Daha sonra hangi öğrenme algoritmasının kullanılacağına karar verilmelidir. Son olarak da sinir ağı eğitilir yani ağdaki ağırlıklara başlangıç değeri verilir ve eğitim örneklerine göre ağırlıklar güncellenir.

1.7.2. Nöron

Nöron, giriş bağlantılarından sinyaller alır, yeni bir aktivasyon seviyesi hesaplar ve çıkış bağlantılarıyla çıkış sinyali olarak gönderir. Giriş sinyali veri ya da diğer nöronların çıkışı olabilir. Çıkış sinyali ya problemin çözümü ya da diğer nöronlara giriş olabilir. Şekil 8 bir nöronun yapısını göstermektedir.



Şekil 8. Nöron yapısı

Nöron, giriş sinyallerinin ağırlıklandırılmış toplamını hesaplar ve sonucu bir eşik Θ ile kıyaslar. Sonuç eşikten küçükse nöron çıkışı -1'dir, eşikten büyük veya eşiğe eşitse çıkış 1'dir. Böylece nöronun kullandığı aktivasyon fonksiyonu aşağıdaki gibidir.

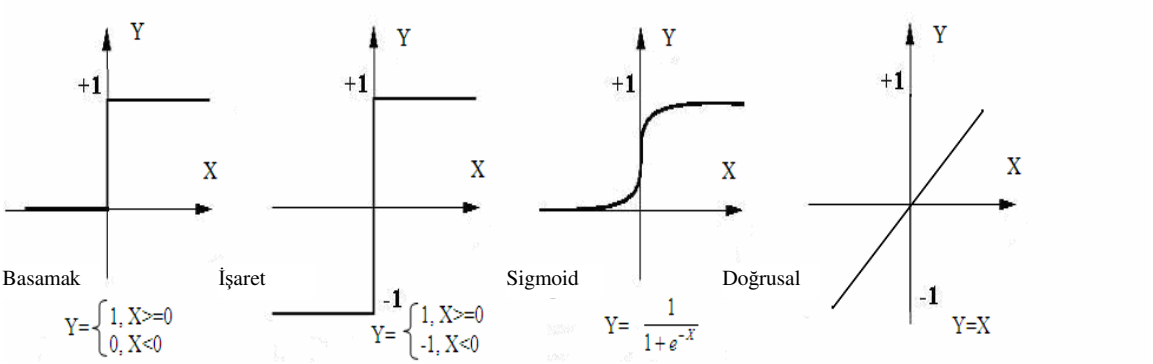
$$X = \sum_{i=1}^{n} x_i \, w_i \tag{1}$$

$$Y = \begin{cases} +1 \text{ eğer } X >= \Theta \\ -1 \text{ eğer } X < \Theta \end{cases}$$

X, nöronun ağırlıklandırılmış girişi; x_i i. giriş değeri; w_i i. girişin ağırlığı; *n* nöronun giriş sayısı ve *Y* de nöronun çıkışıdır. Buradaki aktivasyon fonksiyonu *işaret (sign) fonksiyonu*dur. Böylece işaret aktivasyon fonksiyonu ile nöronun çıkışı aşağıdaki şekilde ifade edilebilir.

$$Y=sign(\sum_{i=1}^{n} x_i w_i - \theta \tag{2}$$

Birçok aktivasyon fonksiyonu test edilmiştir ancak birkaçı pratik olarak uygulama bulmuştur. Bunlardan dördü Şekil 9'da görülmektedir. *Basamak* ve *işaret* fonksiyonları (*hard limit functions*) sınıflandırma ve tanıma işlemlerinde sıkça kullanılmaktadır.

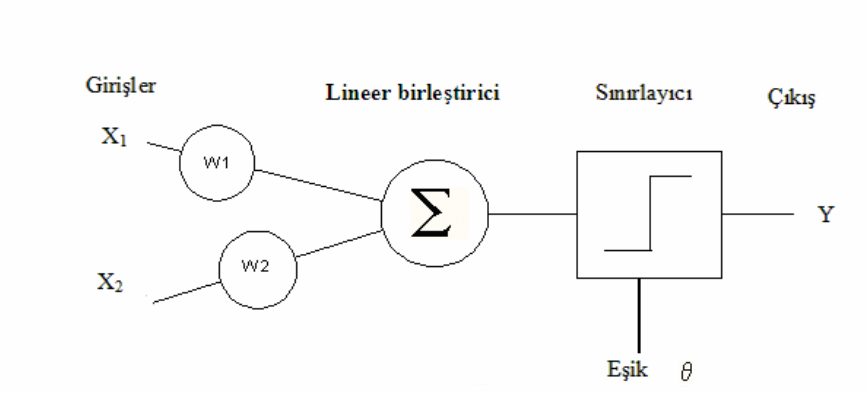


Şekil 9. Aktivasyon fonksiyonları

*Sigmoid fonksiyon*u herhangi bir giriş değerini 0-1 aralığında bir değere dönüştürür. Bu fonksiyon *geri yayılım ağları*nda kullanılmaktadır. *Doğrusal aktivasyon fonksiyonu* nöronun ağırlıklandırılmış girişine eşit çıkış üretir.

1.7.3. Perceptron

Perceptron bir sinir ağının en basit şeklidir. Ayarlanabilir ağırlıklı tek nörondan ve bir sınırlayıcıdan (*hard limiter*) oluşur. Şekil 10'da tek katmanlı ve iki girişli perceptron görülmektedir.

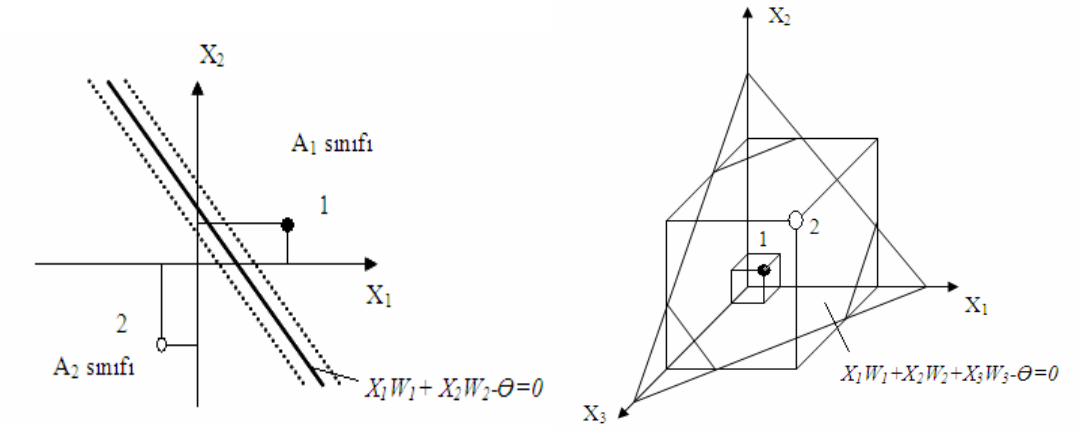


Şekil 10. Tek katmanlı iki girişli perceptron

Perceptronun yapısında lineer birleştirici (linear combiner) ve bunu izleyen sınırlayıcı (hard limiter) vardır. Girişlerin ağırlıklandırılmış toplamı sınırlayıcıya verilir ve çıkışından girişi pozitifse 1, negatifse -1 alınır. Perceptronda amaç girişleri sınıflamak yani bir başka deyişle x_1, x_2, ..., x_n girişlerini örneğin A_1 ve A_2 sınıflarına yerleştirmektir. Böylece, n boyutlu uzay bir hiper düzlem ile iki karar bölgesine ayrılır. Hiper düzlem aşağıdaki lineer olarak ayrılabilir (linearly separable) fonksiyonla tanımlıdır.

$$\sum_{i=1}^{n} x_i w_i - \theta = 0 \tag{3}$$

2 giriş durumunda, x_1 ve x_2, karar sınırı Şekil 11.a'da gösterildiği gibi bir doğru şeklini alır. Sınır doğrusunun üzerinde kalan 1 noktası A_1 sınıfına ait, doğrunun altında kalan 2 noktası da A_2 sınıfına ait olarak değerlendirilmektedir. Eşik Θ, karar sınırını kaydırmak için kullanılabilir.



(a) (b)

Şekil 11. (a) 2 girişli perceptrondaki lineer ayırd etme, (b) 3 girişli perceptrondaki lineer ayırd etme

Şekil 11.b'de görülen 3 girişli perceptron için bulunan ayırma düzlem aşağıdaki eşitlikle tanımlıdır.

$$x_1 w_1 + x_2 w_2 + x_3 w_3 - \theta = 0 \tag{4}$$

Perceptronda sınıflandırma işlemi, perceptronun gerçek çıkışı ile hedeflenen çıkışı arasındaki farkı azaltacak şekilde ağırlıklarda ayarlama yapılması ile gerçekleştirilir. Başlangıç ağırlıkları rastgele olarak atanır, genellikle [-0.5, 0.5] aralığındadır ve sonra ağırlıklar çıkışları eğitim örnekleriyle tutarlı halde elde etmek için güncellenir. Perceptron için ağırlık güncelleme süreci basittir. İterasyon *p*'de gerçek (hesaplanan) çıkış *Y(p)*, ve hedeflenen çıkış $Y_h(p)$ olarak alındığında hata aşaüıdaki şekilde olur.

$$e(p) = Y_h(p) - Y(p) \qquad p=1,2,3... \tag{5}$$

Burada iterasyon *p*, *p*. eğitim örneğinin perceptrona sunulmasını ifade etmektedir. Eğer hata, *e(p,)* pozitifse perceptron çıkışı *Y(p)* arttırılmalıdır; hata negatifse de çıkış azaltılmalıdır. Herbir perceptron girişi toplam giriş *X(p)*' ye $x_i(p)*w_i(p)$ kadar katkıda bulunduğundan, giriş değeri $x_i(p)$ pozitifse ağırlığı $w_i(p)$'deki artış perceptron çıkışı *Y(p)'* yi arttırmaya eğilimlidir; giriş negatifse de ağırlığındaki artış çıkışı azaltmaya eğilimlidir. Buradan aşağıdaki *perceptron öğrenme kuralı* çıkarılabilir.

$$w_i(p+1)=w_i(p)+\alpha *x_i(p)*e(p) \tag{6}$$

α, 1' den küçük pozitif bir sabittir ve *öğrenme katsayısı* olarak adlandırılır. Perceptron öğrenme kural kullanılarak perceptronun sınıflandırma işlemi için eğitim algoritması aşağıdaki şekilde çıkarılabilmektedir.

adım 1: Başlangıç

Ağırlıklara ($w_1, w_2,...,w_n$) ve eşik 'e (Θ) başlangıç değerleri [-0.5, 0.5] aralığından rastgele olarak atanır.

adım 2: Etkinleştirme (aktivasyon fonksiyonundan geçirme)

Giriş değerleri $x_1(p), x_2(p), ..., x_n(p)$ ve hedeflenen çıkış $Y_h(p)$ uygulanarak perceptron aktifleştirilir ve çıkış hesaplanır.

$$Y(p)=step[\sum_{i=1}^{n} x_i(p)w_i(p)-\theta\,] \quad (7)$$

Burada *n* perceptron giriş sayısıdır ve *step*, aktivasyon fonksiyonudur.

adım 3: Ağırlık güncelleme

Perceptronun ağırlıklarının güncellenmesi

$$w_i(p+1)=w_i(p)+\Delta w_i(p) \quad (8)$$

şeklinde hesaplanır. $\Delta w_i(p)$, iterasyon p'deki ağırlık değişimidir ve *delta kuralı* ile hesaplanır:

$$\Delta w_i(p) = \alpha * x_i(p)*e(p) \quad (9)$$

adım 4: İterasyon

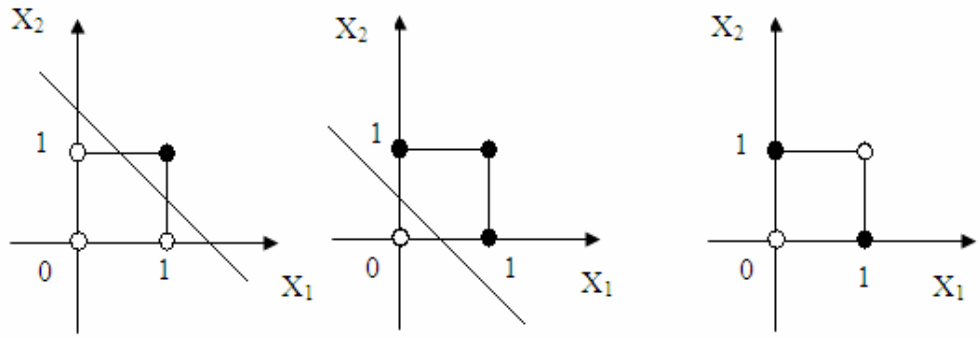
İterasyon sayısı *p* 1 arttırılır ve *adım 2*' ye dönülür. İşlemler yakınsama olana kadar sürdürülür.

1.7.3.1. Perceptronun Temel Lojik İşlemler AND, OR ve XOR İçin Eğitimi

AND, OR ve XOR için doğruluk tablosu Tablo 2' de görülmektedir. Tablo x_1 ve x_2 için tüm mümkün durumları ve sonuçları içermektedir. Perceptron girişleri sınıflandırmak üzere eğitilecektir. AND için, sınıflandırma adımları uygulandığında eğitim başarı ile tamamlanarak kabul edilebilir bir hata değeriyle sonuçların hedeflenene yakın elde edildiği görülmüştür. Benzer şekilde, perceptron OR işlemini de öğrenebilmektedir. Ancak, tek katmanlı perceptron XOR işlemini gerçekleştirmesi için eğitilememektedir. Şekil 12'de AND, OR, XOR fonksiyonları iki girişin değerlerine göre ifade edilmektedir. Fonksiyon çıkışını 1 yapan giriş değerlerini temsil eden noktalar siyah, 0 yapanlar beyaz ile işaretlenmiştir.

Tablo 2. Lojik işlemler için doğruluk tablosu

Girişler X_1	Girişler X_2	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0



(a) (b) (c)

Şekil 12. (a) AND, (b) OR, (c) XOR lojik işlemlerinin 2 boyutlu grafiksel gösterimi

Şekil 12.a ve b'deki siyah ve beyaz noktalar farklı taraflarda kalacak şekilde tek bir doğru ile ayrılabilirler. Fakat, Şekil 12.c'deki siyah ve beyaz noktalar tek bir doğru ile ayrılamamaktadır. Perceptron siyah noktalarla beyaz noktaları ayıracak bir doğru varken karşılık gelen fonksiyonu temsil edebilir. Böyle fonksiyonlara *lineer olarak ayrılabilir* (linearly separable) denir. Sonuç olarak, bir perceptron AND ve OR işlemlerini öğrenebilirken XOR'u öğrenemez.

Bir perceptronun sadece lineer olarak ayrılabilen fonksiyonları öğrenebilmesi (1) eşitliğinden çıkarılmaktadır. Perceptron çıkışı Y sadece toplam ağırlıklandırılmış giriş X eşik Θ'ten büyük veya eşiğe eşitse 1 değerini alır. Böylece tüm giriş uzayı $X = \Theta$ ile tanımlanan sınırla ikiye bölünmektedir. Örneğin, AND için sınır doğrusu aşağıdaki eşitlikle tanımlanmaktadır.

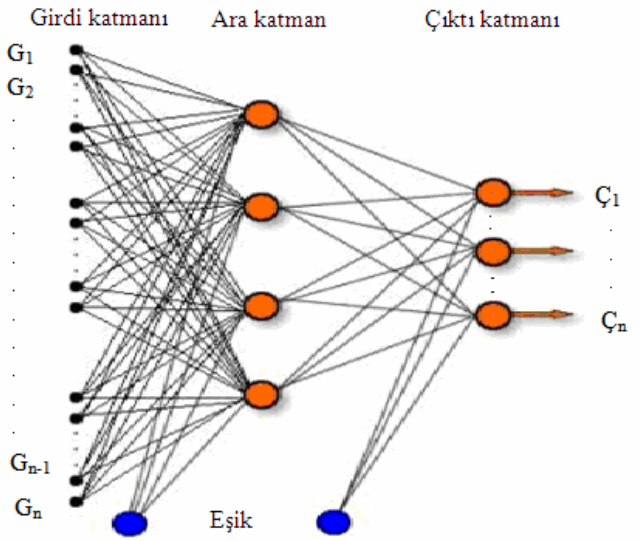
$$x_1 w_1 + x_2 w_2 = \theta \tag{10}$$

Bu doğrunun altında kalan bölge yani çıkışın 0 olduğu bölge $x_1 w_1 + x_2 w_2 - \theta < 0$ ile bulunur, doğrunun üzerindeki yani çıkışın 1 olduğu bölge de $x_1 w_1 + x_2 w_2 - \theta >= 0$ ile bulunur.

Perceptronun sadece lineer olarak ayrılabilen fonksiyonları öğrenebilmesi, bu özellikteki fonksiyonlar çok sayıda olmadığından perceptronların bu açıdan yetersiz kaldığını göstermektedir. Bu problemi aşmak için *çok katmanlı sinir ağları* (*multilayer neural networks*) kullanılmaktadır.

1.7.4. Çok Katmanlı Sinir Ağları

Çok Katmanlı Sinir Ağı (ÇKA), bir ya da daha fazla ara katman içeren sinir ağlarıdır. Yapısal olarak Şekil 13'te de görüldüğü gibi *Girdi Katmanı*, *Ara Katman* ve *Çıktı Katmanı*'ndan oluşmaktadır. Girdi katmanı giriş sinyallerini alır ve ara katmandaki nöronlara dağıtır. Çıktı katmanı ara katmandan gelen sinyalleri alır ve ağın çıkışını üretir.



Şekil 13. ÇKA'nın yapısı

Ara katmandaki nöronlar, özellikleri tespit eder; nöronların ağırlıkları giriş örneklerindeki gizli özellikleri temsil eder. Bu özellikler çıkışı belirlemede çıktı katmanında kullanılır. Bir ara katmanla giriş işaretlerinin herhangibir sürekli fonksiyonu temsil edilebilir, iki ara katmanla süreksiz fonksiyonları dahi temsil edilebilir. Ara katman, hedeflenen çıkış değerini sakladığı için gizli katman olarak da adlandırılabilir. Ara

katmanın hedeflenen çıkışının ne olması gerektiğini bilmek için açık bir yol yoktur. Başka bir deyişle, ara katmanın hedeflenen çıkışı katmanın kendisi tarafından belirlenir. Ticari amaçlı kullanılan YSA'ı bir ya da iki ara katman içermektedirler. Her katman 10-1000 arasında nöron içerebilmektedir. Deney amaçlı YSA'ı üç ya da dört ara katman ve milyonlarca nöron içerebilmektedir. Ama en pratik uygulamalarda toplam üç katman kullanılmaktadır, çünkü her ek katman hesaplama yükünü üstel olarak arttırmaktadır.

Girdi katmanında, giriş değerleri ($G_1, G_2,..., G_n$) alınarak, bilgi işleme yapılmadan ara katmana gönderilmektedir. Gelen her bilgi geldiği gibi ara katmana gönderilmektedir. Her proses elemanının bir girişi ve bir çıkışı vardır. Çıkış, sonraki katmandaki tüm proses elemanlarına gönderilir. Ara katmanda, girdi katmanından gelen bilgiler işlenerek sonraki katmana gönderilir. Ara katmandaki her proses elemanı de sonraki katmandaki tüm proses elemanlarına bağlıdır. Çıktı katmanı da ara katmandan gelen bilgileri işleyerek ağa verilen girişlerin karşılığı olan ağ çıkışlarını ($Ç_1, Ç_2,..., Ç_n$) üretir. Çıktı katmanındaki proses elemanlarının birer çıkışı vardır ve önceki katmanda bulunan tüm proses elemanları ile bağlantılıdırlar.

1.7.4.1. ÇKA' da Öğrenme Kuralı

ÇKA öğretmenli öğrenmeye dayalıdır. Bu ağlara eğitim esnasında giriş ve bu girişlere karşılık üretmesi beklenen çıkışlar birlikte verilir. Girişler ve girişlere karşılık gelen çıkışlar eğitim seti olarak kullanılmaktadır. ÇKA' nın öğrenme kuralı en küçük kareler yöntemine dayalı *Delta Öğrenme Kuralı*nın genelleştirilmiş hali olduğundan *Genelleştirilmiş Delta Kuralı* olarak adlandırılır ve iki aşamadan oluşur: Ağın çıkışının hesaplandığı *ileri doğru hesaplama* ve ağırlıkların güncellendiği *geriye doğru hesaplama*.

1.7.4.1.1 İleri Doğru Hesaplama

Eğitim setindeki bir örnek ağa girdi katmanından sunulur ve burada herhangibir işlem yapılmadan ara katmana gönderilirler. Yani girdi katmanındaki k. proses elemanının çıkışı aşağıdaki şekildedir.

$$Ç_k^i = G_k \tag{11}$$

Ara katmandaki her proses elemanı girdi katmanındaki elemanlardan gelen bilgileri bağlantı ağırlıklarının (A_1, A_2,...) etkisi ile aldığından ara katmandaki elemanlara gelen net giriş aşağıdaki şekilde hesaplanır.

$$NET_j^a = \sum_{k=1}^{n} A_{kj} Ç_k^i \tag{12}$$

A_{kj} *k.* girdi katmanı elemanını *j.* ara katman elemanına bağlayan bağlantının ağırlık değerini göstermektedir. *j.* ara katman elemanının çıkışı ise bu net girişin aktivasyon fonksiyonundan (sigmoid) geçirilmesi ile hesaplanır. Burada türevi alınabilir bir fonksiyon kullanılmaktadır. Sigmoid fonksiyonunun kullanılması ile çıkış aşağıdaki şekilde olacaktır.

$$Ç_j^a = \frac{1}{1 + e^{-(NET_j^a + \beta_j^a)}} \tag{13}$$

Burada β_j, ara katmanda bulunan *j.* elemana bağlanan eşik değer elemanının ağırlığını göstermektedir. Bu eşik değeri biriminin çıkışı sabit olup 1'e eşittir. Ağırlık değeri ise sigmoid fonksiyonunun uyumunu belirlemek üzere konulmuştur. Eğitim esnasında ağ bu değeri kendisi belirlemektedir.

Ara katmanın bütün proses elemanları ve çıktı katmanının proses elemanlarının çıkışları aynı şekilde *NET* girişlerinin hesaplanması ve sigmoid fonksiyonundan geçirilmesi ile belirlenir. Çıktı katmanındaki çıkışlar ($Ç_1$, $Ç_2$, ...) bulununca ileri hesaplama işlemi tamamlanmış olur.

1.7.4.1.2 Geriye Doğru Hesaplama

Ağa sunulan giriş için ağın ürettiği çıkış beklenen çıkışlarla (B_1, B_2, ...) kıyaslanır. Aradaki fark hata değeridir. Geriye doğru hesaplamada bu hata ağırlık değerlerine dağıtılarak bir sonraki iterasyonda hatanın azaltılması sağlanır. Çıktı katmanındaki m. proses elemanı için oluşan hata aşağıdaki şekilde hesaplanır.

$$E_m = B_m - Ç_m \tag{14}$$

E_m bir proses elemanı için oluşan hatadır. Çıktı katmanı için oluşan toplam hatayı (*TH*) bulmak için bütün hataların toplanması gerekir. Bazı hata değerleri negatif olacağından toplamın sıfır olmasını önlemek amacıyla ağırlıkların kareleri hesaplanarak sonucun karekökü alınır. ÇKA ağının eğitilmesindeki amaç bu hatayı en aza indirmektir. *TH*' nin hesaplanması:

$$TH=\frac{1}{2}\sum_{m} E_m^2 \tag{15}$$

Toplam hatayı en aza indirmek için bu hataya neden olan proses elemanlarına hatanın dağıtılması yani proses elemanlarının ağırlıklarının değiştirilmesi gerekir. Ağdaki ağırlıklar, ara katmanla çıktı katmanı arasındaki ağırlıkların ve ara katmanlar arası veya ara katmanla girdi katmanı arasındaki ağırlıkların farklı şekilde değiştirilmesi ile güncellenmektedir.

1.7.4.1.2.1 Ara Katmanla Çıktı Katmanı Arasındaki Ağırlıkların Değiştirilmesi

Ara katmandaki *j*. proses elemanını çıktı katmanındaki *m*. proses elemanına bağlayan bağlantının ağırlığındaki değişim miktarına ΔA^a denirse; *t*. iterasyonda ağırlığın değişim miktarı aşağıdaki şekilde hesaplanır.

$$\Delta A_{jm}^a(t) = \lambda \delta_m Ç_j^a + \alpha \Delta A_{jm}^a(t-1) \tag{16}$$

λ *öğrenme katsayısı*nı, α *momentum katsayısı*nı göstermektedir. Öğrenme katsayısı ağırlıkların değişim miktarını, momentum katsayısı da ÇKA' nın öğrenmesi esnasında yerel bir optimum noktaya takılıp kalmaması için ağırlık değişim değerinin belirli bir oranda bir sonraki değişime eklenmesini sağlar. δ_m *m*. çıkış biriminin hatasını göstermektedir ve aşağıdaki şekilde hesaplanır.

$$\delta_m = f'(NET).E_m \tag{17}$$

$f'(NET)$ aktivasyon fonksiyonunun türevidir ve sigmoid fonksiyonu kullanıldığında eşitlik aşağıdaki şekilde olacaktır.

$$\delta_m = Ç_m(1 - Ç_m)E_m \tag{18}$$

Böylece ağırlıkların t. iterasyondaki değerleri aşağıdaki şekilde olacaktır.

$$A^a_{jm}(t) = A^a_{jm}(t-1) + \Delta A^a_{jm}(t) \tag{19}$$

Çıktı katmanında bulunan proses elemanlarının eşik değer ağırlıklarının, $\beta^ç$, değişim miktarı bu birimin çıkışının 1 olması da göz önüne alınarak aşağıdaki şekilde hesaplanır.

$$\Delta\beta^ç_m(t) = \lambda\delta_m + \alpha\Delta\beta^ç_m(t-1) \tag{20}$$

Eşik değerin t. iterasyondaki ağırlığının yeni değeri de aşağıdaki şekilde hesaplanacaktır.

$$\beta^ç_m(t) = \beta^ç_m(t-1) + \Delta\beta^ç_m(t) \tag{21}$$

1.7.4.1.2.2 Ara Katmanlar Arası veya Ara Katman Girdi Katmanı Arasındaki Ağırlıkların Değiştirilmesi

Ara katmanla çıktı katmanı arasındaki ağırlıkların değişiminde her ağırlık için sadece çıktı katmanındaki bir proses elemanının hatası hesaba katılmaktadır. En son ara katmana gelen bilgiler girdi katmanı veya önceki ara katmandan geldiğinden, bu hataların oluşmasında girdi katmanı ve ara katman arasındaki ağrılıkların payı vardır. Bu yüzden girdi katmanı ve ara katman arasındaki ağırlıkların değiştirilmesinde çıktı katmanındaki proses elemanlarının hepsinin hatası hesaba katılır. Bu ağırlıklardaki değişim miktarı ΔA_i, hata terimi δ^a, ağırlıkların yeni değerleri $A^i_{kj}(t)$, eşik değer ağırlıkları $\beta^a_j(t)$ aşağıdaki şekilde hesaplanır.

$$\Delta A_{kj}^{i}(t) = \lambda \delta_{j}^{a} Ç_{k}^{i} + \alpha \Delta A_{kj}^{i}(t-1) \tag{22}$$

$$\delta_{j}^{a} = f^{'}(NET).\sum_{m} \delta_{m} A_{jm}^{a} \tag{23}$$

$$\delta_{j}^{a} = Ç_{j}^{a}(1-Ç_{j}^{a}).\sum_{m} \delta_{m} A_{jm}^{a} \tag{24}$$

$$A_{kj}^{i}(t) = A_{kj}^{i}(t-1) + \Delta A_{kj}^{i}(t) \tag{25}$$

$$\Delta \beta_{j}^{a}(t) = \lambda \delta_{j}^{a} + \alpha \Delta \beta_{j}^{a}(t-1) \tag{26}$$

$$\beta_{j}^{a}(t) = \beta_{j}^{a}(t-1) + \Delta \beta_{j}^{a}(t) \tag{27}$$

Bir iterasyon ileri ve geri hesaplamalar yapılarak tamamlanır. Yeni örnek için sonraki iterasyona geçilir ve aynı işlemler öğrenme tamamlanana kadar sürdürülür.

1.7.5. ÇKA' nın Çalışması

ÇKA' nın hedeflenen problemin çözümünde kullanılması için öncelikle üzerinde çalışılan problemin doğasına uygun bir eğitim seti hazırlanır. Eğitim setindeki örnekler teker teker ağa sunularak ağın örnekleri öğrenmesi yani izin verilen hata miktarı dahilinde çıkış üretmeyi sağlayan ağırlıkların hesaplanması gerçekleştirilir. Proses elemanlarını birbirine bağlayan ağırlıklara ve eşik değer biriminin ağırlıklarına başlangıçta rasgele değerler atanır. Daha sonra ağ öğrenme işlemi esnasında ağırlıkları uygun şekilde günceller. İleri yönde hesaplama ile ağın çıkışı bulunduktan sonra ortaya çıkan hata değerine göre yapılan geriye hesaplamayla ağırlıklar değiştirilir. Üretilen hata belli bir düzeyin altına düşene kadar eğitime devam edilir. Öğrenme işlemi tamamlandıktan sonra test amaçlı sorulan örnekler için ağın ürettiği çıkışlar değerlendirilerek ağın başarısı ölçülür.

Ağın yapısında kaç girdi birimi, kaç ara katman, her ara katmanda kaç proses elemanı olacağına ve çıktı katmanındaki eleman sayısına, öğrenilmesi istenen olaya göre karar verilir.

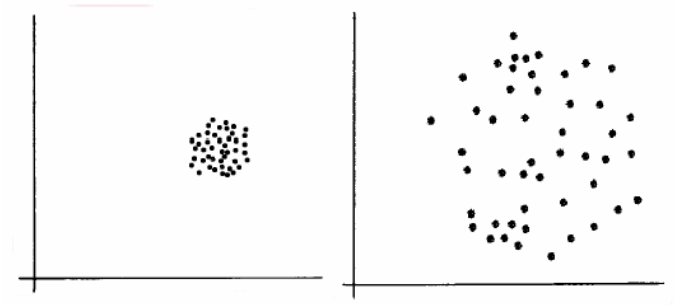
Temel olarak, öğrenmede ağa sunulan giriş değerleri için beklenen çıkışı üretmeyi sağlayacak ağırlıklar aranmaktadır. Başlangıçtaki değerleri rasgele olan ağırlıklar, örnekler

ağa sunuldukça istenen değerlerine ulaşmaktadır. Ancak istenen ağırlık değerlerinin ne olduğu bilinmemektedir. Giriş bilgisi ağ üzerine dağılmakta ve ağırlık değerleri kendi başlarına anlam ifade etmemektedirler. Problem uzayında sonuca ilişkin en az hatayı veren ağırlık değerleri kümesi *en iyi çözüm*dür. Öğrenme esnasında ağ bu çözümü yakalamaya çalışır. Ağın farklı bir çözüme takılması mümkün olduğundan sonuçlarda belli bir tolerans değeri kadar hata kabul edilmektedir. Tolerans değeri altındaki herhangi bir noktada öğrenmenin tamamlandığı kabul edilmektedir. En iyi çözüm olmamalarına rağmen hata düzeyinin altında kalan çözümlere de ulaşılabilir. Bir problem için birden fazla çözüm üretilebileceğinden (*yerel çözümler*) YSA'nın her zaman en iyi çözümü ürettiği söylenemez. Üretilen çözümün en iyi çözüm olduğunu bilmek de zordur. Hatanın belli bir değerin altına düşürülememesinin nedenleri; örneklerin problem uzayını çok iyi temsil etmemeleri, ağ için uygun parametrelerin seçilmemiş olması (öğrenme katsayısı, momentum katsayısı, aktivasyon fonksiyonu), başlangıç değerlerinin uygun olmaması, katman ve katmanlardaki eleman sayısının yetersiz olması gibi etkenler olabilmektedir.

1.8. Temel Bileşen Analizi

1.8.1. Giriş

Benzerlik açısından ele alındığında, yüz resimlerinin piksel değerlerinin karşılaştırılması ile yüksek başarı elde edilmesi güçtür; çünkü her yüzdeki yapı (gözler, ağız, burun, ten tonları) birbirine yakındır. Bu açıdan bakıldığında, yüzler arasında az değişiklik ve yüksek ilişki vardır. Örneğin 100*100 boyutlu her yüz resminin, aslında 10000 boyutlu resim uzayında bir nokta ile ifade edildiği düşünülebilir (Şekil 14.a). Bir yüz resminin diğerleri ile karşılaştırılmasında verimlilik elde etmek için başka bir uzaya (yüz uzayına) dönüşüm işlemi tercih edilmektedir (Şekil 14.b). Bu dönüşüm ile yüzleri temsil eden noktaların birbirlerine göre farklılıkları ve benzerliklerinin irdelenmesi daha başarılı olarak gerçekleştirilebilmektedir. Şekil 14'de bu durum görsel olarak ifade edilmektedir [43]. Bu bölümde kullanılan matematiksel ifadelerin tanımları Ek 1 ve Ek 2' de verilmektedir.



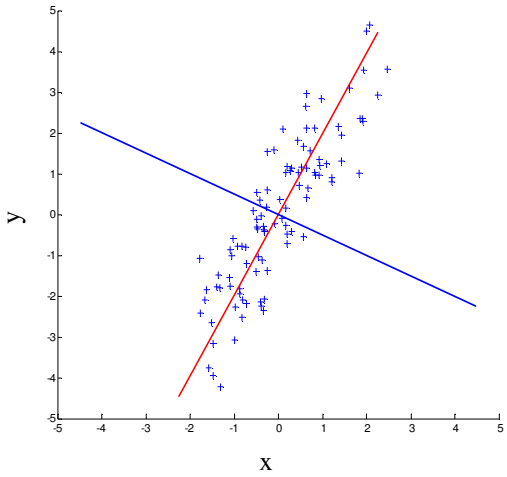
(a) (b)

Şekil 14.(a) resim uzayındaki yüzler (b) yüz uzayındaki yüzler [43]

Bahsedilen dönüşüm işlemi, Temel Bileşen Analizi' ne (TBA) dayalıdır. TBA, burada yüzler arasındaki çok belirgin olmayabilen değişiklikleri belirleme imkânı sunar ve yüzleri burun, kaş genişliği gibi geometrik farklılıkları kullanarak değerlendirmez. Bunun yerine, TBA insan yüzlerinden oluşan bir kümeyi yüzlerdeki değişimi temsil eden birtakım bileşenleri belirlemek için analiz eder [43]. Bu bileşenler *özyüz* olarak ifade edilmektedir ve bir yüz resmi bu özyüzler kullanılarak tanımlanmaktadır. Bir yüz resminin, resim uzayından yüz uzayına izdüşümü yapılınca onu temsil eden vektör ilgili özyüze tekabül eden değer bulunarak oluşturulur. Bu özyüzler, normalize edilmiş yüz resimlerinin kovaryans matrisinin özvektörleridir. Özvektör bulma işlemini hesaplama açısından kolaylaştıran bir teknik kullanılmaktadır[19, 22].

TBA, karmaşık veri kümelerinden anlamlı bilgi çıkarmak amacıyla çok kullanılan bir analiz yöntemidir. TBA, karmaşık veri kümesinin altında yatan basitleştirilmiş yapısını ön plana çıkarmak için küçük boyutlara aktarılmasını mümkün kılar [44]. Başka bir deyişle, verinin içindeki benzerliklerin ve farklılıkların daha iyi ayırdedilmesini sağlayacak şekilde ifade edilmesini sağlar.

TBA'nın yaptığı işlemi grafiksel olarak gözlemlemek için aşağıdaki iki boyutlu (x,y) veri kümesi üzerinde TBA' yı inceleyelim (Şekil 15).



Şekil 15. Veri kümesi için TBA' nın tayin ettiği yeni eksenler [45].

"x" ve "y" den farklı eksenler ile verileri ifade etmenin daha verimli olacağı fikriyle, TBA doğrultusu boyunca verinin varyansının en büyük olduğu ekseni arar. Bu eksenler Şekil 15'te gösterilmektedir. Bu ortogonal eksenler veri kümesinin özelliğini yansıtır. Şekil 15'te kırmızı doğru boyunca verinin dağılımının en büyük olduğu görülmektedir, sonraki en geniş dağılım da mavi doğru boyuncadır. Verilerin en yaygın olduğu eksen sayesinde verilerin birbirlerine göre olan farklılıkları ya da benzerlikleri o eksen üzerindeki izdüşümleri ile daha iyi ayrıştırılabilmektedir.

1.8.2. TBA' nın Amacı

TBA'nın amacı, veri kümesini ifade edecek en anlamlı tabanı hesaplamaktır. Bu yeni taban, veri kümesi içerisindeki gereksiz bilgiyi eleyecek ve gizli yapıyı ortaya çıkaracak özelliktedir. Aşağıda örnek bir veri kümesinin TBA ile hedeflenen yönde dönüşümü anlatılmaktadır.

Veri kümesi a*b boyutlu bir matris *(X)* olarak ele alınıp herbir sütununun kümeye ait örnek vektör olarak işlendiği düşünülürse, herbir vektör a boyutlu ve ortonormal tabanlı bir vektör uzayındandır. Bu vektörler birim uzunluklu taban vektörlerinin lineer kombinasyonlarını oluştururlar.

Herbir sütunu veri kümesinden birer örnek olan orjinal *X* veri kümesini a*b boyutlu *Y* matrisi ile temsil etmek için *P* dönüşüm matrisi *PX=Y* eşitliğiyle ifade edilen şekilde kullanılabilir. *P* matrisinin satırları $\{p_1 , p_2 , \ldots , p_a\}$ *X* in kolonlarını ifade etmek için

kullanılan yeni taban vektörleridir. Y nin herbir kolonu, X in herbir kolonu x_i ile P nin ilgili satırının nokta çarpımı ile elde edilir ($y_i = [p_1 . x_i ; ... ; p_a . x_i]$). Dolayısıyla y_i, $\{p_1 , p_2 , ... , p_a\}$ tabanına izdüşümdür ve P nin satırları X' in kolonlarını yani veri kümesindeki örnekleri temsil etmek için kullanılan taban vektörleridir.

Dönüşümdeki $\{p_1 , p_2 , ... , p_a\}$ satır vektörleri X' in *temel bileşenler*idir ve Y' nin sergileyeceği özellikleri belirlemektedirler.

Orjinal veri kümesi X, fazla gereksiz bilgi içermektedir; diğer bir ifadeyle bir örnekteki herbir bileşen diğerleriyle yüksek ilişki içerisindedir. Dolayısıyla X in kovaryans matrisi, $Cov(X)$, köşegen (diyagonal) değildir. Burada amaç, herbir bileşenin diğerleriyle ilişkide olmamasıdır. Dolayısıyla, yeni bileşenlerin kovaryans matrisi köşegen olmalıdır [46]. Kovaryans matrisinin köşegen şekli, bir bileşenin kendisi ile hesaplanan varyansının en büyük değeri alacağı, diğer bileşenlerle hesaplanan kovaryansın sıfır değeri (ilişki yoksa kovaryans değeri sıfırdır) alacağı anlamına gelir. Böylece farklı bileşenler arasındaki ilişki kaldırılmış olur. Yani X' in dönüştürülmüş hali olan Y'nin kovaryans matrisinin, $Cov(Y)$, aşağıdaki gibi olması hedeflenmektedir. (σ_{ij} i. ve j. bileşen arasındaki kovaryans değerini verir.)

$$Cov(Y) = Y*Y^T = \begin{bmatrix} \sigma_{11} & 0 & ... & 0 \\ 0 & \sigma_{22} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \sigma_{aa} \end{bmatrix} \tag{28}$$

TBA' nın $Cov(Y)$' yi köşegen yapmak için seçtiği yol aşağıda açıklanmaktadır [44]. Kovaryans matrisi $Cov(Y)$' nin $(1/(n-1))YY^T$ şeklindeki açılımında $Y=PX$ dönüşümü yerine yazılırsa

$$\begin{aligned} Cov(Y) &= (1/(n-1))*YY^T \\ &= (1/(n-1))*(PX)(PX)^T \\ &= (1/(n-1))*PXX^T P^T \\ &= (1/(n-1))*P(XX^T)P^T \\ &= (1/(n-1))*PAP^T \end{aligned} \tag{29}$$

elde edilir. Burada XX^T yerine A simetrik matrisi alınmıştır. Simetrik bir matris, özvektörlerinden oluşan ortogonal matris ile aşağıdaki şekilde köşegenleştirilir.

$$A=EDE^T \tag{30}$$

Burada; D köşegen matris, E de A nın özvektörlerinin kolonlara yerleştirilmesiyle elde edilen matristir. P dönüşüm matrisinin herbir satırı, p_i, XX^T' nin özvektörü seçildiğinde, $P=E^T$, $A=P^TDP$ eşitliği elde edilir. P ortogonal olduğu için tersi transpozuna eşittir, $P^{-1} = P^T$. $A=P^TDP$ ve $P^{-1} = P^T$ eşitlikleri $Cov(Y)=(1/(n-1))*PAP^T$'de yerine konduğunda

$$\begin{aligned} Cov(Y) &= (1/(n-1))*PAP^T \\ &= (1/(n-1))*P(P^TDP)P^T \\ &= (1/(n-1))*P\,P^TDPP^T \\ &= (1/(n-1))*PP^{-1}DPP^{-1} \\ &= (1/(n-1))*D \end{aligned} \tag{31}$$

elde edilir. Buradan görülür ki, D köşegen olduğundan $Cov(Y)$' de köşegendir ve bu duruma P matrisinin seçimi ile ulaşılmıştır. Böylece dönüşüm matrisi P, XX^T matrisinin özvektörlerini satırlarında içeren matris olarak alındığında; veri kümesini içeren X matrisinin dönüştüğü Y matrisinin kovaryansı hedeflendiği gibi köşegen özelliğe sahip olur. Ayrıca, Y' nin kovaryans matrisindeki i. köşegen değer X' in p_i boyunca varyansıdır. $A=EDE^T$' den aşağıdaki şekilde elde edilen ve (özdeğerleri içeren) köşegen D matrisi, köşegen elemanlarında X' in varyansını taşır.

$$\begin{aligned} &A=EDE^T \\ &AE=EDE^TE \\ &AE=ED\ (matris*özvektörleri=özvektörleri*özdeğerleri) \\ &E^TAE=D \end{aligned} \tag{32}$$

Sonuç olarak; veri kümesi X'in dönüştürülmüş hali olan Y'nin içerisindeki ilişkileri gösteren kovaryans matrisinin köşegen özellikte olması X kümesinin P dönüşüm matrisi ile yeni bir uzaya taşınması ile sağlanmıştır. P' deki vektörler yeni uzayın eksenleridir. Bu uzayda aynı veriler yeni uzaya izdüşümleriyle temsil edilmektedir. Bu izdüşümleri Y

içermektedir ve veri kümesinin *Y* ile temsili sayesinde artık veriler arasındaki ilişkileri ölçmek daha verimli bir şekilde gerçekleştirilebilecektir. Verilerin aynı bileşenlerini kıyaslamakla verilerin verimli bir şekilde ayıredilmesi bu bileşenleri sağlayan yeni eksenlerle mümkün kılınmaktadır. Çünkü artık verilerin farklı bileşenleri arasındaki ilişkiler kaldırılmıştır.

Veri kümesini içerisindeki benzerlikleri ve farklılıkları irdeleyebilmek için en verimli şekilde ifade etmek, doğrultusu boyunca verinin dağılımını en çok kapsayan eksenleri yani en büyük varyansı veren özvektörlerin (dolayısıyla en büyük özdeğerli özvektörlerin) temsil ettiği eksenleri, *temel bileşenleri*, seçmekle başarılır.

Temel bileşenler tüm özvektörleri kapsamak zorunda değildir. İlgili temel bileşenin taşıdığı bilgi miktarı özdeğeri ile ölçülmekte olduğundan, özvektörler özdeğerlerine göre azalan sırada sıralanır ve sondaki bazı vektörler işleme dahil edilmeyebilir[46, 22]. Yeni uzaydaki boyut sayısını düşürmede fazla bilgi kaybetmeden, kullanılacak özdeğer sayısını belirlerken en büyük değerlilerin yanında küçük değerlilerin gözardı edilmesi aşağıdaki şartla sağlanabilmektedir.

$$\frac{\sum_{i=1}^{x} \lambda_i}{\sum_{i=1}^{a} \lambda_i} > 0{,}90 \tag{33}$$

λ_i, *Cov(X)*' in özdeğerlerini göstermektedir. *a* tane özdeğer içerisinden toplamlarının tüm özdeğerler toplamına oranı 0,90'dan büyük olan en büyük x tane özdeğer seçilecektir. *X* veri kümesi seçilen özdeğerlere karşılık gelen özvektörlerle temsil edilen uzaya taşınarak amaçlandığı üzere verimli bir şekilde ifade edilir.

Tüm özvektörler kullanılmadığında orjinal veri matrisini yeniden elde etmek için uygulanan ters dönüşüm işleminde oluşan hata aşağıdaki şekilde hesaplanabilir [47].

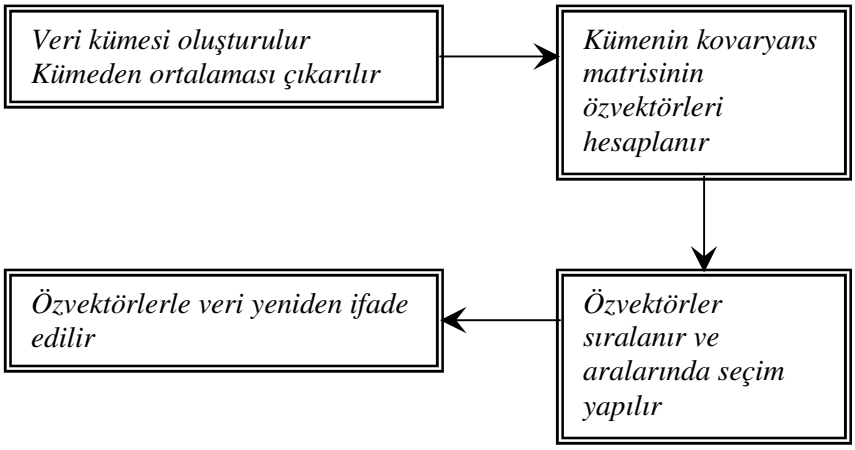
$$\sum_{i=1}^{a} \lambda_i - \sum_{i=1}^{x} \lambda_i = \sum_{i=x+1}^{a} \lambda_i \tag{34}$$

Ters dönüşüm işleminde, tüm özvektörler kullanıldığında ($x = a$) hata değeri *0*'dır. En büyük özdeğerlere karşılık gelen özvektörler seçildiğinde hata küçük değer alır. Bu bölümde kovaryans matrisinin hesaplanmasında kullanılan YY^T ifadesindeki Y matrisinin ortalama değerinin sıfır olduğu varsayılmıştır. Sonraki hesaplamalarda da görüleceği gibi kovaryans matrisi hesaplamasında veriyi ifade eden matristen veri ortalaması çıkarılmaktadır.

Burada ifade edilen yani TBA ile amaçlanan durum sayısal veriler üzerinde konunun daha iyi irdelenmesi için aşağıda örneklendirilmektedir.

1.8.3. TBA nın İki Boyutlu Bir Örnek Kümesi Üzerinde İncelenmesi

Şekil 16'da, TBA' da uygulanan adımlar ifade edilmektedir. Bu adımlara uygun olarak örnek bir veri kümesi üzerinde işlemler ayrıntılı olarak açıklanmaktadır.



Şekil 16. TBA' daki uygulama adımları

Öncelikle, kovaryansı alınacak veri matrisini oluşturmak için her veriden veri ortalaması çıkarılır. Aşağıda 2 boyutlu 10 veri ve çıkarma işleminin sonuçları görülmektedir. (Dikkate alınacak örnek verileri uygulama adımlarında elde edilen sonuçları karşılaştırmak amacıyla [48] nolu kaynaktan alınmıştır.)

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

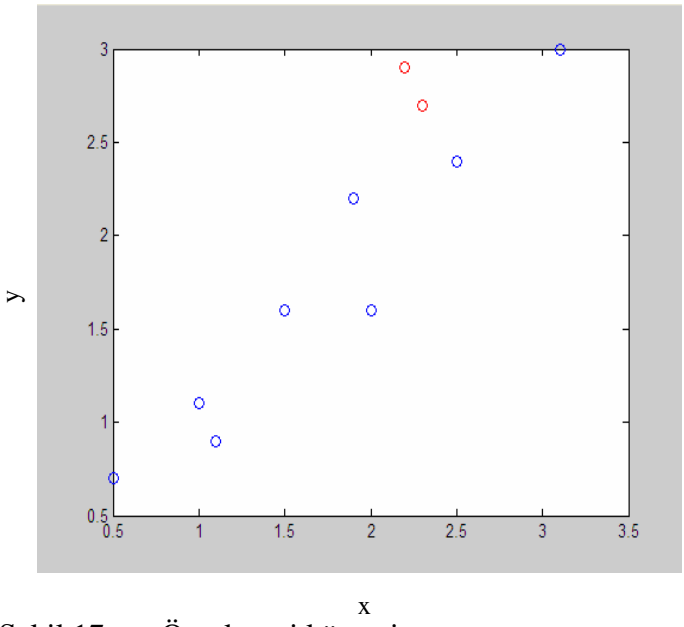
ort_x=1.81 ort_y=1.91

Veri matrisi X

x-ort_x	y-ort_y
0.69	0.49
-1.31	-1.21
0.39	0.99
0.09	0.29
1.29	1.09
0.49	0.79
0.19	-0.31
-0.81	-0.81
-0.31	-0.31
-0.71	-1.01

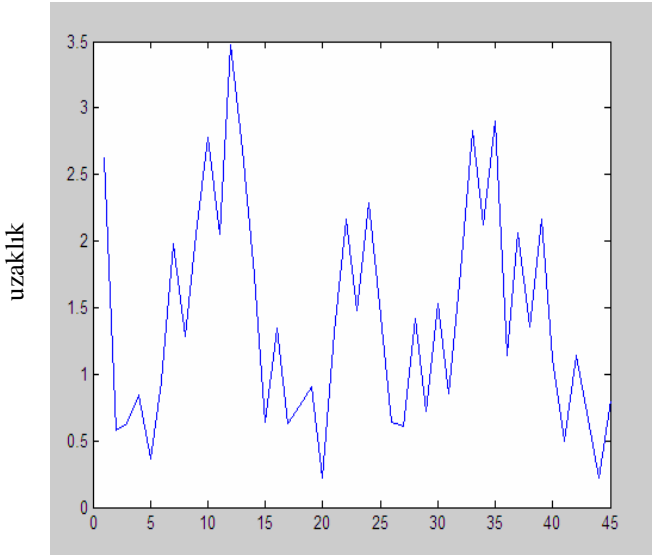
X - ort(X) = W

Aşağıda orjinal veri, eksenlere göre noktalar halinde işaretlenerek gösterilmektedir (Şekil 17).



Şekil 17. Örnek veri kümesi

Birbirine Öklit uzaklığı bakımından en yakın iki çiftten biri kırmızı ile işaretlenmiştir ve aralarındaki uzaklık 0.22361 olmaktadır. Veri çiftleri arasındaki uzaklıklar (*10*(10-1)/2 tane uzaklık*) aşağıda grafiksel olarak görülmektedir (Şekil 18).



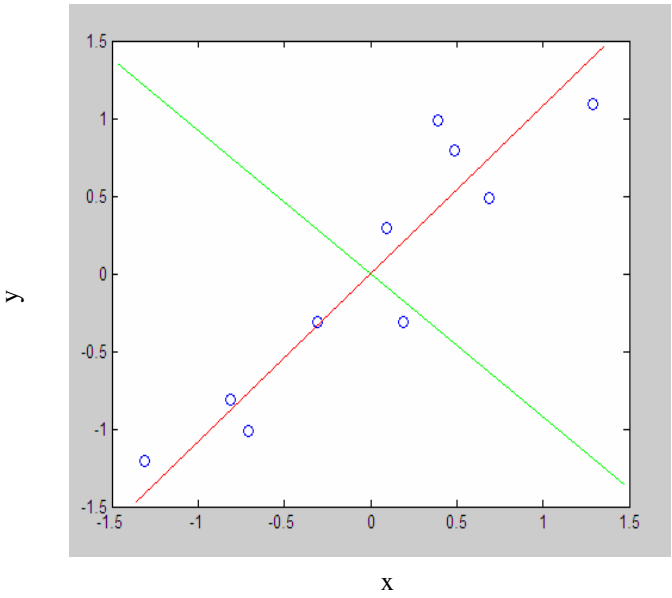
Şekil 18. Veri çiftleri arasındaki uzaklıklar

Bu veri matrisinin özvektörler uzayına izdüşümünü bulmak için $Cov(W) = W*W^T$ üzerinden bulunan kovaryans matrisi, özvektörleri ve özdeğerleri aşağıdadır.

$$Cov(W) = \begin{bmatrix} 0.6166 & 0.6154 \\ 0.6154 & 0.7166 \end{bmatrix} \tag{35}$$

$$Ozvektör(Cov(W)) = \begin{bmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{bmatrix} \quad Özdeğer(Cov(W)) = \begin{bmatrix} 0.0491 & 0 \\ 0 & 1.2840 \end{bmatrix}$$

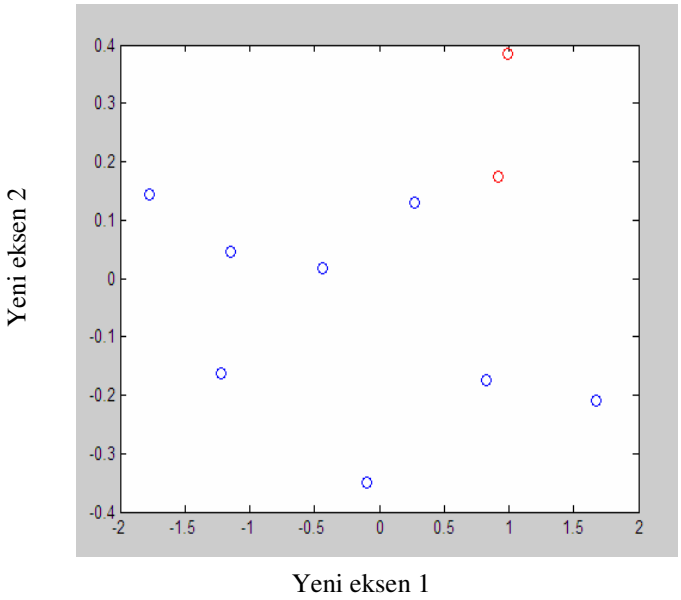
Kovaryans matrisindeki köşegen dışındaki elemanlar pozitif olduğundan x ve y bileşen değerlerinin birlikte artacağı görülmektedir, bu durum verilerin grafiksel gösteriminden de gözlemlenir. Herbir örneğin boyutu 2 olduğundan kovaryans matrisi *2*2* boyutlu olmuştur. Özvektörlerin boyu 1' dir ve birbirlerine diktir *(0.6779^2+0.7352^2=1, 0.6779*(-0.7352)+0.7352*0.6779=0).* Buradan görülür ki bu vektör matrisi dönüşüm matrisi olarak kullanılabilir. Özvektörler W matrisindeki veriler üzerinde Şekil 19'da çizilmiştir. (Şekil 19'da özvektörler çizilirken, özvektör üzerindeki iki noktanın orjinal eksenlerdeki koordinatları hesaplanıp bunlar arasına doğru çizilmiştir.)



Şekil 19. Hesaplanan özvektörler

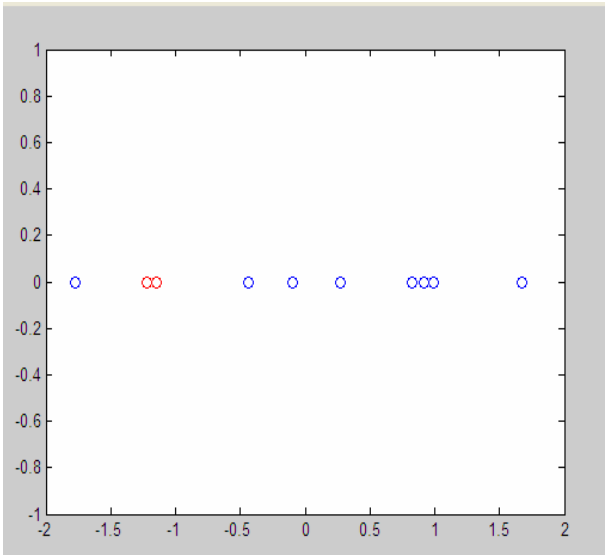
Kırmızı ile çizilen özvektör (p_1), veriyi en iyi kapsayan doğru olarak gözükmektedir ve bu veri kümesinin kendisi boyunca nasıl ilişkili olduğunu göstermektedir. Yeşille çizilen özvektör (p_2) de verideki diğer daha az önemli yapıyı yansıtmaktadır. Sonuç olarak, kovaryans matrisinin özvektörleri ile veriyi karakterize eden doğrular bulundu [48]. Sonraki aşama veriyi bu doğrulara göre ifade etmek için dönüştürmektir. En büyük özdeğerli özvektör veri kümesinin *temel bileşen*idir. Buradaki temel bileşen kırmızı olandır ve veri boyutları arasındaki en önemli ilişkidir. En önemli özvektörleri bulmak için özvektörler özdeğerlere göre sıralanmaktadır. Küçük özdeğerli özvektörler fazla bilgi kaybına neden olmadıklarından elenebilmektedir. Dönüşüm işleminde, seçilen özvektörler satırlara gelecek şekilde oluşturulan matris sağdan veri matrisi (W) ile çarpılmaktadır. Böylece orjinal veriler seçilen vektörler cinsinden ifade edilmiş olur.

Şekil 20 iki özvektör, Şekil 21 ise tek özvektör kullanılarak elde edilen dönüştürülmüş veriyi göstermektedir.



Veri koordinatları	
p_1	p_2
0.8280	*-0.1751*
-1.7776	*0.1429*
0.9922	*0.3844*
0.2742	*0.1304*
1.6758	*-0.2095*
0.9129	*0.1753*
-0.0991	*-0.3498*
-1.1446	*0.0464*
-0.4380	*0.0178*
-1.2238	*-0.1627*

Şekil 20. İki özvektör ile elde edilen dönüşüm



Veri koordinatları
p_1
0.8280
-1.7776
0.9922
0.2742
1.6758
0.9129
-0.0991
-1.1446
-0.4380
-1.2238

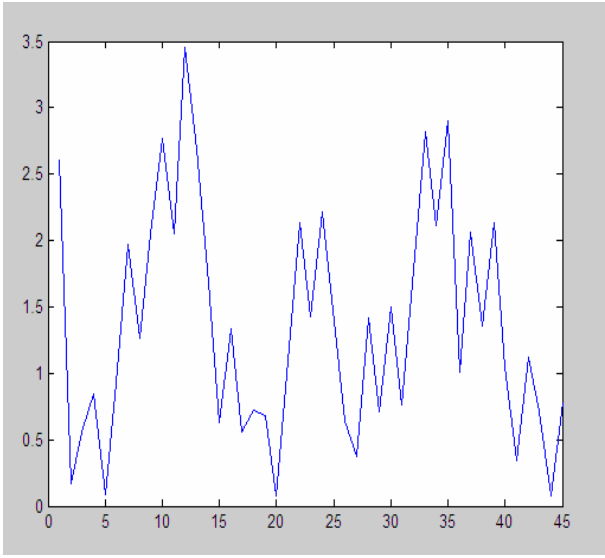
Şekil 21. Bir özvektör ile elde edilen dönüşüm

İki özvektör kullanılarak yapılan dönüşümde hiç veri kaybı olmadığından Şekil 20, Şekil 19' un özvektörler eksenlere gelecek şekilde döndürülmüş halidir. Şekil 21 *temel bileşen* kullanılarak elde edilen dönüşümdür ve dönüşüm sonucu veri tek boyuta inmektedir. Diğer dönüşümle kıyaslandığında burda elde edilen değerler diğer değerlerin ilk kolonunda içerilmektedir. Dolayısıyla, Şekil 21'deki noktalar Şekil 20'deki noktaların x konumlarıyla bir doğru üzerine düşerler [48].

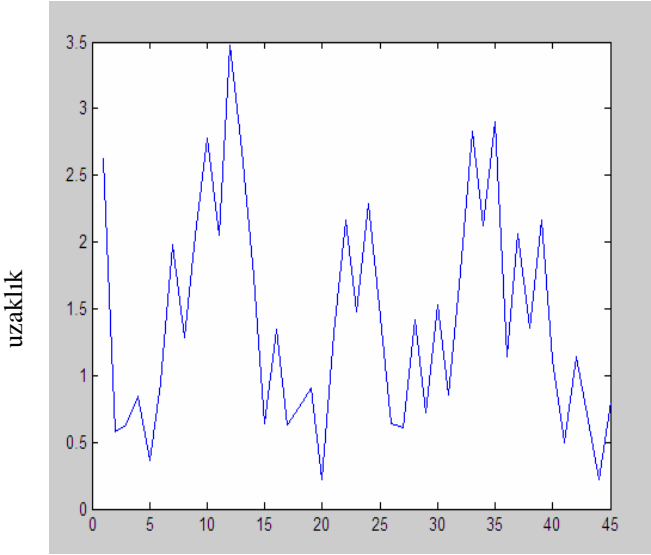
Aşağıda Şekil 22 ve Şekil 23'te sırayla tek özvektör ve 2 özvektör ile yapılan dönüşümlerden sonra veriler arasındaki uzaklıklar gösterilmektedir. 2 özvektörle yapılan dönüşümde uzaklıklar aynı kalmaktadır fakat tek özvektörle yapılan dönüşümde örneğin en yakın 2 nokta arasındaki uzaklık *0.22361*'den *0.079248*'e düşmüştür.

Sonuç olarak; veri kümesi içerisindeki ilişkileri en iyi açıklayan eksenlere göre ifade edildi. Bu haliyle verileri ifade eden noktaların birbirlerine göre durumları daha verimli bir şekilde kıyaslanabilmektedir.

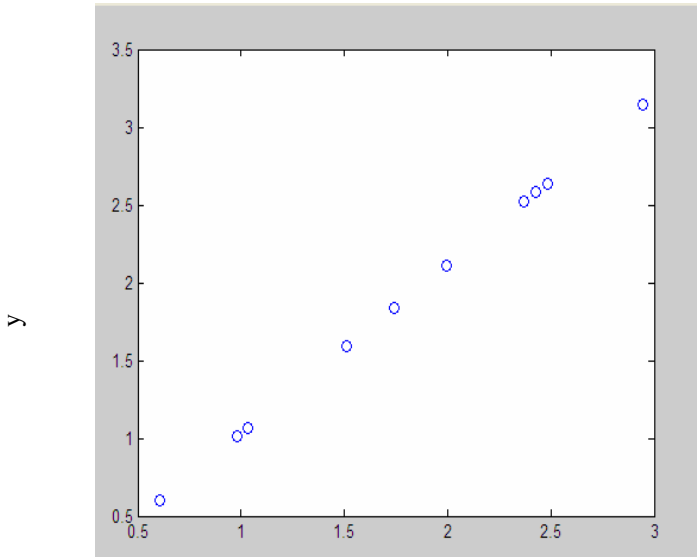
Dönüşüm işleminde tüm özvektörler kullanılırsa işlemlerden geri dönülerek orjinal veri hatasız olarak elde edilir. Dönüşümde özvektörlerin tamamı kullanılmadıysa orjinal veriye geri dönerken bir miktar bilgi kaybı olmaktadır. Orjinal veriye geri dönmek için dönüşüm sonucu sağdan dönüşüm matrisinin tersi ile çarpılır. Dönüşüm matrisi ortogonal özvektörlerden oluştuğu için tersi transpozuna eşittir. Böylece, dönüşüm sonucu elde edilen matrisin dönüşüm matrisinin transpozu ile çarpımına, işlemler başında veriden çıkarılan ortalama eklendiğinde orjinal veriye geri dönülür. İki özvektör kullanılarak yapılan dönüşümden geri dönüldüğünde orjinal veri tam olarak elde edildi. Tek özvektörle yapılan dönüşümden geri dönüldüğünde de aşağıdaki değerler elde edildi (Şekil 24). Şekil 24'te temel bileşen etrafındaki varyasyonun korunduğu görülmektedir.



Şekil 22. Tek özvektörle yapılan dönüşümden sonraki uzaklıklar



Şekil 23. İki özvektörle yapılan dönüşümden sonraki uzaklıklar



x	*y*
2.3713	*2.5187*
0.6050	*0.6032*
2.4826	*2.6394*
1.9959	*2.1116*
2.9460	*3.1420*
2.4289	*2.5812*
1.7428	*1.8371*
1.0341	1.0685
1.5131	1.5880
0.9804	1.0103

Şekil 24. Tek özvektörle yapılan dönüşümden geri dönüldüğünde elde edilen veri kümesi

2. YAPILAN ÇALIŞMALAR

2.1. Ön İşlemler

Bu bölümde, çalışmada uygulanan görüntü işleme adımları açıklanmaktadır.

2.1.1. Gri Seviyeye Çevirme

İşlenecek görüntüler gri seviyededir. Gri seviyeyi bulmak için ele alınan rengin üç bileşeninin (RGB) ağırlıklandırılmış toplamı hesaplanır. Kırmızı bileşenin yaklaşık %30' u, yeşilin %59'u ve mavinin %11'i alınarak bulunan, hedeflenen gri seviyedir [49]. Bu yüzdeler insan gözünün temel renklere olan farklı hassasiyeti göz önüne alınarak seçilmektedir. İnsan gözü kırmızı, yeşil ve maviyi farklı parlaklık seviyelerinde algılar[50]. Bir pikselin insan gözüyle algılanan parlaklığı 0.30*R+0.59*G+0.11*B şeklinde hesaplanır. Hesaplanan ağırlıklandırılmış değer, gri seviyeye dönüştürülecek resimde her bileşene (kırmızı, yeşil, mavi) atanır (Şekil 25). Aşağıda aynı resmin renkli hali ve gri seviyeye dönüştürülmiş hali görülmektedir (Şekil 26).

```
for(int i=0;i<boy;i++)
for(int j=0;j<en;j++)
{
y=piksel.r*0.30+piksel.g*0.59+piksel.b*0.11;
piksel[i][j]=TColor(RGB(y,y,y));
}
```

Şekil 25. Gri seviyeye dönüşüm

Şekil 26. Örnek renkli resim ve gri seviyeye dönüşümü

2.1.2. Sınır İzleme

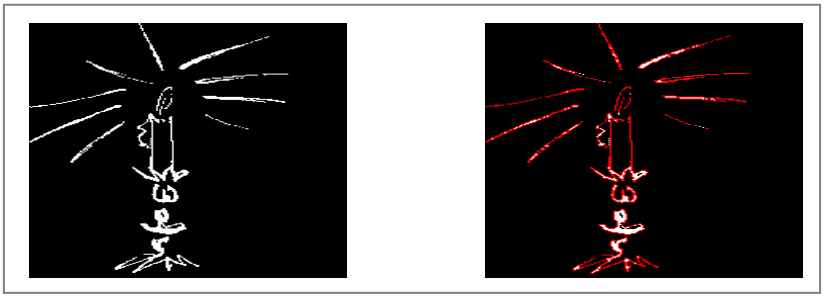
İkili resimdeki beyaz bölgelerin sınırlarının belirlenmesinde piksellerin Moore komşulukları dikkate alınmaktadır [51]. Bir pikselin Moore komşuluğu o pikselin etrafındaki 8 pikseli kapsar (Şekil 27.b). İlk beyaz pikselden başlanarak, bu pikselin etrafında sırayla 0,1,2,3,4,5,6,7 yönünde olmak üzere hareket edilir. Beyaz piksele rastlanıldığında, bu piksel sınır pikseli olarak kaydedilir ve bir önceki konuma geri dönülür. Geri gelinen konumdan, yeni bulunan pikselin etrafında ilerlenecektir. Bunun için yeni piksele göre konum hesaplanır ve bir sonraki konuma gidilir. Bu işlemi özetleyen algoritma ve örnek bir resime uygulanması ile elde edilen sınır aşağıda verilmiştir (Şekil 27.a ve 28).

```
B: sınır pikselleri dizisi
p: güncel sınır pikseli
c: p pikselinin incelenen komşusu

Beyaz pikseli, w, B ye koy
p=w
w ye gelmeden önceki konuma dön
c=p nin sıradaki komşusu
 while(c!=s)
   if(c==white)
     c yi B ye koy
     p=c
     c=p den önceki konum
  else
     c=p nin sıradaki komşusu
```

1	2	3
0	P	4
7	6	5

(a) (b)

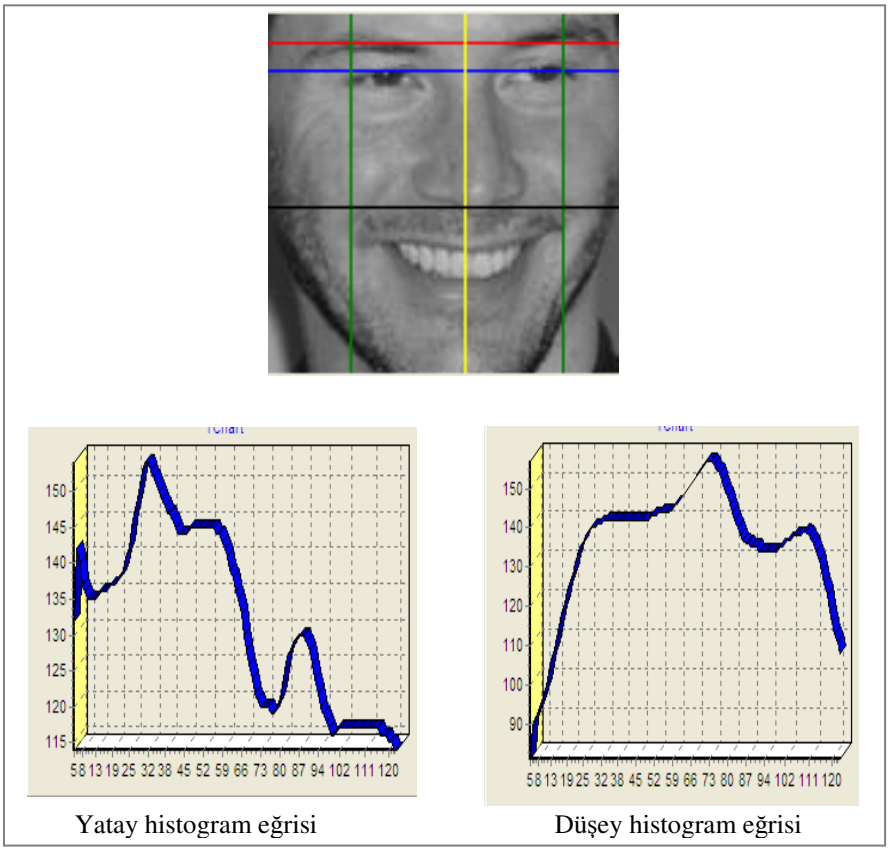
Şekil 27. (a) sınır izleme algoritması, (b) bir pikselin Moore komşuluğu



Şekil 28. Örnek resim ve sınırları (kırmızı)

2.1.3. Göz Konumu Belirleme

İncelenen görüntüde gözlerin yeri gri seviyedeki görüntünün yatay histogram eğrisinden yararlanılarak tespit edilmektedir [52]. Düzleştirilmiş yatay histogram eğrisindeki ilk en küçük değeri oluşturan satır kaşların (Şekil 29'daki kırmızı yatay çizgi), ikincisi gözlerin konumunu (Şekil 29'daki mavi yatay) vermektedir. Düzleştirilmiş düşey histogram eğrisindeki tepe noktası burun hizasını (sarı çizgi), solundaki minimum değer sol göz başlangıcını sağındaki de sağ göz başlangıcını (yeşil dikey çizgiler) vermektedir.



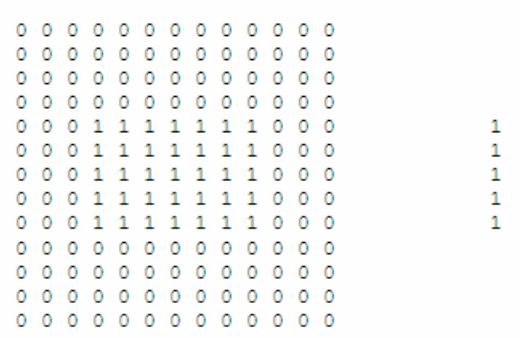
Şekil 29. Yatay ve düşey histogram eğrilerinden yararlanılarak göz seviyesinin bulunması

2.1.4. Aşındırma (Erosion)

Aşındırma işleminde 2 tür veri vardır. Birincisi işlemin uygulanacağı görüntü (A), ikincisi de (B) yapısal element (structuring element) adı verilen görüntü üzerinde aşındırma işleminin etkisini belirleyen veri kümesidir (Şekil 30). Burada ikili görüntü üzerindeki aşındırma işlemi incelenmektedir. Üzerinde işlem yapılacak görüntüdeki arka plan bilgisi 0, diğerleri 1' dir. Aşındırma

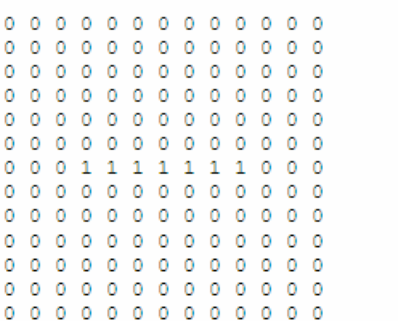
$$A \ominus B = \{z|(B)_z \subset A\} \tag{36}$$

şeklinde tanımlıdır [53]. A 13*13 boyutunda, B' de 5*1 boyutunda elemanları Şekil 30' daki gibi olan iki matris olsun.



Şekil 30. Aşındırma uygulanacak görüntü matrisi ve yapısal element

Aşındırma işlemi gerçekleştirilirken yapısal element görüntü üzerinde, yapısal elementin merkezi görüntü üzerinde 1 olan piksellerle örtüşecek şekilde gezdirilir. Eğer yapısal elementteki her piksel için, görüntüde o konuma karşılık gelen pikseller 1 ise, görüntüdeki merkeze karşılık gelen piksel olduğu gibi bırakılır; görüntüdeki piksellerden herhangibiri 0 ise, merkez piksel de 0 yapılarak eritilir [54]. Başka bir deyişle; B' nin merkezi A üzerinde gezdirilirken B, A' da tamamen kapsandığında B'nin merkezinin üzerinde olduğu A' daki piksel değiştirilmez, aksi durumda piksel silinir (Şekil 31).



Şekil 31. Aşındırma sonucu

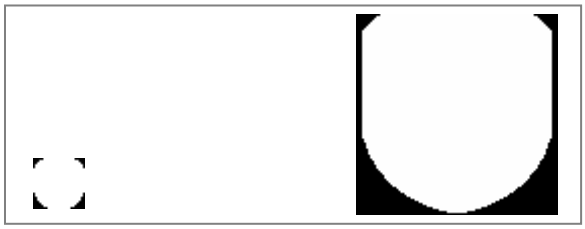
2.1.5. Görüntü Boyutu Değiştirme

Bir görüntünün boyutunu istenilen değerlere getirirken; hedef boyuttaki görüntünün herbir koordinatının orjinal resimde hangi koordinatları temsil edeceği belirlenerek o koordinatlardaki piksel değerleri ortalaması ile yeni görüntü oluşturulur. Örneğin, 100*100 boyutunda bir görüntünün 20*20' ye düşürülmesi durumunda; 20*20 boyutlu görüntünün herbir koordinatı için 100*100 boyutlu görüntüde 25 piksel değerinin ortalaması hesaplanır. Örneğin, 20*20 boyutlu görüntüdeki (10, 15) koordinatlı pikselin değeri için 100*100 boyutlu görüntüdeki (50, 75) ve (55, 80) koordinatları arasındaki piksel değerleri ortalaması hesaplanır. Aşağıda bu şekilde gerçekleştirilen örnekler görülmektedir (Şekil 32).

Şekil 32. Görüntü Boyutu Değiştirme Örnekleri

2.1.6. Oval Maske

Üzerinde işlem yapılacak yüz resminin bazı bölgeleri önemsenmemektedir. Özellikle köşelerdeki bölgelerde taşınan bilgi sonucu verimli yönde etkilemediğinden göz ardı edilmektedir. Çalışmada aşağıdaki iki maske kullanılmıştır (Şekil 33). Maskenin uygulanacağı resimde, maskelerdeki siyah piksellere karşılık gelen pikseller işleme dahil edilmeyecektir.



Şekil 33. Oval maskeler

Aşağıda bu maskelerin uygulandıkları görüntü üzerinde eledikleri kısımlar gözlenmektedir (Şekil 34).

Şekil 34. Maskenin resimlere uygulanışı

2.1.7. Parlaklık Düzenlemesi

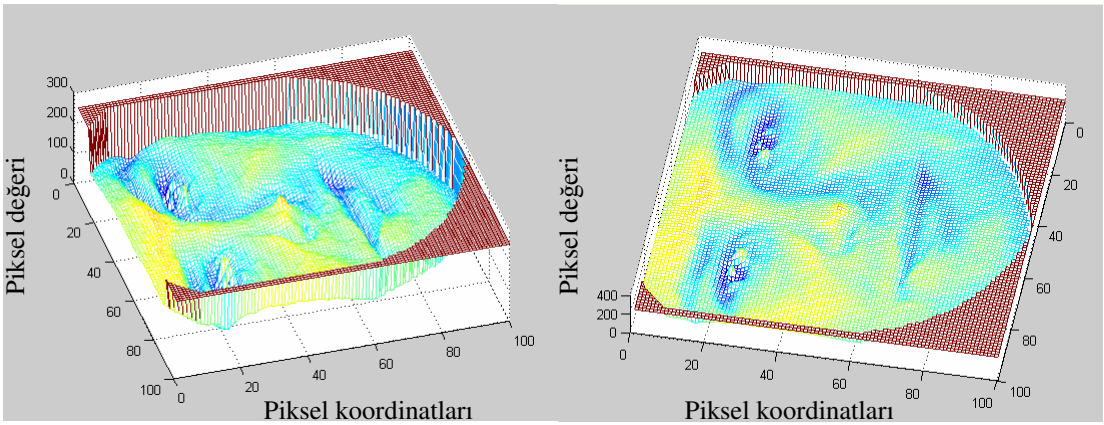
Veritabanında kullanılan resimlerin koyuluk ve açıklıklarının birbirlerinden çok farklı olmaması için resimlerin piksel değeri ortalamalarına göre her resimde bir miktar öteleme yapılmaktadır. En büyük ortalama degeri *m*, en küçük ortalama değeri *n* ise; *ek=(m+n)/2* değeri ötelemede kullanılacaktır. Her resmin pikselleri *ek-resim_ortalaması* kadar ötelenmektedir. Böylece her resmin ortalaması *ek* kadar olmaktadır. Sonuç olarak birlikte işleme koyulacak resimlerin ortalamaları eşitlenmektedir. Aşağıda örnek iki resmin bu işlemden önceki ve sonraki durumları ve örneklerden biri için de piksel değerlerinin grafiksel ifadesi gösterilmektedir (Şekil 35).

(1), ortalama=148

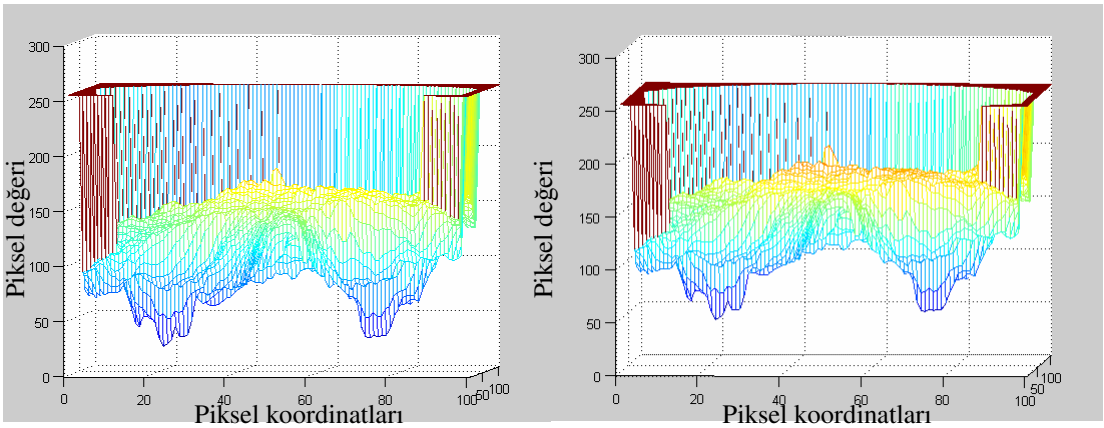
(2), ortalama=197

(1*), ortalama=167

(2*), ortalama=167



(1)' in pikselleri önden



(1) üstten

(1*) üstten

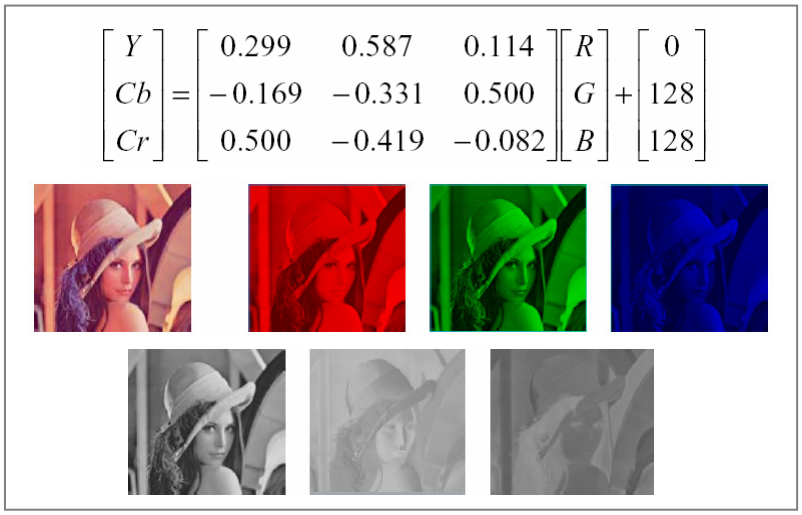
Şekil 35. Düzenlemeden önce ve sonra görüntüler

2.2. Ten İşleme

Yüz tespiti için resimlerde ten rengi belirleme çok kullanılan faydalı bir yöntemdir. Ten bölgesi tespiti işleminde, incelenen resme ait herbir piksel renk bileşenlerine dayalı olarak ten ya da ten değil şeklinde sınıflandırılmaktadır. Bu amaçla, ten renginin belirlenmesinde giriş resminin farklı bir renk uzayına taşınması söz konusudur. İncelenecek giriş resmi RGB (Red Green Blue) formatındadır. RGB bileşenleri aydınlatma koşullarına bağlı olduğundan, bu formattaki resimler üzerinde çalışan bir yöntem aydınlatma koşulları değiştiğinde başarısız olabilir. Ten olarak sınıflandırılacak pikselleri belirlemek için, incelenen resmin YCbCr bileşenleri hesaplanarak, renklilik bilgisini taşıyan Cb ve Cr bileşenleri incelenmektedir. Cb ve Cr bileşenlerinin değerlendirilmesi ile elde edilen ten rengine dayanarak piksellerin bu değere yakınlıklarına göre ten bölgesinde olup olmadıklarına karar verilmektedir.

2.2.1. YCbCr Renk Uzayı

YCbCr renk uzayında, parlaklık (luminance) bilgisi Y bileşeninde, renklilik (chrominance) bilgisi de Cb ve Cr bileşenlerinde tutulur [47, 55]. Cb bileşeni mavi bileşen ve referans değeri arasındaki farktır, Cr bileşeni de kırmızı bileşen ve referans değeri arasındaki farktır [47]. Y bileşeni aslında orjinal renkli resmin gri seviyeli kopyasıdır. Aşağıda (Şekil 36) RGB' den YCbCr' ye geçmek için kullanılan dönüşüm ve örnek bir resim üzerinde sırasıyla R, G, B, Y, Cb, Cr bileşenleri gösterilmektedir [56, 57].



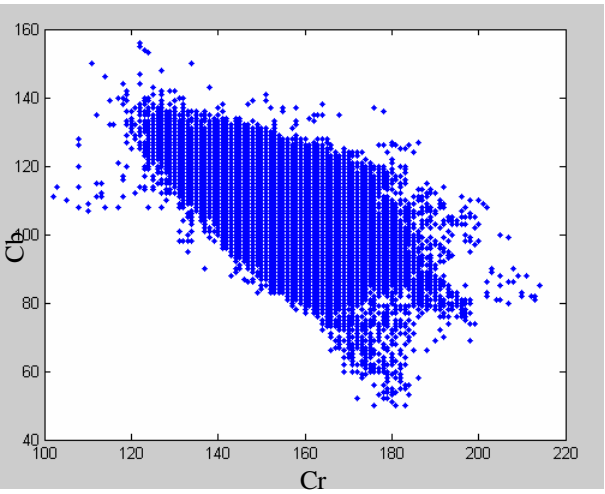
Şekil 36. Orjinal resim ve R, G, B, Y, Cb, Cr bileşenleri [56, 57].

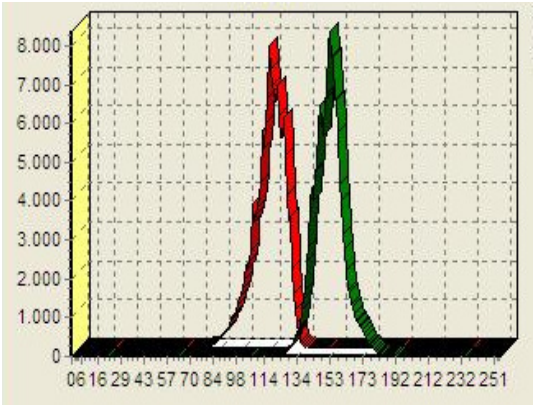
2.2.2. Ten Bölgesi Tespiti

Sisteme giriş olarak verilen resimdeki yüzlerin bulunması işleminde arka plan bilgisi üzerinde gereksiz işlem yapmaktan kaçınmak amacıyla ten işleme bölümü sisteme eklenmiştir. Resimdeki ten içeren bölgeleri belirleyen modül, oluşturulan 271 elemanlı (25*25 boyutlu ve oval masktan geçirilmiş) eğitim setinden (Şekil 37) üretilen ten bölgelerini temsil eden piksel değerleri üzerinde çalışarak, beyaz bölgeler ten bölgelerini gösterecek şekilde ikili resim üretir.

Şekil 37. Ten işlemede kullanılan eğitim setinden bazı örnekler

Herbir pikselin renk bileşenine bakılarak ten olup olmadığına karar verilmektedir. Eğitim setinden elde edilen Cb ve Cr bileşenlerinin standard sapması ve ortalamasına dayanarak pikseller değerlendirilir. Şekil 38'de eğitim setindeki örneklere ait Cb ve Cr bileşenleri görülmektedir. Hesaplanan ortalamadan standard sapma kadar farklı değere sahip renk bileşeni olan pikseller ten bölgesine ait olarak belirlenip sonuç resminde beyaz pikselle temsil edilmektedirler (Şekil 39).





Cb, Cr bileşenleri için histogram

(a) (b)

Şekil 38. a) Eğitim setinin Cb, Cr bileşenleri, b) Cb (kırmızı) ve Cr (yeşil) bileşenlerin sıklığı

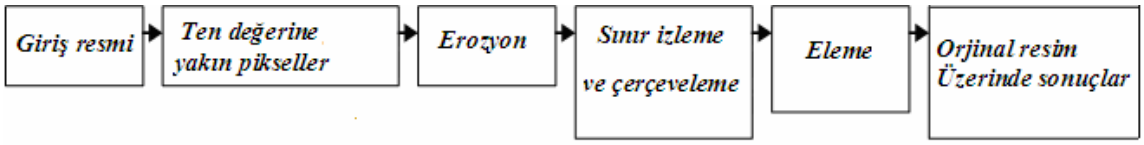
avr = eğitim setinden hesaplanan Cb,Cr ortalaması
n = eğitim seti eleman sayısı
std = hesaplanan standard sapma
x_i = pikselin Cb,Cr değeri

$$std = square\_root((1/n) * \Sigma(x_i - avr)^2)$$

```
if ((piksel.(Cb,Cr) <= avr + std) && ( (piksel.(Cb,Cr) >= avr - std)))
        piksel = white;
```

Şekil 39. Ten bölgesindeki piksellerin tespiti

Aşağıda ten bölgesinin tespitinde izlenen adımlar görülmektedir (Şekil 40).



Şekil 40. Ten işleme bölümünün genel ifadesi

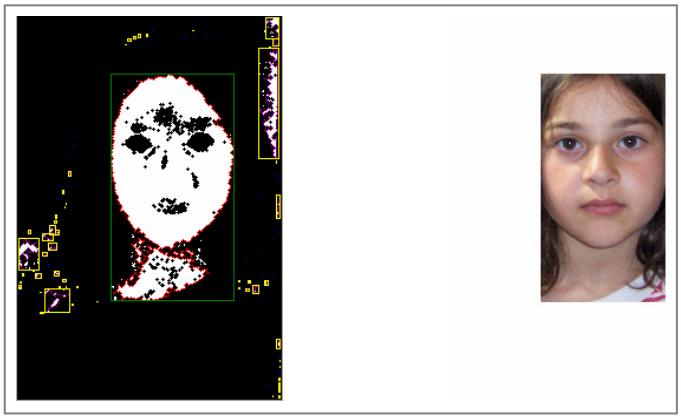
Eğitim setinden Cb bileşeni için standard sapma 10, Cr için 9; ortalama Cb için 113, Cr için 153 olarak hesaplanmıştır. Aşağıdaki resim bu değerler göz önüne alınarak ikili hale dönüştürülmüştür (Şekil 41).

(a) (b) (c)

Şekil 41. a) İşlenecek renkli resim b)Elde edilen ten bölgesi c) Aşındırma sonucu

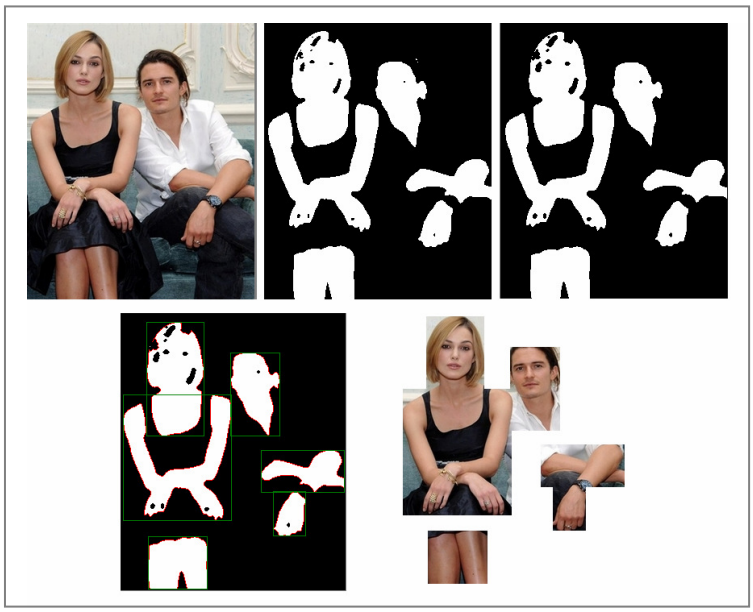
Sonuca bir kez aşındırma işlemi uygulanarak, olası gürültü değerleri yok edilir. Elde edilen beyaz bölgelerin sınırlarının belirlenmesinde piksellerin Moore komşulukları

dikkate alınmaktadır. Sınırları çizilen grupları kapsayan dörtgenler bulunarak bağımsız bölgelerin belirlenmesi işlemi gerçekleştirilir ve bölge alanı, en küçük yüzü kapsayacak 25*25 boyutlu kareden küçük olanlar elenir (Şekil 42).



Şekil 42. Sınırları belirlenmiş, eleme yapıldıktan sonra kalan ten bölgesi ve orjinal resimde sonucun gösterilişi (kırmızı pikseller bulunan ten bölgesinin sınırlarını göstermektedir, tek pikseller mavi ile gösterilmiştir, sarı dörtgenler elenen bölgeleri kapsamaktadır.)

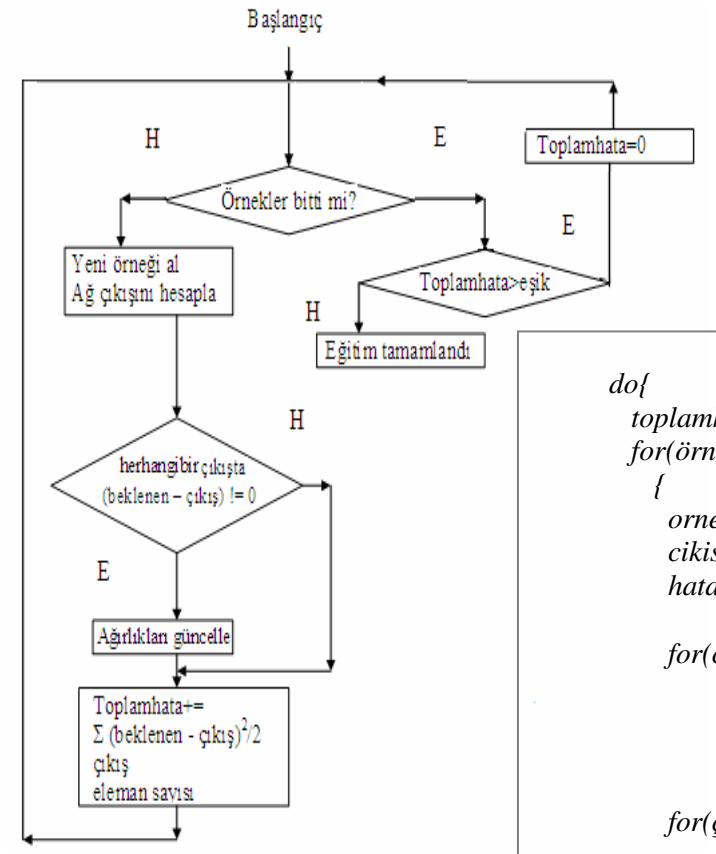
Aşağıda başka bir örnek resim üzerinde işlem aşamaları sırayla gösterilmektedir (Şekil 43).



Şekil 43.Örnek üzerinde ten işleme aşamaları

2.3. ÇKA' nın Gerçeklenmesi

Çalışmada çok katmanlı bir sinir ağı kullanılmıştır ve Şekil 44'te kullanılan ağın eğitim algoritması verilmektedir. Örnekler bitene kadar, ileri ve geri yönde hesaplama yapılmaktadır. Örnekler bittiğinde toplama hata değeri belirlenen eşikten büyükse toplam hata değeri sıfırlanarak örnekler güncel ağırlık değerleri ile yeniden eğitilmektedir. Örnekler için elde edilen ağ çıkışı beklenen değerden farklı olduğu sürece ağırlıklar güncellenmektedir. Örnekler tamamen ağa sunulduktan sonra toplam hata değeri eşik değerin altına düştüğünde eğitim işlemi sonlanır ve ağırlıklar kaydedilir. Eğitimin test edilmesi aşamasında elde edilen ağırlık değerleri ile test edilmek için ağa sunulan örnek kullanılarak ağ çıkışı hesaplanır. Eğitimde kullanılan fonksiyonlar Ek 3'dedir.




```
do{
  toplamhata=0.0
  for(örnek sayısı kadar)
    {
      ornekhatasi=0.0
      cikis_hesapla
      hata=false

      for(cikis sayısı kadar)
            hata=(hata||((beklenen-çıkış)!=0))

          if(hata) {ağırlık güncelle}

      for(çıkış sayısı kadar)
            ornekhatasi+=(beklenen-çıkış)^2

      toplamhata=toplamhata+ornekhatasi/2
      }
  }while(toplamhata>esik);
```

(a) (b)

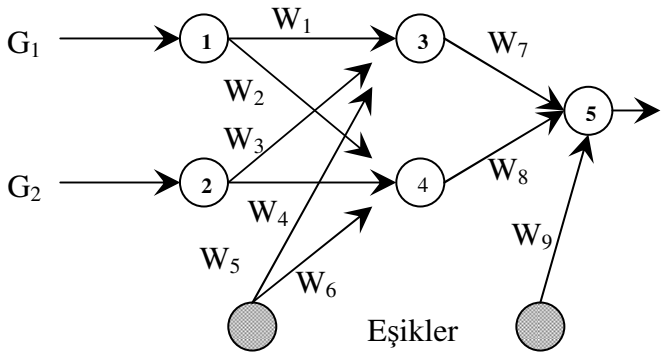
Şekil 44. (a) Eğitim algoritmasının akış diyagramı, (b) Eğitim algoritması

2.3.1. ÇKA ile XOR Problemi

Belirlenen eğitim algoritması (Şekil 44), XOR problemini temsil eden örnekler ve beklenen değerler üzerinde uygulanarak Tablo 3'deki sonuçlar elde edilmiştir. Öğrenme katsayısı 0.5, momentum katsayısı 0.8, aktivasyon fonksiyonu olarak sigmoid, eşik değeri olarak 0.001 alınmıştır. (Başlangıç değerleri de rasgele alınmıştır.) Ağın yapısında probleme uygun olarak girdi katmanında 2, ara katmanda 2 ve çıktı katmanında 1 proses elemanı kullanılmıştır (Şekil 45). Tablo 3' de görüldüğü gibi ağ problemi çok düşük hatalarla çözebilecek şekilde öğrenmiştir.

Tablo 3. XOR problemi için eğitim tamamlandıktan sonra elde edilen sonuçlar

Girdi 1	Girdi 2	Beklenen çıktı	Ağın çıktısı
0	0	0	0.01950539405660462
0	1	1	0.9791421309451811
1	0	1	0.9784779504624121
1	1	0	0.026781264180183943

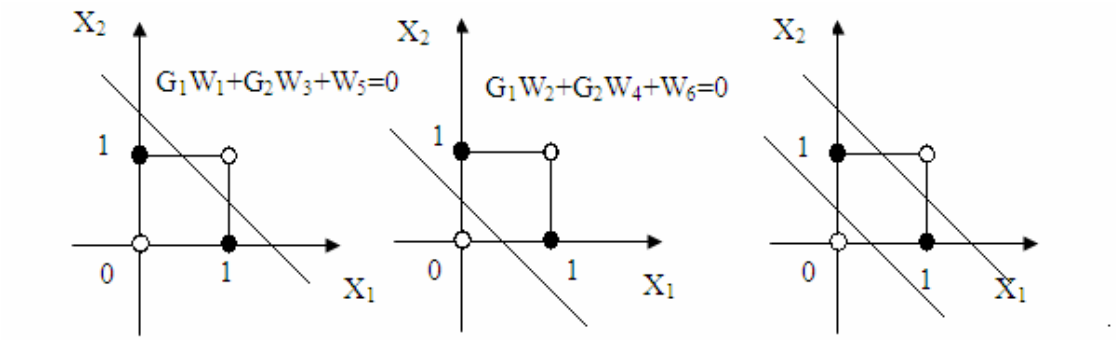


Ağırlıklar:

W_1=-4.480116570519896
W_2=-6.349470336097978
W_3=-4.563910564346504
W_4=-6.979251673679791
W_5=6.697861445905171
W_6=2.6312272354498103
W_7=9.374218839460294
W_8=-9.50673915787366
W_9=-4.411726581390527

Şekil 45. XOR problemi için kullanılan ÇKA

XOR işlemi için ÇKA' nın bulduğu karar sınırlarını kolay gözlemlemek amacıyla aktivasyon fonksiyonu olarak işaret fonksiyonu kullanılabilir. Eğitim sonucu elde edilen Şekil 45'teki ağırlık değerleri ve aktivasyon fonksiyonu olarak işaret fonksiyonu kullanıldığında ağ, (0,1,1,0) beklenen değerleri için (-1,1,1,-1) çıkışlarını üretti. Bu durum Şekil 46.a ve 46.b' de nöron 3 ve nöron 4' ün belirlediği sınır doğruları ile gösterilmektedir. Nöron 5 de bu doğruların lineer kombinasyonu olarak çalışmaktadır (Şekil 46.c).

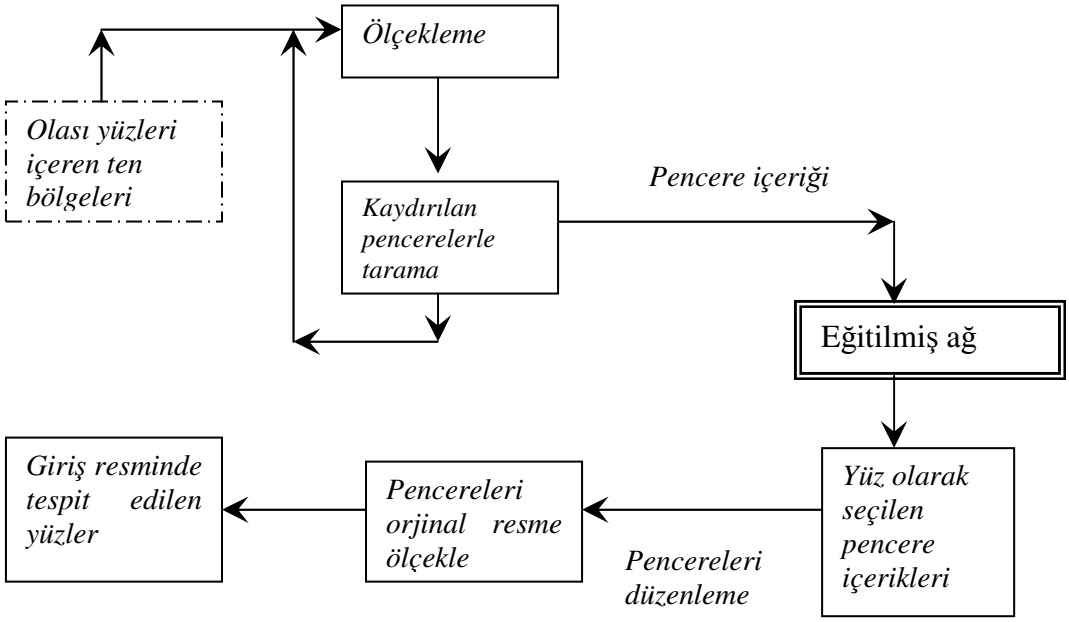


(a) (b) (c)

Şekil 46. (a) Nöron 3 ile üretilen sınır doğrusu, (b) Nöron 5 ile üretilen sınır doğrusu, (c) Tüm ağ tarafından üretilen sınır doğrusu

2.4. Yüz Tespiti

Ten işleme bölümünden elde edilen sonuç YSA'ya dayalı sorgulama kısmında işlenmektedir (Şekil 47).



Şekil 47. YSA' ya dayalı sorgulama

2.4.1. Ağın Eğitimi

Bu aşama için öncelikle ağın eğitimi gerçekleştirilmektedir. Ağ, yüz ve yüz olmayan resimler arasında seçim yapabilmesi amacıyla eğitilmektedir. Yüz ve yüz olmayan örneklerden oluşan veritabanına sırayla gri görüntüye çevirme, parlaklık düzenlemesi ve oval masktan geçirme işlemleri uygulanmaktadır. Eğitim setinde kullanılan ve bu işlemlerden geçirilmiş 25x25 boyutundaki gri seviyeli görüntülerden bazıları Şekil 48'de görülmektedir. Eğitim seti görüntüleri, ağa sunulmadan önce piksel değerleri belirlenen referans değerine göre ötelenerek tüm görüntüler aynı derecede parlak hale getirilmektedir.

Görüntü ağa sunulmadan önce oval masktan geçirilerek köşelerdeki yüzün tespitinde etkisi olmayan bilgiler elenebilmektedir. Maskenin siyah köşe değerlerine karşılık gelen, görüntüdeki piksel değerleri ağa sunulmayarak hem yüz bilgisi ön plana çıkarılmış olur hem de ağın giriş sayısı azaltılmış olur. Kullanılan maske ile 25x25 olan ağ giriş sayısı uygulamada 551 olarak alınmıştır. Yüz belirleme işlemi için veritabanının eğitminde, yüz olarak belirlenen örneklerin iyi ayırd edilmesini sağlamak amacıyla yüz olmayan örneklerin seçimi önem taşımaktadır. Eğitim seti için seçilen örnekler, ağın verimli bir şekilde eğitilmesi için işlendikten sonra, rastgele alınarak sisteme verilmektedir. Eğitim işlemi ağırlıkların güncellenmesi olarak özetlenmektedir. Örneklerden gelen bilgi, ağırlıklara aktarıldığından, ağırlıklar önceki öğrenmeleri yok etmeyecek şekilde güncellenmelidir. "Öğrenme oranı" olarak isimlendirilen sabit, ağırlıkların değişim miktarını kontrol etmek için kullanılır. Bu sabit çok küçük seçilirse öğrenme işlemi çok uzun zaman alır; çok büyük seçilirse de eski öğrenilenlerin kaybolmasına neden olabilir. Öğrenme katsayısının optimum değeri genellikle denenerek belirlenir.

Şekil 48. a) veritabanındaki yüz grubundan birkaç örnek b) veritabanındaki yüz olmayanlar grubundan birkaç örnek

Eğitim aşamasında, yüz tespiti için oluşturulan veritabanındaki 280 yüz, 320 yüz olmayan örnek ağa sunuldu, sırasıyla (1,0) değerleri beklenen çıkışlar olarak belirlendi ve 0.001 tolerans değeriyle eğitim tamamlandı. Kullanılan arayüz Ek 4'dedir.

Aşağıda eğitilmiş ağın yanlış olarak tespit ettiği bazı kısımlar görülmektedir (Şekil 49), bunlar önişlemden geçirilip ağa yüz olmayan örnekler olarak verilerek tekrar eğitildiğinde bu gibi yanlış tespitler elenmektedir.

Şekil 49. Belirlenen kötü örneklerden birkaçı

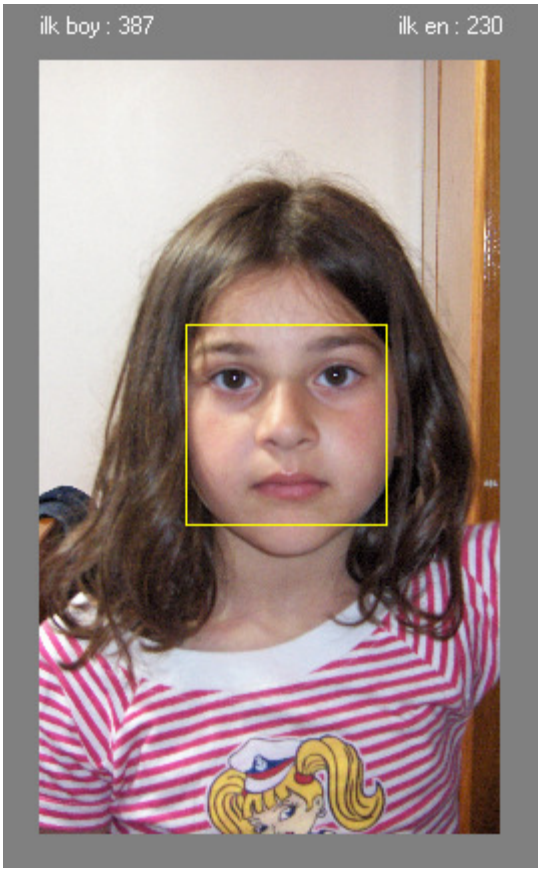
2.4.2. YSA ile Yüz Tespiti

Yüz tespiti işleminde, ağın eğitiminden sonra ikinci aşama olarak, olası tüm yüzlerin bulunabilmesi amacıyla görüntünün çeşitli boyutlara küçültülerek taranması gerçekleştirilmektedir (Şekil 50). Tespit edilecek yüzleri içeren görüntü gri seviyeye çevrildikten sonra, eğitim setindeki görüntülerin boyutuna (25x25) adım adım, görüntünün en boy oranı korunacak şekilde ölçeklenmektedir. Her adımda görüntü içerisinde 25x25 boyutunda pencere gezdirilerek pencere içeriği eğitilmiş ağa sunulmaktadır. Hesaplanan ağ çıkışı belirli bir eşik değerinden büyük ise pencere o anki konumuyla görüntüde bir yüzü kapsamaktadır ve köşe koordinatları gerçek görüntü boyutuna ölçeklenerek kaydedilir. Şekil 51'de görüntüde ölçekleme ve pencere gezdirme yapılarak yüz tespiti örneklendirilmiştir.

```
for(seçilen küçültme adımı sayısı kadar){
    For(tüm resimde pencere kaydırma adımı ile 25*25 lik pencere gezdir){
        Pencerenin içeriğine oval mask uygula
        YSA ya gönder
        Output>=threshold ise sonuca ekle
    }
    Yakın pencereleri belirle ve yerlerine ortalama pencereyi al
    Küçült
}
pencereleri küçültme oranlarına göre orjinal resime ölçekle.
```

Şekil 50. Yüz arama algoritması

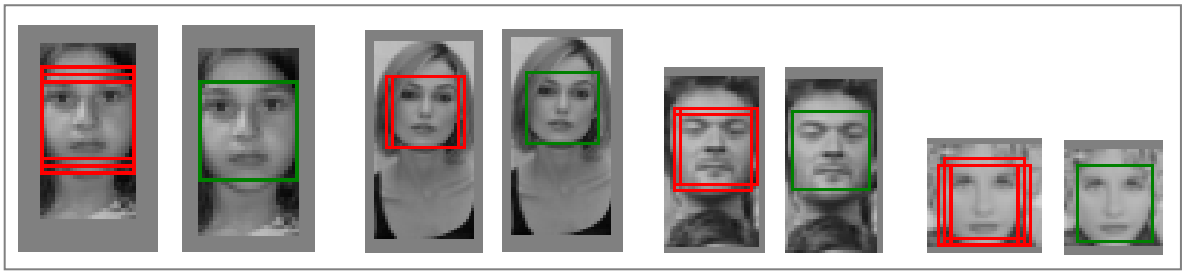
(a)



(b)

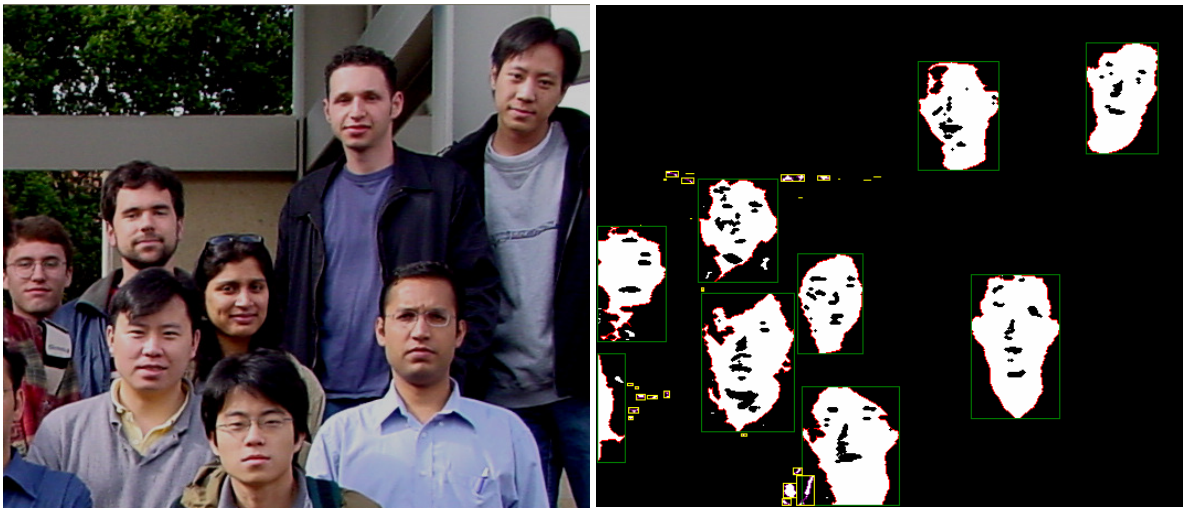
Şekil 51. Ölçekleme ve pencere gezdirme ile yüz tespiti

Pencere kaydırırken, bir yüzü aynı oranda kapsayarak ağın eşikten büyük çıkış üretmesine neden olacak pencere içerikleri meydana gelebilmektedir. Dolayısıyla, aynı yüzü kapsayan kaymış pencereler sonuç olarak ortaya çıkabilmektedirler. Bu durumda pencereler aynı gruba alınacak kadar birbirlerine yakın olduklarından, köşe koordinatlarının ortalaması alınarak tek pencereye indirgenirler. Pencereler arasındaki uzaklıklar sol üst köşe koordinat noktaları arasındaki uzaklık olarak alınmaktadır. Şekil 52' de bu durum örneklendirilmiştir.



Şekil 52. Kaymış kareler ve ortalamaları

Aşağıda ten bölgelerinin tespiti yapılan giriş resmi üzerinde yukarıda açıklandığı gibi eğitilmiş ağ kullanılarak yapılan tarama sonucu elde edilen yüzler gösterilmektedir (Şekil 53, 54 ve 55). Şekil 53'teki giriş resminde bulunan ten bölgeleri Şekil 54'te, bu bölgeler üzerinde tespit edilen yüz bölgeleri de Şekil 55'te verilmiştir. Ayrıca Şekil 56'da da bazı görüntüler üzerindeki yüz tespiti sonuçları görülmektedir.



(a) (b)

Şekil 53. (a) Giriş resmi ve (b) ten bölgesi sınırları

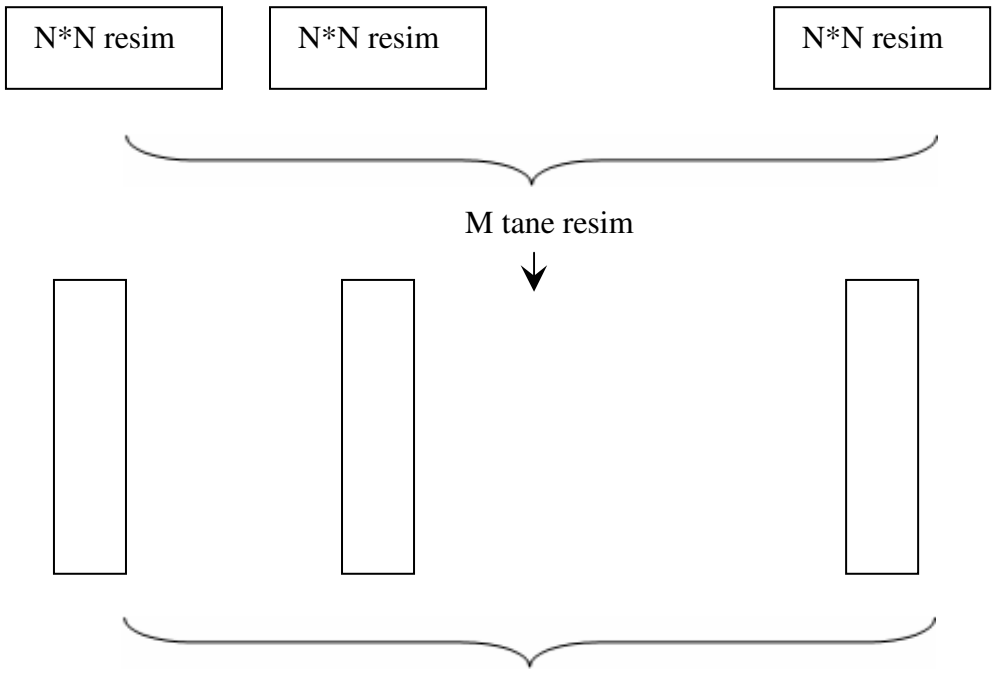
Şekil 54. Tespit edilen ten bölgeleri

Şekil 55. Tespit edilen yüzler

Şekil 56. Yüz tespiti örnekleri

2.5. TBA' nın Uygulanışı

TBA' daki algoritma bu bölümde açıklanmaktadır. Örnekleri sütunlarında içeren *X* veri matrisi, resimlerden oluşan örnekler için şu şekilde oluşturulmaktadır: 2 boyutlu yüz resimleri satırları ard arda eklenerek 1 boyutlu vektörler haline getirilir ve *X* veri matrisinin sütunlarına yerleştirilir (Şekil 57).



Şekil 57. Veri matrisinin oluşturulması

Aşağıda da görüldüğü gibi *N*N* boyutlu *M* tane resim için veri matrisinin boyutu N^2*M olmaktadır.

$$X = [\, x_1\, x_2 \ldots x_M \,]$$
$$x_i = [\, d_1\, d_2 \ldots d_N{}^2 \,]^T,\ i=1,2,\ldots,\ M \tag{37}$$

Kovaryans matrisinin hesaplanması için örneklerin ortalaması herbir örnekten çıkarılır ve işlemler bu veri matrisinden veri ortalaması çıkarılarak elde edilen (N^2*M boyutlu) matris, $Q(=[q_1\ q_{2\ldots}\ q_M])$, üzerinden devam ettirilir.

$$ort = \frac{1}{M} * \sum_{i=1}^{M} x_i\ ,\quad q_i = x_i - ort \tag{38}$$

Q matrisi ile ifade edilen veriler matrisin kovaryansının özvektörleri ile tanımlanan uzaya taşınacaktır. $Cov(Q) = QQ^T$ matrisinin boyutu N^2*N^2 olacağından ve özvektör sayısı da en fazla N^2 olacağından bu matrisin özvektörlerinin (u_i) direk olarak hesaplanması zor bir işlemdir.

$$Cov(Q) = \frac{1}{M} * \sum_{i=1}^{M} q_i q_i^T \quad = \quad QQ^T \quad (N^2 * N^2 \ matris) \tag{39}$$

Bu yüzden, bu özvektörler aşağıdaki şekilde Q^TQ matrisinin özvektörlerinden elde edilmektedir [22]. Q^TQ' nun boyutu $M*M$'dir ve en fazla M bileşenli M tane özvektöre sahip olabilir.

$$Q^TQ\ v_i = \mu_i v_i \quad (v_i, Q^TQ\text{' nin özvektörleridir ve } \mu_i, \text{ özdeğerleridir.}) \tag{40}$$

Her iki tarafı Q ile çarparak $Q\ Q^TQ\ v_i = \mu_i Qv_i$ elde edilir ve buradan QQ^T' nin özvektörlerinin $u_i = Qv_i$ (Qv_i , $\| Qv_i \| = 1$ olacak şekilde normalize edilmelidir), özdeğerlerinin de μ_i olduğu görülür (matris*özvektörleri = özdeğerleri*özvektörleri). Bu özvektörlerden sıfırdan farklı özdeğerlere karşılık gelenleri yeni uzay için ortonormal bir taban oluşturmaktadır. Özvektörler ilgili özdeğerlere göre büyükten küçüğe sıralanır. Yüz resmi bu özvektörlerden $x<<M<<N^2$ tanesinin eksen olduğu uzaya aşağıdaki izdüşüm işlemi ile taşınır. (u_i ' ler sütun vektörleri olduklarından transpoz işlemi uygulanıyor. Herbir u_i ' nin N^2 bileşeni vardır.)

$$Y_j=[y_1\ y_2\ \ldots\ y_x]^T, \qquad y_i = u_i^T q_j \qquad i=1, \ldots, x \qquad j=1, \ldots, M$$
$$Y = [Y_1, Y_2, \ldots Y_M] \tag{41}$$

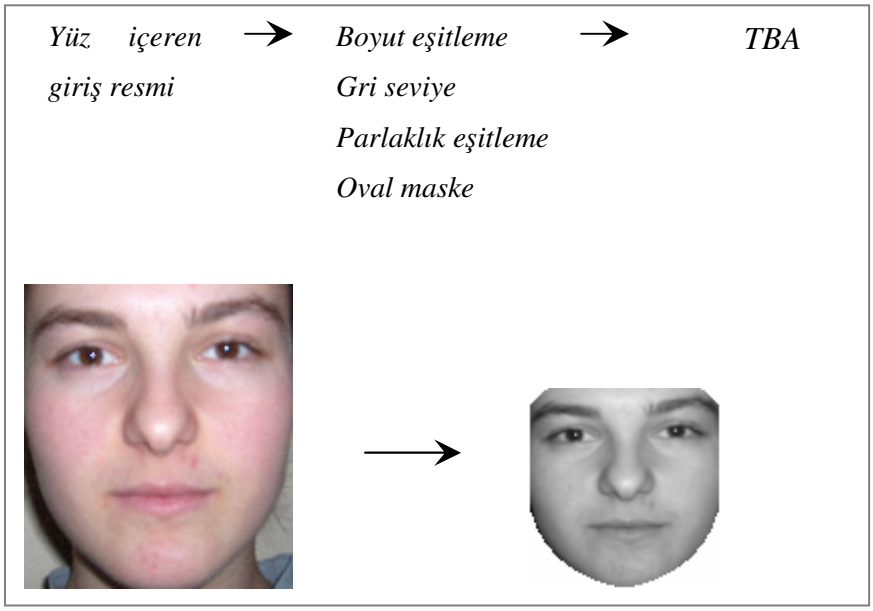
y_i , Y_j yüz resminin yeni uzaydaki i. koordinatıdır. Burada, özvektörler de, u_i , görüntü olarak ifade edilmektedir ve *özyüz* olarak adlandırılmaktadırlar [22]. Sonuç olarak; $x*M$ boyutlu Y matrisinin sütun elemanı Y_j, yüz resminin yeni uzayda temsil edilmesinde herbir özyüzün (yeni eksenlerin) katkısını içermektedir. Dolayısıyla, hedeflendiği gibi yüzler yeni uzaydaki koordinatlarıyla eksenlere göre aşağıdaki şekilde ifade edilmektedirler. Başka bir deyişle, veritabanındaki herbir yüz q_i (ortalama çıkarıldıktan sonra), seçilen özvektörlerin lineer kombinasyonu ile temsil edilmektedir.

$$q_j = y_1u_1 + y_2u_2 + \ldots + y_xu_x = \sum_{k=1}^{x} y_k u_k \tag{42}$$

Özyüzler uzayına izdüşümü yapılan resimler artık Y_j ile temsil edilmektedirler.

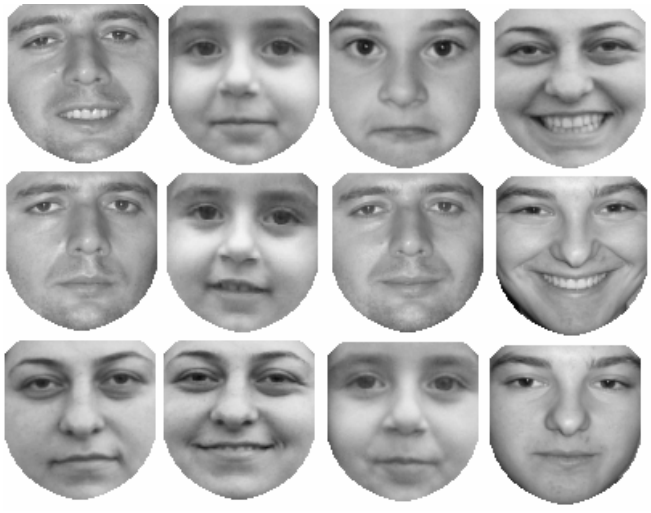
2.5.1. TBA' nın Gerçeklenmesi

TBA analizi, orjinal resimleri aralarındaki benzerlikler ve farklılıklar açısından yorumlama imkanı sunmaktadır. Bu amaçla, TBA ile işlenecek resim veritabanındaki tüm resimlere aşağıdaki ön işlemler sırası ile uygulanarak resimler TBA için hazır hale getirilmektedir (Şekil 58).



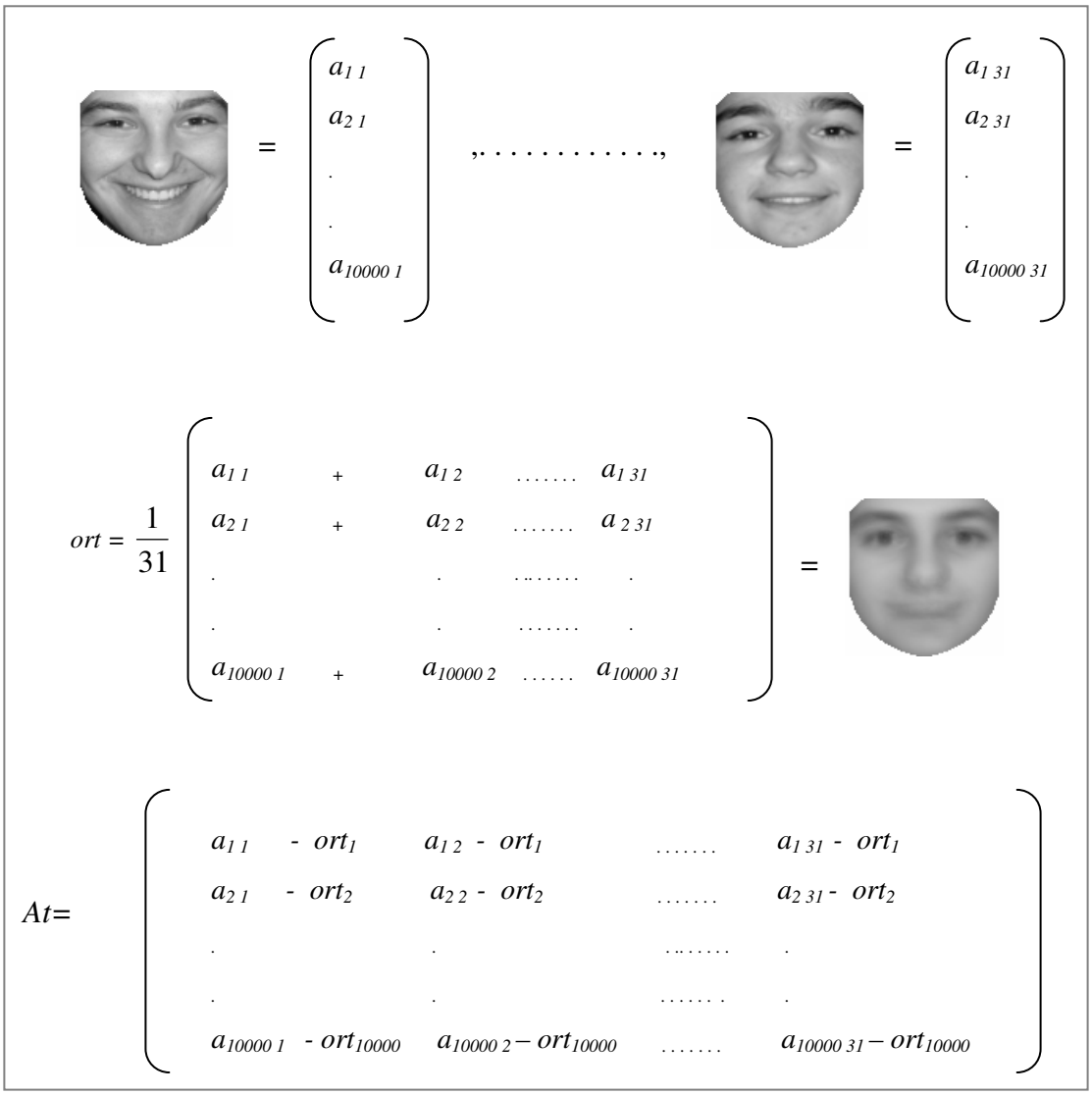
Şekil 58. Önişlem

Aşağıda bu işlemlerden geçirilerek 31 elemanlı veritabanına eklenen resimlerden bazıları görülmektedir (Şekil 59). Veritabanında her kişiye ait en az 2 en çok 4 resim bulunmaktadır. Veritabanının tamamı Ek 5'tedir.



Şekil 59. Veritabanının bir kısmı

Ön işlemden geçirilen resimler TBA'den geçirilmek üzere herbir yüz resminin piksel değerleri ayrı bir sütuna gelecek şekilde bir matriste (program kodunda bu matris *At*' dir.) toplanmaktadır. Resimlerin piksel değerleri matrise yerleştirilirken herbir resimden tüm resimler toplanarak elde edilen ortalama yüz resmi çıkarılmaktadır (Şekil 60).



Şekil 60. Veri matrisinin oluşturulması

Resim boyutları *100*100* olduğundan her resim bu aşamada aslında *10000* boyutlu uzayda bir nokta olarak temsil edilmektedirler. Yani her bir yüz için herbiri tek bir piksel değerine karşılık gelen *10000* ölçüm değeri mevcuttur. TBA kullanılarak bu noktaların (resimlerin) daha az boyutlu uzayda ifade şekilleri bulunacaktır ve yeni boyut sayısı *10000* den çok küçük bir değer olacaktır. Bu aşamadan sonra yüz resimlerinin yeni ifade

şekillerine göre birbirlerine uzaklıkları bulunacaktır. Bu amaçla; öncelikle matrisin (*At*) kovaryansının özvektörlerini bulmak için yukarıda anlatılan şekilde At^T*At' den yararlanılmaktadır (Şekil 61). (*At* nin boyutu *10000*31* yani bir resim boyutu*örnek sayısıdır.)

```
B= (( At ' * At ) / (num_1-1));  % (At '), At nin transpozudur.
[C D] = eig(B);
```

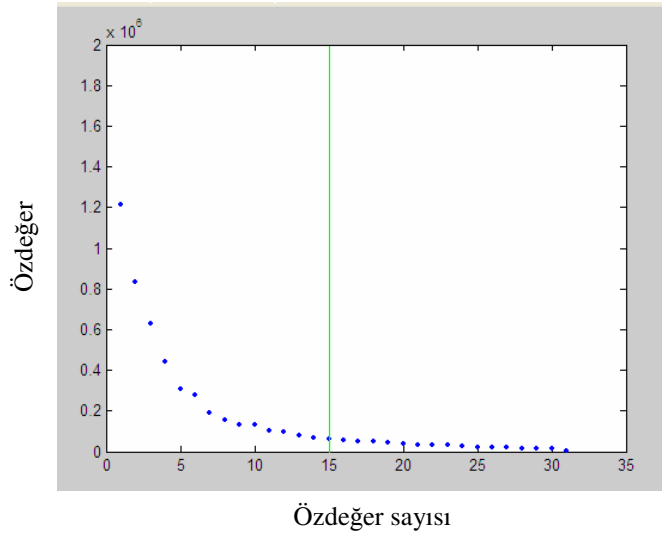
Şekil 61. Özvektörlerin elde edilişi

Burada *C* özvektörleri *D* de özdeğerleri taşır. *C*, matrisi özvektörleri sütünlarında taşır ve boyutu *31*31* (örnek sayısı* örnek sayısı)' dir. Özvektörlerden aşağıdaki (Şekil 62) hesaplamayla belirlenen en büyük varyanslara karşılık gelen *15* tanesi (bu değer *num_3* değişkeninde tutuluyor) buradaki dönüşüm işleminde kullanılmaktadır.

```
[F S]=BigToSmal(diag(D)');  % özdeğerler sıralanır
s1=sum(diag(D));            % diyagonaldeki elemanlar toplanır
s2=0;
for i=1:num_1
   s2=s2+D(S(i),S(i))/s1;
   if s2>=0.90
      D(S(i),S(i))
      num_3=i; % toplamlarının tüm özdeğer toplamlarına oranı %90 ı geçenlerin sayısı
      break;    end      end   % özvektör sayısı olarak alınır
```

Şekil 62. Özvektör sayısının elde edilişi

Tüm özdeğerleri gösteren Şekil 63 incelendiğinde gerçekten de sıralı özdeğerlerden ilk *15* tanesinin tamamının *%90*'ı olduğu gözlenir.



Şekil 63. Özdeğerler

Kullanılacak özvektör sayısı belirlendikten sonra özvektörler karşılık gelen özdeğerlere göre sıralanıp, At^T matrisinin kovaryansının özvektörlerine geçilir (Şekil 64):

```
for i=1:num_3    % döngüyle At^T*At nın özvektörleri özdeğerlere göre sıralanır
   G(:,i)=C(:,S(i));     end
Ozv=At*G;              % At*At^T nin özvektörleri hesaplanır
```

Şekil 64. Dönüşümde kullanılacak özvektörlerin hesaplanması

Özvektörler boyları 1 olacak şekilde normalize edilir. Vektör boyu bileşenlerinin karelerinin toplamının karekökü ile hesaplandığından her bileşen vektör boyuna bölünerek vektör boyu 1 yapılır (Şekil 65).

```
for a=1: num_3
T(a)=sqrt(sum(Ozv(:,a).*Ozv(:,a)));
Ozv(:,a)=Ozv(:,a)/T(a);      end
```

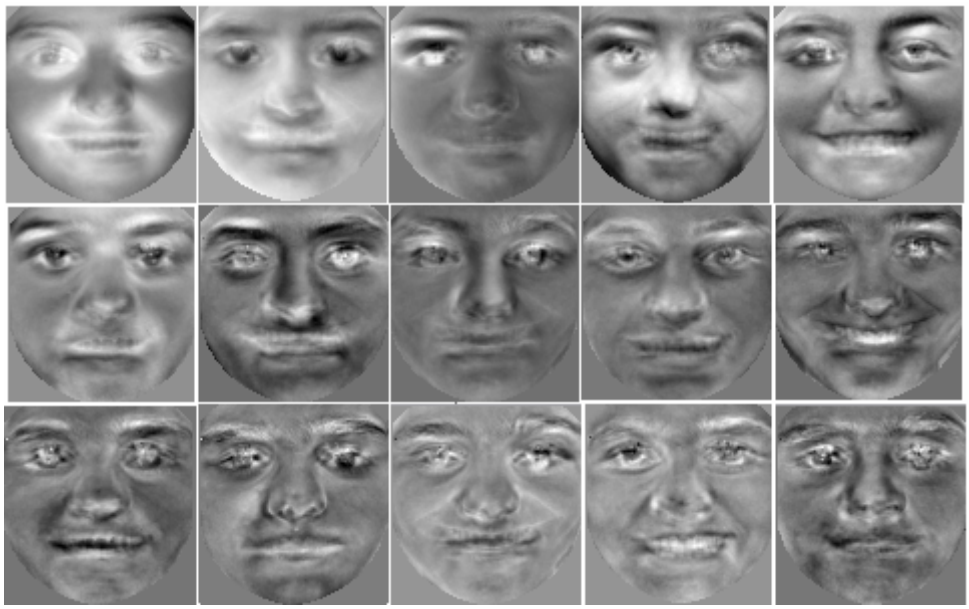
Şekil 65. Özvektörlerin normalize edilişi

Hesaplanan özvektörler resimlerin aktarılacağı yeni uzaya dönüşümü sağlayacak dönüşüm matrisini oluşturmaktadırlar. Dönüşümde kullanılacak özvektörleri sütunlarında taşıyan *Ozv* matrisinin boyutu *10000*15* olarak hesaplanmaktadır. Bu özvektörlerin

```
for x=1: ornek_sayisi
  Ayuz = TOzv(x,:);MX = max(Ayuz);MN = min(Ayuz);
  for a=1:boy
     for b=1:en
        ara=Ayuz((a-1)*en+b);    Yuz(a,b)=255*((ara-MN)/(MX-MN));       end end end
```

Şekil 66. Özyüzlerin elde edilişi

bileşenleri *0-255* arasına Şekil 66'daki gibi ölçeklenir ve elde edilen *15* özyüz Şekil 67'de görülmektedir.

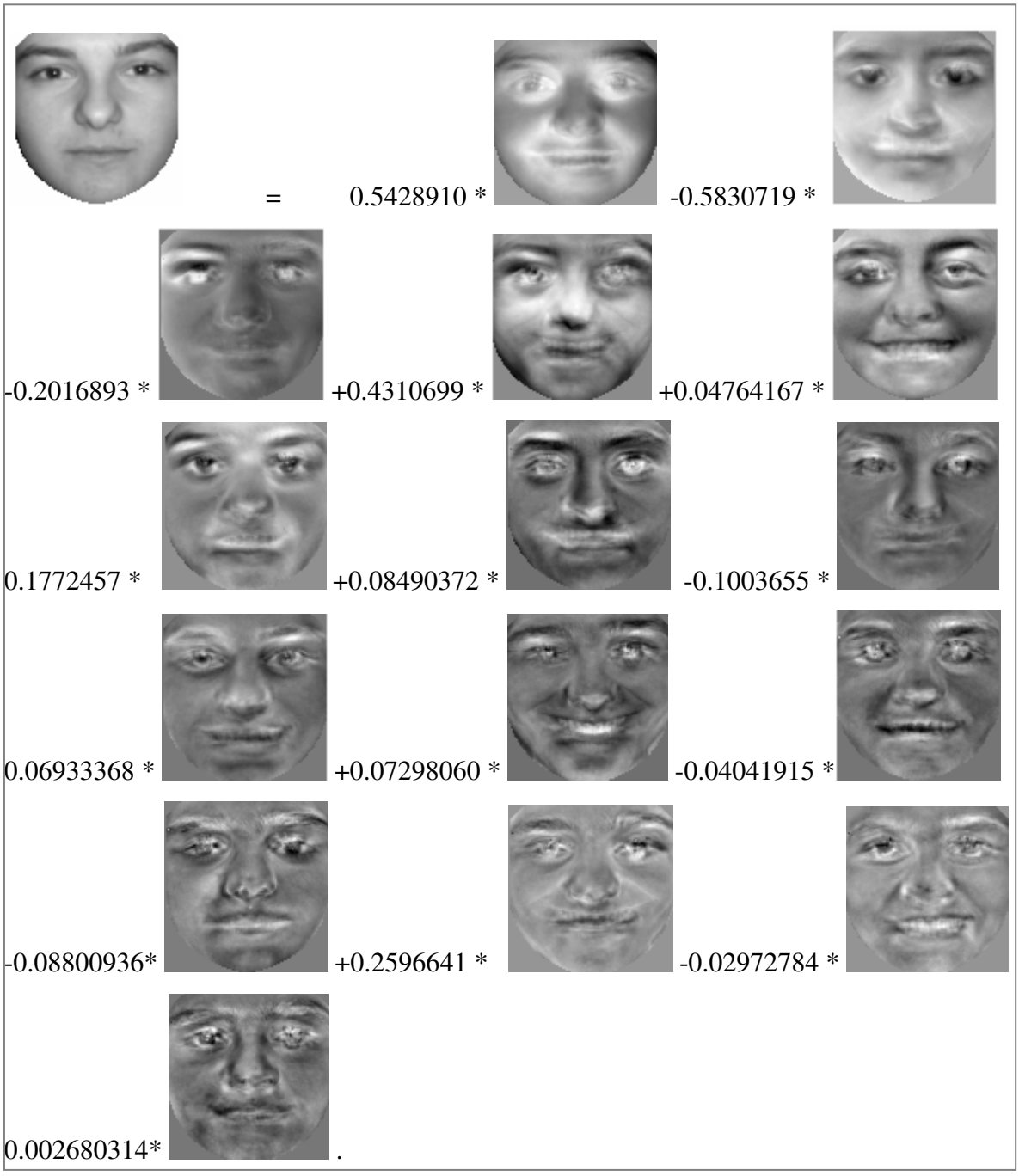


Şekil 67. Özyüzler

Boyut sayısı belirlenen özyüz uzayına bir giriş resminin taşınması o resmin uzaya izdüşümü ile gerçekleştirilmektedir. Bunun için, elde edilen özvektörler veri matrisi ile çarpılır. *Ozv* matrisinde özvektörler sütunlarda olduğundan, dönüşüm işleminde matrisin transpozu alınarak, herbir özvektörün *At*' nin sütunlarındaki resimlerle çarpımıyla resimlerin özvektörlerle belirlenen uzaya dönüşüm işlemi gerçekleştirilmektedir.

$$H = Ozv' * At; \tag{43}$$

Elde edilen *H* matrisi özyüz uzayına taşınan resimleri sütunlarında tutar ve boyutu *15*31* dir, böylece artık resimler yeni taşındıkları uzayda 15 bileşenle temsil edilmektedirler. Örnek olarak, aşağıda bir giriş resminin dönüşüm yapıldıktan sonra özyüzler cinsinden ifade edilişi örneklendirilmiştir (Şekil 68).



Şekil 68. Bir yüzün özyüzler cinsinden ifade edilişi

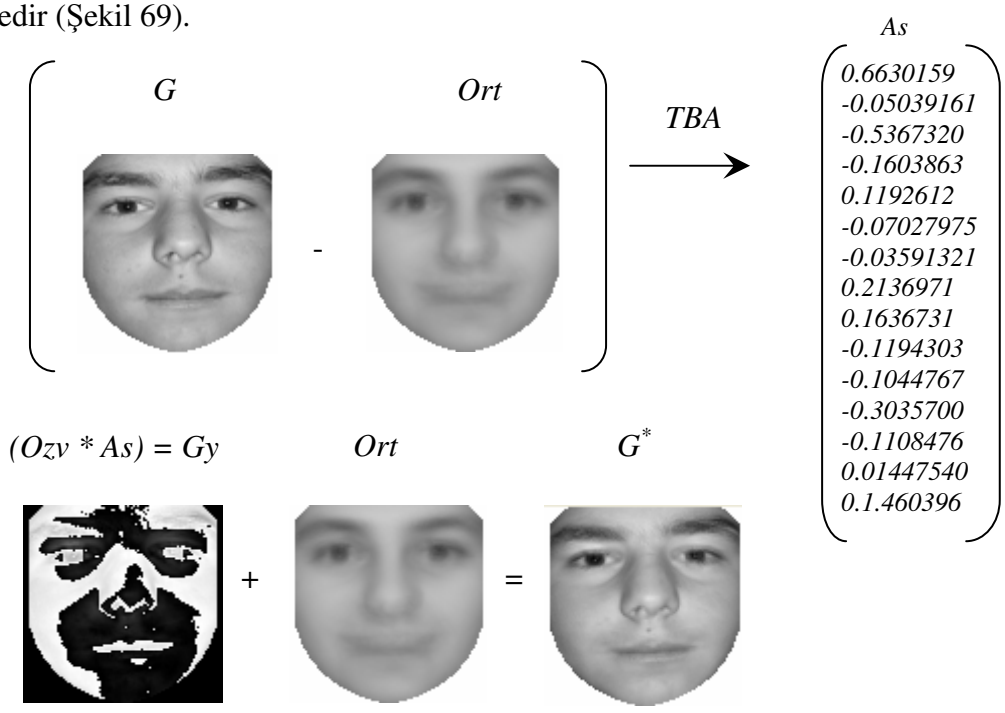
Resim özyüzler uzayına taşındıktan sonra, yeniden benzer bir hesaplamayla geri dönülerek resmin ilk hali çok az bir hata ile elde edilebilmektedir. Giriş resmini tek sütunda içeren *10000*1* boyutlu *G* matrisini özyüz uzayına taşıyarak aşağıdaki şekilde *15*1* boyutlu karşılığı elde edilir. (*Gy* matrisi, giriş resminden veritabanındaki tüm resimlerin ortalaması çıkarılarak elde edilir, *Gy = G – Ort*.)

$$As = Ozv\,' * Gy; \tag{44}$$

Gy' yi tekrar elde etmek için aşağıda da görüldüğü gibi *As* yi soldan *Ozv* ile çarpmak yeterlidir. (*Ozv* ortogonal olduğu için tersi transpozuna eşittir.)

$$Gy = (Ozv')^{-1} * As; \quad Gy = (Ozv')' * As; \quad Gy = Ozv * As; \tag{45}$$

Gy matrisine ortalama resim eklenerek giriş resmi tekrar elde edilir. Giriş resminin özyüz uzayına taşındıktan sonra tekrar elde edilmesi aşağıda bir örnek üzerinde ifade edilmektedir (Şekil 69).



Şekil 69. Giriş resminin özyüz uzayına taşındıktan sonra G^* olarak elde edilmesi

En son elde edilen resimle ilk halinin birbirine yakın olması, bu metodun ayrıca çok iyi bir sıkıştırma tekniği olduğunu da göstermektedir. Hesaplanan tüm özvektörler kullanıldığında ise resim tamamen ilk haliyle elde edilmektedir (Şekil 70).

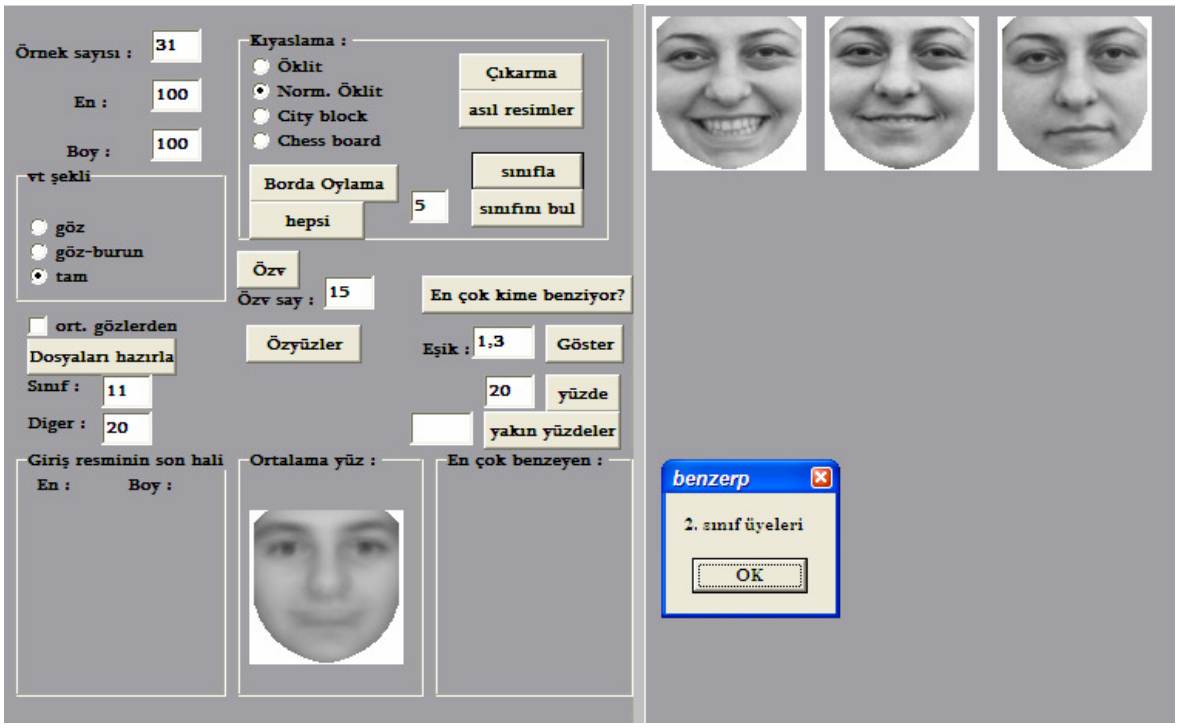
Şekil 70. Giriş resminin özyüz uzayına tüm özvektörlerle taşındıktan sonra tekrar elde edilmesi

2.5.2. TBA ile Yapılan Dönüşüm Sonuçlarının Değerlendirilmesi

TBA ile dönüştürülen yüzlerin birbirleri ile kıyaslanmasında aralarındaki *normalize edilmiş Öklit* uzaklığına bakılmaktadır. Kıyaslanacak 2 vektör χ ve χ^* ise aralarındaki *normalize edilmiş Öklit* uzaklığı aşağıdaki şekilde tanımlıdır.

$$\sum_{i=1}^{K} \frac{1}{\lambda_i}(\chi_i - \chi_i^*)^2 \tag{46}$$

Bu şekilde, tüm eksenler boyunca varyasyonlara aynı derecede önem verilmiş olunur. Bu kıyas kullanılarak yapılan sınıflamalardan biri aşağıda Şekil 71'de görülmektedir.



Şekil 71. Sınıflama

Aşağıda Şekil 72'de veritabanından olmayan bir giriş resmine bu yöntemle bulunan veritabanındaki en yakın resim görülmektedir.

Şekil 72. Giriş resmine en yakın resmin bulunması

3. SONUÇLAR VE İRDELEMELER

3.1. Yüz Tespiti ile İlgili Sonuçlar ve Yorumlar

Ten işleme ve YSA' nın birlikte kullanıldığı yüz tespiti kısmında %95 başarı sağlanmıştır. Yüz tespiti kısmında işlenecek giriş görüntüsünde ten bölgeleri bulunarak ağa sunulacak görüntü bölümü yani görüntü üzerinde yapılan tarama sayısı azaltılmıştır. Ten bölgelerinin tespiti, yüz aranacak bölgeleri belirlediğinden, yüz tespiti bölümünün başarısını etkilemektedir. Ten rengini bulurken, incelenen görüntünün YCbCr uzayına dönüştürülmüş piksel değerlerinin belli bir aralığa düşüp düşmediği incelenirken aralıkla tanımlanan bölge Cb Cr değerleri üzerinde bir dörtgen olmaktadır. Ancak verilerin dağılımı (Şekil 38.a) daha çok bir üçgene benzemektedir. Bu yüzden bazı değerler gerçekten ten renginde oldukları halde aralık dışında kalmaktadır. Buna rağmen, ten bölgesi tespiti kısmında yüksek oranda başarı elde edilmiştir.

Yüz tespiti kısmının, doğru sonuca varması ten bölgesi tespiti ile güçlendirilmiş olsa da, başarısı ağın iyi eğitilmiş olması ile yakından ilgilidir. Ağın eğitiminin ardından sorulan örneklere başarılı sonuçlar üretebilmesini etkileyen birçok etkenden biri ağın eğitiminde kullanılan örneklerdir. Ağın eğitiminde kullanılan veritabanının genişletilmesi ile sistemin başarısı arttırılabilir. Örneklerin seçimi, ağırlıklar örneklere göre değiştirildiğinden ağın performansı üzerinde direk etkilidir. Belli bir aşamaya kadar öğrenilemeyen örneklerden vazgeçilmesi performansı arttırabilmektedir. Örneğin, 1000 iterasyondan sonra beklenen değer ile hesaplanan ağ çıkışı arasındaki farkın 0.6'dan büyük olduğu örneklerin atlanmasının eğitim aşamasını hızlandırdığı görülmüştür.

Ağın yüzleri ayırd edebilmesi için eğitmek zor bir işlemdir, çünkü yüz olmayan örnekleri belirlemek zordur. Ayrıca, ilk eğitimden sonra sorgulama kısmında üretilen, istenilen sonuca yakın olmayan sonuçların elenmeleri amacıyla, kötü örnek setine (beklenen değeri 0 olan set) eklenmelerinin ve eğitimin tekrarlanmasının sonucu iyileştirdiği gözlendi.

Ağın eğitiminde 280 yüz örneği ve 320 yüz olmayan örnek olmak üzere 600 örnek kullanıldı. Ağın performansını etkileyen diğer unsurlardan ağdaki katman sayısı ve katmanlardaki eleman sayıları doğru seçilerek ağın istenen hata değerine inmesi sağlanmaktadır. Oval maskeden geçirilen girişler ile ağdaki giriş sayısı 551 olarak belirlenmiştir. Geriye doğru hesaplamada kullanılan öğrenme katsayısı ve momentum katsayısı değerleri de ağın öğrenme başarısını etkilemektedir. Momentum katsayısı 0.2,

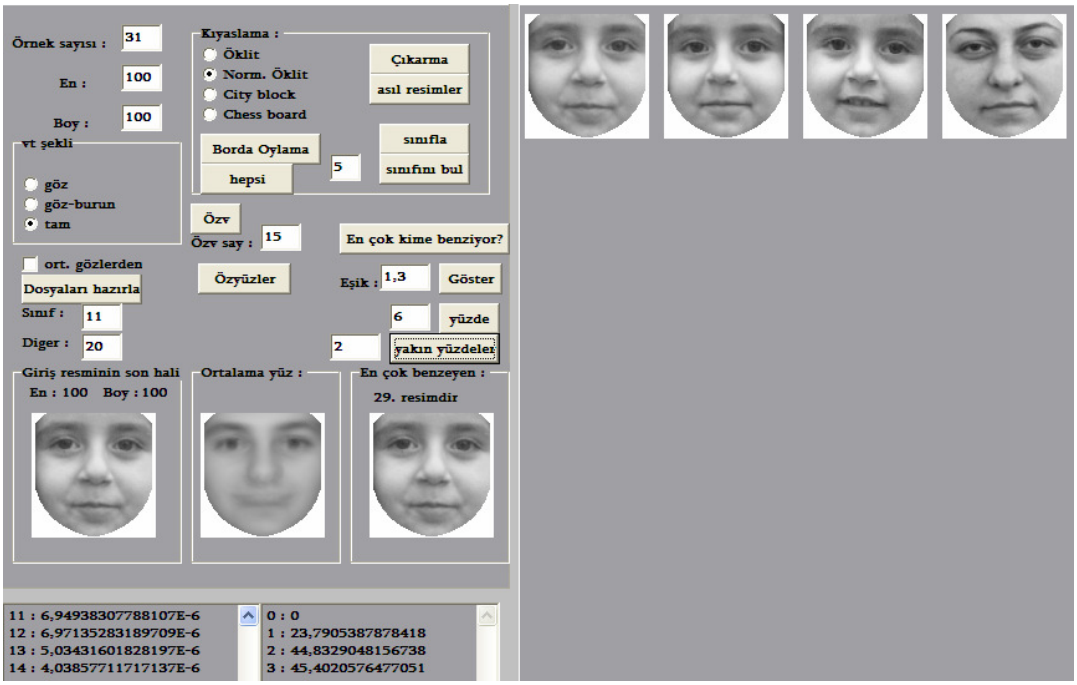
öğrenme katsayısı 0.1 olarak alındı. Öğrenme katsayısının büyük değerleri ağın yerel çözümler arasında dolaşmasına neden olmaktadır, küçük değerleri de öğrenme zamanını arttırmaktadır. Momentum katsayısının küçük seçilmesi yerel çözümlerden kurtulmayı zorlaştırabilir, büyük değerleri de tek çözüme ulaşmada sorun çıkarabilir. Ara katmandaki eleman sayısı da ağın başarısını etkilemektedir ve çalışmada 40 olarak alınmıştır.

Ayrıca, görüntü küçültülürken kullanılan ölçekleme oranı ve içeriği ağa sunulan pencerenin kaydırma adımı parametreleri de yüzün yakalanmasında hassasiyeti etkilemektedirler.

Sistemin giriş bilgisi olarak dik yönde bakan yüzler ele alınmıştır. Fakat $\pm 15^0$ 'lik yüz dönmeleri de sistem tarafından tespit edilmektedir. Ayrıca daha fazla dönmüş ve eğik yüzleri yakalamak için ikinci bir sinir ağı sisteme eklenebilir.

3.2. Benzerlik incelemesi ile ilgili Sonuçlar ve Yorumlar

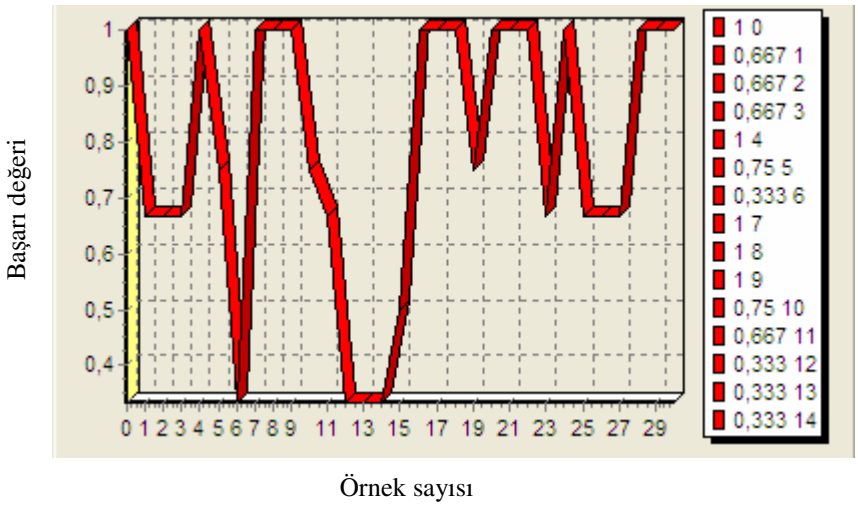
TBA analiziyle dönüştürülen 31 elemanlı veritabanındaki görüntüler arasındaki ilişkiler *normalize edilmiş Öklit* uzaklığı ile değerlendirildiğinde, %80 başarı elde edilmektedir. Veritabanı kişilerin daha farklı yüz görüntüleriyle genişletildiğinde daha hassas bir başarı oranına ulaşılabilir. Aşağıda (Şekil 73) *normalize edilmiş Öklit* uzaklığı kullanılarak yapılan örnek kıyaslama sonuçları görülmektedir. Şekil 73'te sol alttaki giriş görüntüsü için sağ üstte benzer olarak bulunan örnekler görülmektedir. Başarı hesaplamasında, veritabanındaki herbir örnek için yapılan kıyaslamalarda kişilere ait görüntülerin sonuç listesinde görünme yerlerine göre ağırlıklandırılmış ortalamalar göz önüne alınmıştır. Herbir örnek kişi için kişiye ait tüm görüntüler sonuç listesinde ard arda ilk sıralarda gelirse, o örneğin başarıya katkısı 1 olarak alınmaktadır, yani bir örnekten en iyi durumda başarı hesaplamasına 1 değeri gelir. Eğer, örnekteki kişiye ait 3 yüz görüntüsü varsa ve bunlardan yalnızca 2 tanesi ilk 3' te yer almışsa bu kez o örnek için başarıya katkı 2/3 olarak alınmaktadır. Bu şekilde herbir örnek için kıyaslama ile elde edilen değerlendirme grafiği Şekil 74'te görülmektedir.



(a)

(b)

Şekil 73. (a) 1. ve (b) 2. örnek üzerinde *Normalize edilmiş Öklit* uzaklığı ile kıyaslama



Şekil 74. *Normalize edilmiş Öklit* uzaklığı kullanılarak elde edilen değerlendirme grafiği

Yalnızca *normalize edilmiş Öklit* uzaklığı ile elde edilen sonuçla karşılaştırma yapmak için *Öklit*, *normalize edilmiş Öklit*, *City-Block*, *Chess-board* uzaklık ölçme yöntemlerinin verdiği sonuçlara dayalı oylama yoluyla (Borda sayısı yöntemi (bkz. Ek 6)) farklı bir değerlendirme de yapılmıştır. Aşağıda *K* bileşenli χ ve χ^* vektörleri arasındaki *Öklit*, *City-Block*, *Chess-board* uzaklıkları sırasıyla (1), (2) ve (3) ile verilmektedir.

$$(\sum_{i=1}^{K} (\chi_i - \chi_i^*)^2)^{\frac{1}{2}} \tag{47}$$

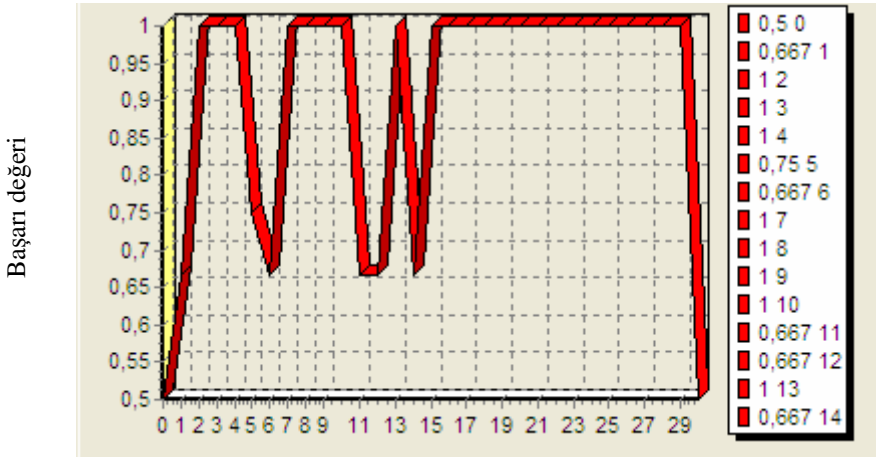
$$\sum_{i=1}^{K} \left|\chi_i - \chi_i^*\right| \tag{48}$$

$$\max(\left|\chi_i - \chi_i^*\right|),\ i=1,...K \tag{49}$$

Aşağıda, *Öklit* uzaklığı kullanılarak elde edilen bir kıyaslama örneklendirilmektedir (Şekil 75).

Şekil 75. *Öklit* uzaklığı ile kıyaslama

Aşağıda veritabanındaki örneklerin herbiri için oylama yoluyla yapılan değerlendirme grafiği (Şekil 76) ve örnek bir giriş görüntüsü için 6 aday üzerinde oylama sonucu (Şekil 77) görülmektedir ve bu yolla yapılan değerlendirmede %91 başarı elde edilmiştir.



Şekil 76. Oylama yoluyla elde edilen değerlendirme grafiği

Şekil 77. Örnek giriş için oylama sonucu

Şekil 77'de görüldüğü gibi sonuç listesindeki örneklere verilen oylar 16, 9, 6' dır ve oylar benzerliklerle orantılıdır. Bu örnek için veritabanındaki aynı kişiye ait tüm örnekler ilk sıralarda ard arda bulunduğundan bu sorgulama tam başarıyla sonuçlanmıştır.

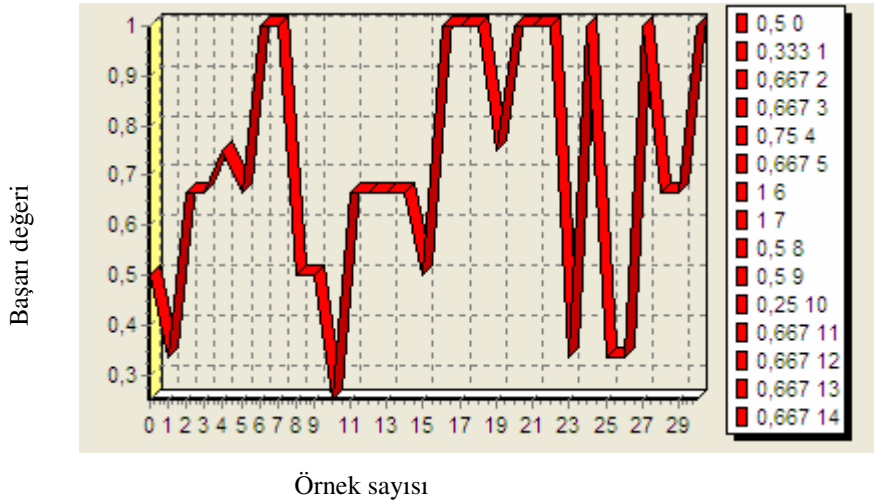
Görüntüleri yalnızca göz ve burun arasındaki bölgeden kıyaslayarak, yüzler arasındaki benzerliği tespit ederken kişilerin yüzündeki duygusal durumundan veya bıyık, sakal gibi etkenlerden kaynaklanan değişikliklerin sonuca etkisini azaltmak mümkün olabilmektedir. Bu amaçla görüntülere TBA uygulanmadan önce göz seviyeleri tespit edilerek bu göz seviyesinden itibaren yukarı ve aşağı yönde her görüntüde eşit uzaklıktaki bölgeler alınarak işlem yapılmaktadır. Aşağıda örnek bir görüntü için tespit edilen ve üst köşeleri maskeden geçirilen göz-burun bölgeleri görülmektedir (Şekil 78).

Şekil 78. Örnek resim üzerinde belirlenen göz-burun bölgesi

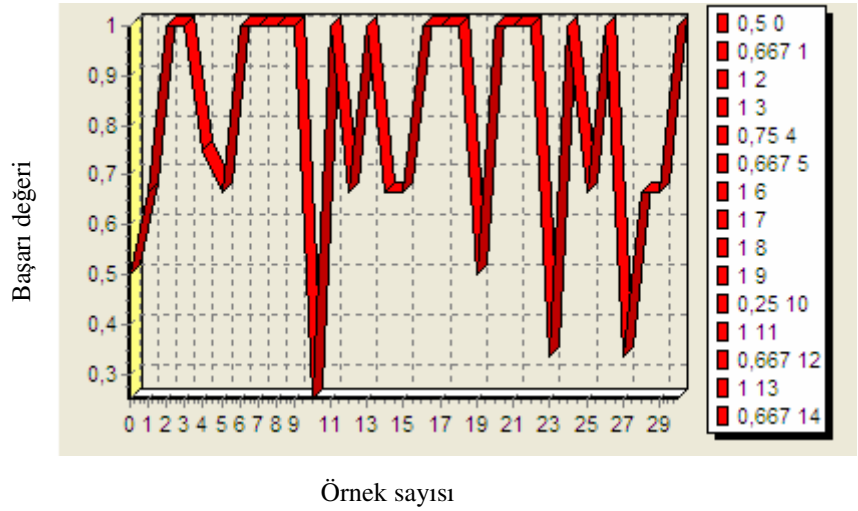
Giriş görüntüsü göz ve burun arası bölge olarak alınıp TBA uygulandıktan sonra *normalize edilmiş Öklit* uzaklığı kullanılarak elde edilen kıyaslama sonuçları aşağıdadır (Şekil 79).

Şekil 79. Göz-burun bölgesinden örnek kıyaslama

Aşağıda göz-burun bölgelerinden oluşan giriş görüntülerine TBA uygulanıp *normalize edilmiş Öklit* uzaklığı ve oylama ile elde edilen değerlendirme grafikleri görülmektedir (Şekil 80 ve 81). Göz-burun bölgelerinden kıyaslama işleminde *normalize edilmiş Öklit* ile %72, oylama ile %81 başarı elde edilmiştir.



Şekil 80. Örneklerin göz-burun bölgesinden *normalize edilmiş Öklit* ile değerlendirilmesi



Şekil 81.Örneklerin göz-burun bölgesinden oylama ile değerlendirilmesi

Aşağıdaki tablodaki (Tablo 4) satırlar herbir örneğin yukarıdaki grafiklerde gösterilen başarıya katkısıdır. Herbir sütunda sırayla; tam yüzlerin oylama ile değerlendirilmesi, göz-burun bölgelerinin oylama ile değerlendirilmesi, tam yüzlerin sadece *normalize edilmiş Öklit* uzaklığı ile değerlendirilmesi ve göz-burun bölgesinin sadece *normalize edilmiş Öklit* uzaklığı ile değerlendirilmesine ait grafiklerde verilen değerler vardır. Buradan görülür ki; tablodaki değerleri sezgisel fonksiyonlarla birleştirmek mümkündür.

Tablo 4. Herbir örneğin başarıya katkısı

	1	2	3	4		1	2	3	4
1	0,5	0,5	1	0,5	17	1	1	1	1
2	0,6667	0,6667	0,6667	0,3334	18	1	1	1	1
3	1	1	0,6667	0,6667	19	1	1	1	1
4	1	1	0,6667	0,6667	20	1	0,5	0,75	0,75
5	1	0,75	1	0,75	21	1	1	1	1
6	0,75	0,6667	0,75	0,6667	22	1	1	1	1
7	0,6667	1	0,3334	1	23	1	1	1	1
8	1	1	1	1	24	1	0,3334	0,6667	0,3334
9	1	1	1	0,5	25	1	1	1	1
10	1	1	1	0,5	26	1	0,6667	0,6667	0,3334
11	0,6667	0,25	0,75	0,25	27	1	1	0,6667	0,3334
12	0,6667	1	0,6667	0,6667	28	1	0,3334	0,6667	1
13	1	0,6667	0,3334	0,6667	29	1	0,6667	1	0,6667
14	0,6667	1	0,3334	0,6667	30	1	0,6667	1	0,6667
15	1	0,6667	0,3334	0,6667	31	0,5	1	1	1
16	1	0,6667	0,5	0,5					

1.sütun: tam yüzlerin oylama ile değerlendirilme sonuçları

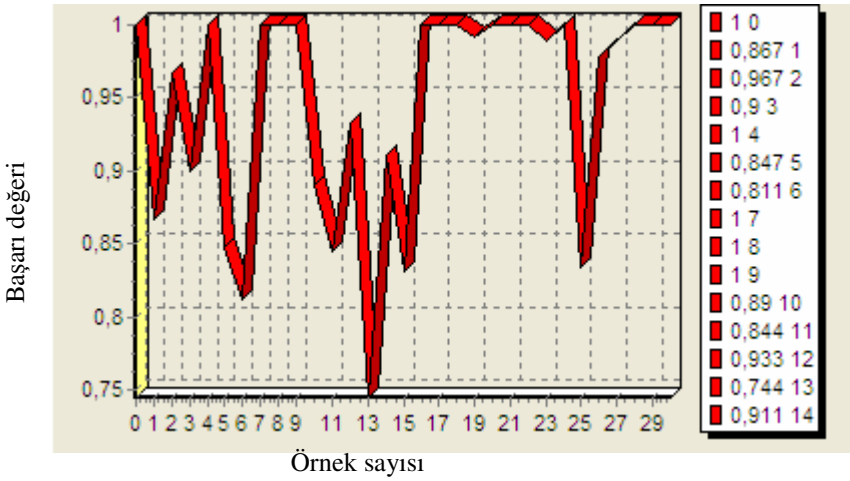
2. sütun: göz-burun bölgelerinin oylama ile değerlendirilme sonuçları

3. sütun: tam yüzlerin *normalize edilmiş Öklit* uzaklığı ile değerlendirilme sonuçları

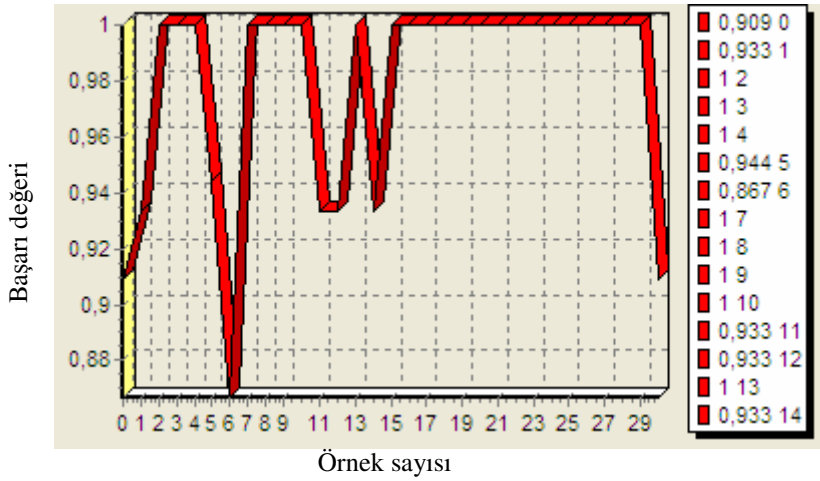
4. sütun: göz-burun bölgelerinin *normalize edilmiş Öklit* uzaklığı ile değerlendirilme sonuçları

Örneklerin başarıya katkısının hesaplanmasında bir kişiye ait farklı yüzlerin sonuç listesindeki yerlerine göre değerlendirme yapılırken biraz daha farklı bir yol izlenirse başarı değerleri daha yüksek çıkmaktadır. Eğer, örnekteki kişiye ait 3 yüz görüntüsü varsa ve bunlardan biri sonuç listesinde 4. sırada gelmişse de bu örnek için başarı değerlendirmesine katkı 2/3 olmaktadır. Bu gibi durumları göz önüne alarak değerlendirmeye almak için şöyle bir yol da izlenebilir: sorgulamada kullanılan örneğe ait 3 tane yüz görüntüsü sonuç listesinde ilk 3 sırada beklenirken, 1., 2. ve 4. sırada bulunmuşsa bu örneğin başarıya katkısı tüm örnek sayısı, 31, göz önüne alınarak (31+30+28)/(31+30+29) = 0.98 şeklinde hesaplanabilir. Bu şekilde *normalize Öklit*

uzaklığını kullanarak tam yüzlerde elde edilen değerlendirme grafiği aşağıdadır (Şekil 82) ve başarı %95'tir. Benzer şekilde, tam yüzlerde oylama ile yapılan değerlendirmede, aynı örnek için başarı katkısı (6 oy üzerinden) (6+5+3)/(6+5+4) olarak hesaplanır ve bu şekilde elde edilen başarı %91' den %98' e yükselmektedir (Şekil 83). Bu şekilde, göz-burun bölgelerinden kıyaslamada *normalize edilmiş Öklit* ile %92 (bkz. Ek Şekil 3), oylama ile %88 (bkz. Ek Şekil 4) başarı hesaplanmıştır. Aslında daha hassas bir değerlendirme yapmak için beklenen sıralamada gelen örneklere eşit değer verme yoluna da gidilebilir.



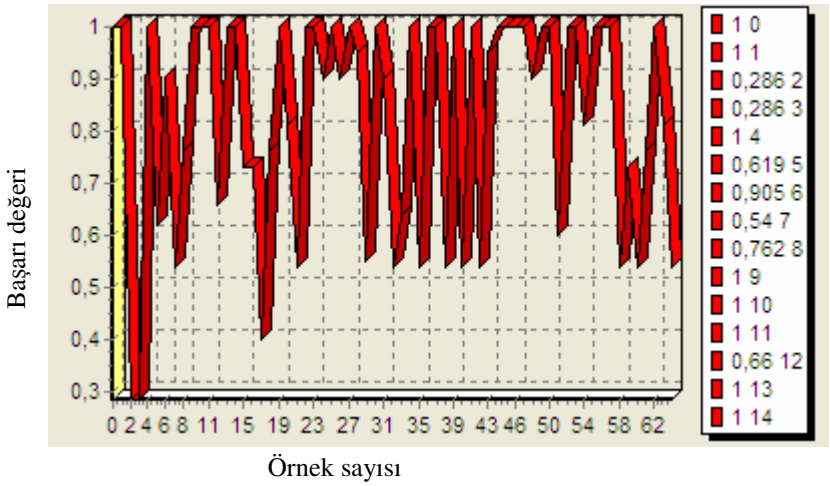
Şekil 82. *Normalize Öklit* uzaklığını kullanarak tam yüzlerde sonuçların yerlerine göre elde edilen değerlendirme grafiği



Şekil 83. Tam yüzlerde oylama ile yapılan değerlendirmede sonuçların yerlerine göre başarı hesaplanması

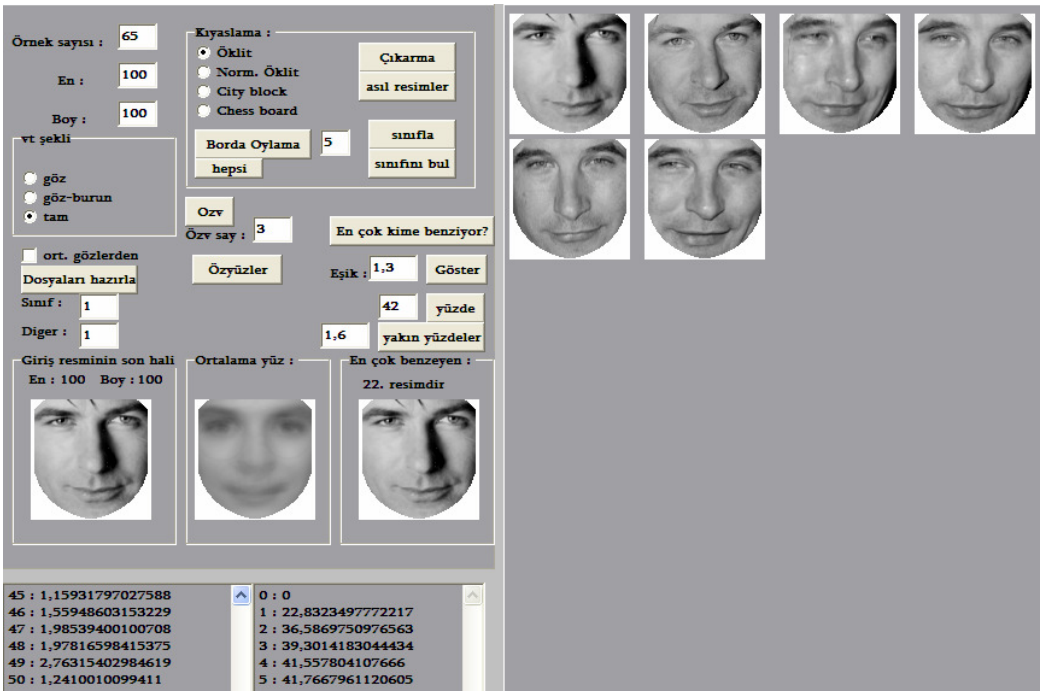
Aşağıda örneklerinden bazıları görülen (Şekil 84) genişletilmiş birbirinden daha farklı örneklerden oluşan veritabanı üzerinde oylamaya dayalı ilk değerlendirmede başarı %83 olmuştur (Şekil 85). Bu veritabanındaki (bkz. Ek 8) birkaç örnek için yapılan kıyaslama sonuçları da aşağıda görülmektedir (Şekil 86, Şekil 87, Şekil 88, Şekil 89).

Şekil 84. Genişletilmiş veritabanından örnekler

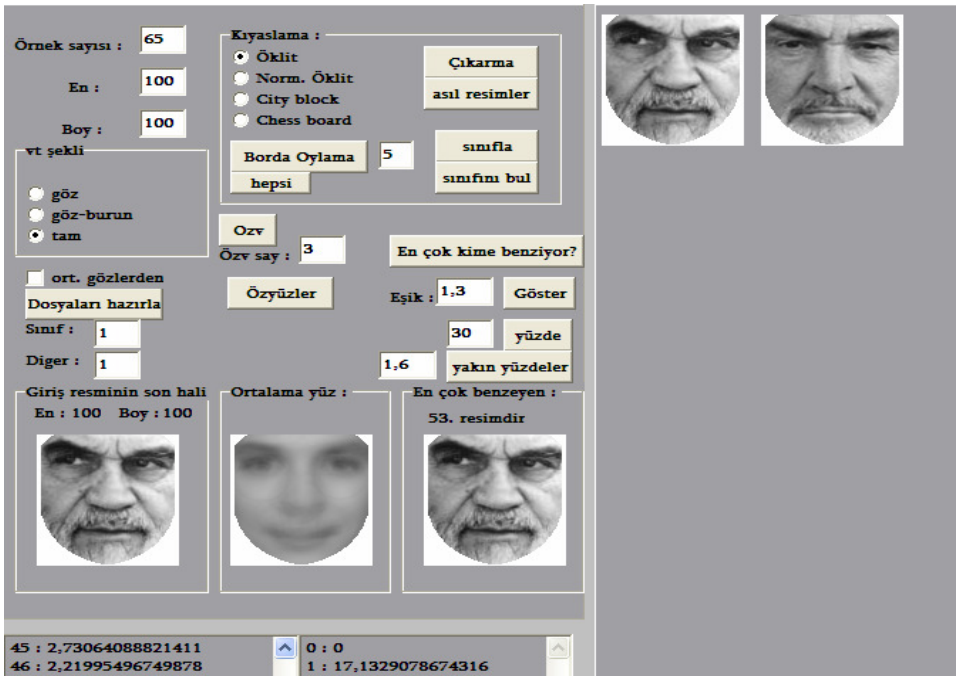


Şekil 85. Genişletilmiş veritabanı üzerinde oylamaya dayalı değerlendirme grafiği

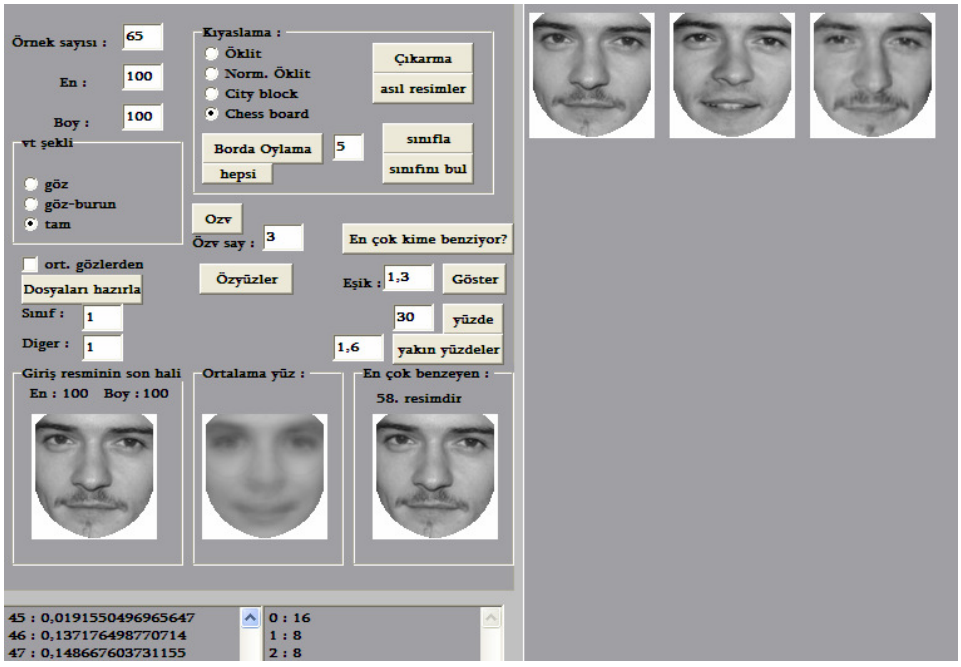
Şekil 86. Genişletilmiş veritabanı üzerinde örnek kıyaslama 1 (bu örnek için yapılan yüz tespiti işlemi aşamaları Ek 9'de gösterilmektedir.)



Şekil 87. Genişletilmiş veritabanı üzerinde örnek kıyaslama 2



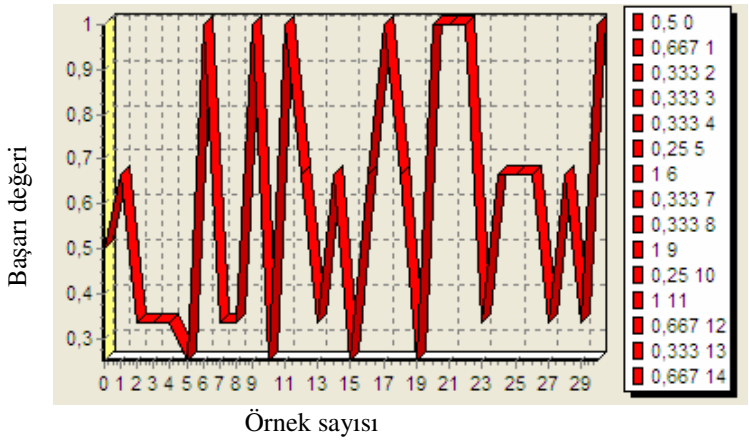
Şekil 88. Genişletilmiş veritabanı üzerinde örnek kıyaslama 3



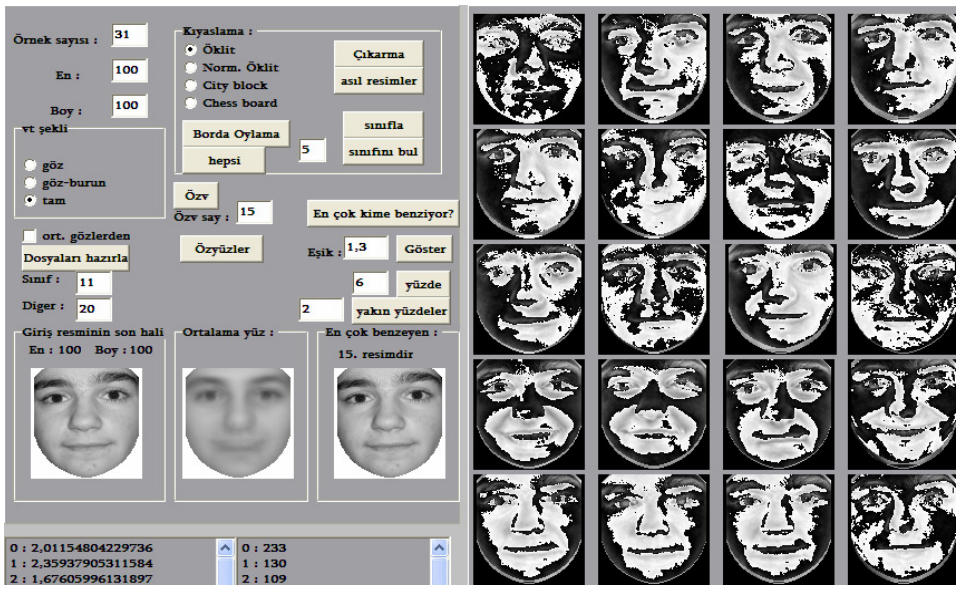
Şekil 89. Genişletilmiş veritabanı üzerinde örnek kıyaslama 4

TBA dönüşümünün sağladığı verimli sonuçlarla karşılaştırmak açısından görüntülerin çıkarma işlemi ile kıyaslanmaları incelendi. Giriş görüntüsü ve kıyaslanacağı veritabanındaki görüntülerin yatay göz konumları örtüşecek şekilde birbirlerinden

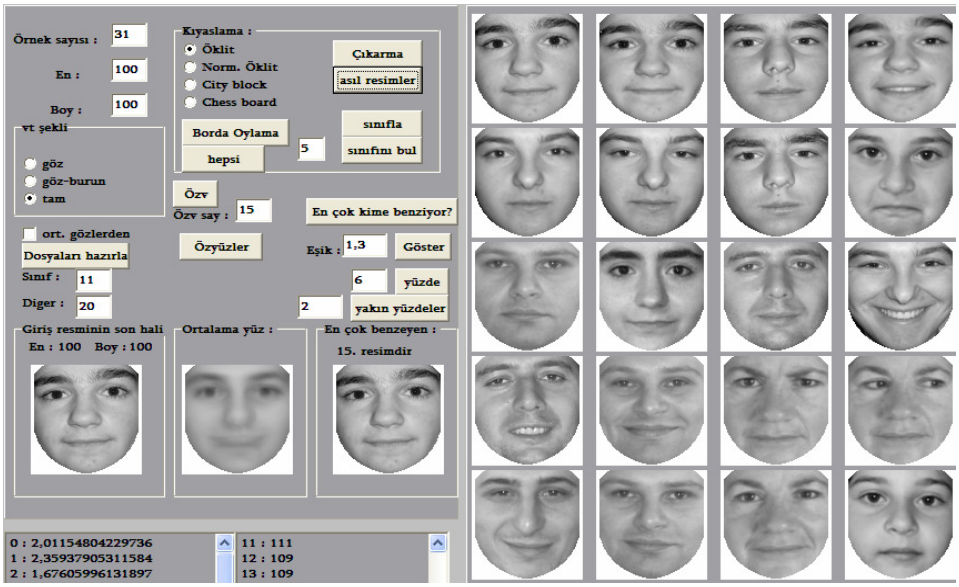
çıkarılmasıyla elde edilen sonuçlar üzerinde görüntülerin listedeki yerlerine göre yapılan değerlendirmede %60 başarı elde edildi (Şekil 90). Bu şekilde yapılan bir kıyaslama örneği aşağıda görülmektedir (Şekil 91, Şekil 92). Şekil 91'de giriş resminin yatay göz hizaları üst üste gelecek şekilde veritabanındaki tüm resimlerden çıkarılmasıyla elde edilen sonuçların bir kısmı görülmektedir. Çıkarma işlemi sonucunda tüm görüntülerdeki 0 değerli pikseller sayılarak büyükten küçüğe doğru sıralanmaktadır. En fazla 0 değerli pikseli içeren çıkarma sonucunu veren görüntü sorulana en benzerdir ve belirlenen en benzer görüntüler Şekil 92' de sıralı olarak görülmektedir.



Şekil 90. Çıkarma sonucu elde edilen değerlendirme grafiği

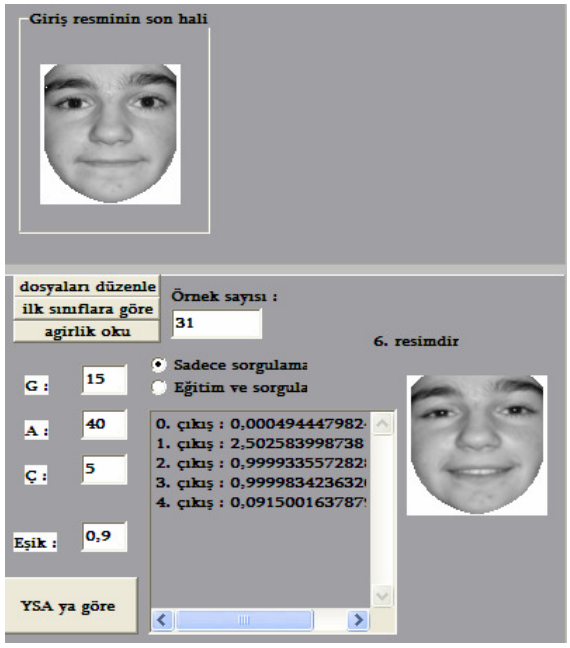


Şekil 91. Veritabanındaki görüntülerin bir kısmı ile yapılan çıkarma işlemi sonucu



Şekil 92. Çıkarma işlemi ile yapılan kıyaslamada en benzer görüntüler

TBA ile elde edilen dönüşüm sonuçlarının değerlendirilmesinde YSA kullanılabilmektedir. Bu amaçla veritabanının %35' i ağın eğitiminde kullanıldı. Eğitim seti dışından sorulan örnekler için %80 başarı elde edildi. Aşağıda eğitimde kullanılmayan giriş resmi için YSA ile elde edilen en benzer sonuç görülmektedir (Şekil 93). YSA ' nın giriş sayısı TBA ile yapılan dönüşüm sonucu elde edilen özvektör sayısı yani 15'tir.



Şekil 93. YSA ile değerlendirme örneği

4. KAYNAKLAR

1 Jain, A. K., Ross, A., ve Prabhakar, S., An Introduction to Biometric Recognition, IEEE transactions on circuits and systems for video technology, 14, 1 (2004) 1-29.

2 www.trigraph.ie/trigraph/Main/Briefings_Biometrics.htm Biyometri. 16 Hziran 2007.

3 Karakaya, T., Yüz Tanıma ve Yapay Sinir Ağları Yardımıyla Yüzün Tespiti, Yüksek Lisans Tezi, K.T.Ü., Fen Bilimleri Enstitüsü, Trabzon, 2005.

4 Parramon, J., Baş ve Portre Çizme Sanatı, Erol Erduran, 6. Basım, Remzi Kitapevi A.Ş., İstanbul, 2000.

5 Craw, I., Tock, D., ve Bennett, A., Finding Face Features, Proc. 2nd European Conf. Computer Vision, 1992, 92-96.

6 Lanitis, A., Taylor, C. J. ve Cootes, T. F., An Automatic Face Identification System Using Flexible Apperance Models, Image and Vision Computing, 13, 5 (1995) 393-401.

7 Hjelms, E. ve Low, B. K., Face Detection: A Survey, Computer Vision and Image Understanding, 83, (2001) 236-274.

8 Rowley, H. A., Baluja, S. ve Kanade, T., Neural Network-Based Face Detection, In Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition (CVPR '96), 1996, 203-207.

9 Rowley, H. A., Baluja, S. ve Kanade, T., Rotation Invariant Neural Network-Based Face Detection, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1998, 38-44.

10 www.lrv.fri.uni-lj.si/~peterp/publications/cvww99.pdf An Automatic Human Face Detection Method. 23 Mart 2007.

11 Shin, M. C., Chang, K. I. ve Tsap, L. V., Does Colorspace Transformation Make Any Difference on Skin Detection, Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision, 2002, 275–279.

12 Samal, A., ve Iyengar, P. A., Automatic Recognition And Analysis of Human Faces And Facial Expressions: A Survey, Pattern Recognition, 25 (1992) 65–77.

13 Goldstein, A. J., Harmon, L. D. ve Lesk, A. B., Identification of Human Faces, Proceedings of the IEEE, 59, 5 (1971) 748–760.

14 Brunelli, R. ve Poggio, T., Face Recognition: Features versus Templates, IEEE Transactions on Pattern Analysis and Machine Intelligence, 15 (1993) 1042–1052.

15 Guo, G., Li, S.Z. ve Chan, K., Face Recognition by Support Vector Machines, Proceedings. Fourth IEEE International Conference Automatic Face And Gesture Recognition, 2000, 196-201.

16 Cox, I. J., Ghosn, J. ve Yianilos, P., Feature-Based Face Recognition Using Mixture-Distance, In Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition (CVPR '96), 1996, 209–216.

17 Valentin, D., Abdi., H., O'Toole, A. J. ve Cottrell, G. W., Connectionist Models of Face Processing: A Survey. PatternRecognition, 27 (1994) 1209–1230.

18 Chellappa, R., Wilson, C. L. ve Sirohey, S., Human And Machine Recognition of Faces: A Survey. Proceedings of the IEEE, 83 (1995) 705–741.

19 Turk, M., ve Pentland, A., Eigenfaces for Recognition, Journal of Cognitive Neuroscience, 3, 1 (1991) 71-86.

20 Delac, K., Grgic, M. ve Grgic, S., Independent Comparative Study of PCA, ICA and LDA on the FERET Data Set, University of Zagreb, FER, Unska 3/XII, 2006, Zagreb, Croatia, 252-260.

21 Turk, M., A Random Walk through Eigenspace, IEICE Trans. Inf. & Syst., E84-D, 12, (2001) 1586-1595.

22 Turk, M., Pentland, A., Face Recognition Using Eigenfaces, Proc. of the IEEE Conf. On Computer Vision and Pattern Recognition, 1991, 586-591.

23 www.biometrics.gov Face Recognition, NSTC Subcommittee on Biometrics. 23 Haziran 2007.

24 Sirovich, L. ve Kirby, M., A Low-Dimensional Procedure for the Characterization of Human Faces, J. Optical Soc. Am. A, 4, 3 (1987) 519-524.

25 Bolme, D., Beveridge, R., Teixeira, M. ve Draper, B., The CSU Face Identification Evaluation System: Its Purpose, Features and Structure, International Conference on Vision Systems, 2003, Graz, Austria, 304-311.

26 Bartlett, M.S., Movellan, J.R. ve Sejnowski, T.J., Face Recognition by Independent Component Analysis, IEEE Trans. on Neural Networks, 13, 6 (2002) 1450-1464.

27 Draper, B., Baek, K., Bartlett, M. S. ve Beveridge, J. R., Recognizing faces with PCA and ICA, Computer Vision Image Understanding, 91, 1-2 (Special Issue on Face Recognition) (2003) 115-137.

28 Delac, K., Grgic, M. ve Liatsis, P., Appearance-based Statistical Methods for Face Recognition, 47th International Symposium ELMAR-2005, 2005, Zadar, 151-158.

29 Belhumeur, P., Hespanha, J., ve Kriegman, D., Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, Proc. of the Fourth European Conference on Computer Vision, 1996, Cambridge, 45-58.

30 Martinez, A. ve Kak, A., PCA versus LDA, IEEE Trans. on Pattern Analysis and Machine Intelligence, 23, 2 (2001) 228-233.

31 Zhao, W., Chellappa, R. ve Krishnaswamy, A., Discriminant Analysis of Principal Components for Face Recognition, Proc. of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition, 1998, Nara, 336-341.

32 Lu, J., Plataniotis, K. N. ve Venetsanopoulos, A. N., Regularized Discriminant Analysis For the Small Sample Size Problem in Face Recognition, Pattern Recognition Letters, 24, 16 (2003) 3079-3087.

33 Beveridge, J. R., She, K., Draper, B. ve Givens, G. H., A Nonparametric Statistical Comparison of Principal Component And Linear Discriminant Subspaces for Face Recognition, Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, 2001, Kaui, 535-542.

34 Baek, K., Draper, B., Beveridge, J. R. ve She, K., PCA vs. ICA: A Comparison on the FERET Data Set, Proc. of the Fourth International Conference on Computer Vision, Pattern Recognition and Image Processing, 2002, Durham, 824-827.

35 http://www.face-rec.org/interesting-papers/General/ImAna4FacRcg_lu.pdf Image Analysis for Face Recognition. 14 Haziran 2007.

36 http : // www . ait . gr / research / RG1 / files / ICCV_06.pdf EBGM vs. Subspace Projection for Face Recognition. 14 Haziran 2007.

37 ftp://ftp.computer.org/press/outgoing/proceedings/wacv07/Data/03_Local%20Graph %20Matching.pdf Local Graph Matching for Face Recognition. 14 Haziran 2007.

38 http : // www . cs . colostate . edu / ~ draper/publications/givens_cvpr04.pdf How Features of the Human Face Affect Recognition : a Statistical Comparison of Three Face Recognition Algorithms. 14 Haziran 2007.

39 http://www.uk.research.att.com/facedatabase.html ORL face database. 15 Haziran 2007.

40 http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/computerVision/ graphMatching/identification/faceRecognition/contents.html Face Recognition by Elastic Bunch Graph Matching. 15 Haziran 2007.

41 Negnevitsky, M., Artificial Intelligence A Guide to Intelligent Systems, 1. Basım, Addison-Wesley, England, 2002.

42 Öztemel, E., Yapay Sinir Ağları, 1. Basım, Papatya Yayıncılık, İstanbul, 2003.

43 www.dcs.gla.ac.uk/~sumitha/papers/FaceRecognition.pdf Frontal View Human Face Detection and Recognition. 24 Nisan 2007.

44 http://www.snl.salk.edu/~shlens/pub/notes/pca.pdf A tutorial on Principal Component Analysis. 24 Nisan 2007.

45 www.cs.cmu.edu/~guestrin/Class/10701-S06/Handouts/recitations/recitation-pca_svd .ppt Dimensionality Reduction. 24 Nisan 2007.

46 Romdhani, S., Face Recognition using Principal Components Analysis, Yüksek Lisans, Master of Science in Electronics Engineering at the University of Glasgow, Glasgow, 1996.

47 Gonzalez, R. C., Woods, R. E. ve Eddins, S. L., Digital Image Processing Using Matlab, 1. Basım, Pearson Prentice Hall, Upper Saddle River, 2004.

48 http : // csnet . otago . ac.nz/cosc453/student_tutorials/principal_components.pdf A Tutorial on Principal Components Analysis. 29 Mayıs 2007.

49 http://www.fho-emden.de/~hoffmann/gray10012001.pdf Luminance Models for the Grayscale Conversion. 29 Mayıs 2007.

50 http://www.sjsu.edu/faculty/watkins/eye.htm The Perceptive Efficiency of the Human Eye as a Function of Wavelength. 29 Mayıs 2007.

51 http://www.imageprocessingplace.com/DIP/dip_downloads/tutorials/contour_tracing_ Abeer_George_Ghuneim/moore.html Moore-Neighbor Tracing. 16 Haziran 2007.

52 Miao Jun, Liu Hong, Gao wen, Zhang Hongming, Deng Gang, Chen Xilin, A System for Human Face and Facial Feature Location, International Journal of Image and Graphics, 3, 3 (2003) 461-479.

53 http://www.img.cs.titech.ac.jp/ipcv/seminar/2004-2/ MorphologicalImage Processing .pdf Morphological Image Processing. 22 Mayıs 2007.

54 http://www.cee.hw.ac.uk/hipr/html/erode.html Erosion. 22 Mayıs 2007.

55 http://www.stanford.edu/class/ee368/Project_03/Project/reports/ee368group02.pdf Face Detection. 22 Mayıs 2007.

56 http://www.cse.lehigh.edu/~spletzer/cse398_Spring05/lec002_CMVision.pdf CMVision and Color Segmentation. 16 Nisan 2007.

57 http://nets.rwth–aachen.de/content/teaching/lectures/sub/mms/mmsSS07/03_Images 1_1P.pdf Images and Graphics. 12 Haziran 2007.

58 Nabiyev Vasif V., Algoritmalar Teoriden Uygulamalara, 1. Basım, Seçkin Yayıncılık, Ankara, 2007.

5. EKLER

Ek 1 (İstatistik)

Varyans

Tek boyuta sahip verilerden oluşan bir veri kümesindeki verilerin dağılımı hakkında bilgi veren istatistik terimidir. Verilerin dağılımı ile ilgili kullanılan bir diğer terim ise standart sapma olup bu terim varyansın karakökü ile ifade edilir. Standart sapma yerine varyans ile veri dağılımının ölçülmesi sonraki tanımlamalar ile daha yakın ilişki kurulmasını sağlayacaktır. Varyans formülü

$$s^2 = \frac{\sum_{i=1}^{n} \left(X_i - \overline{X}\right)^2}{(n-1)} \tag{E.1}$$

şeklinde tanımlanmaktadır. Burada s^2, varyansı; X_i, veri kümesindeki her bir veriyi; $\overline{X}$, veri kümesindeki verilerin ortalamasını ve n ise toplam veri sayısını göstermektedir [48].

Kovaryans

İki boyuta sahip verilerin oluşturduğu veri kümesinde verilerin boyutları arasındaki ilişkiyi ölçen istatistik terimidir. Öyle ki, bir boyutun başka bir boyutla olan ilişkisini sunabildiği gibi kendisi ile olan ilişkisini de sunabilmektedir, o zaman kovaryans, varyans ile aynı olmaktadır. Dolayısıyla boyutların kendileri ile olan ilişkilerini veren kovaryansları o boyutlar için varyanslar ile aynı olmaktadır. Ayrıca boyutların birbirleriyle olan ilişkisi eğer bu boyutlar bağımsız değil ise kovaryansın değerinin sıfırdan farklı olması ile anlaşılmaktadır. Eğer boyutlar bağımsız ise boyutların yalnızca kendileri ile ilişkisi vardır ve bu da varyans ile ölçülebilmektedir. Kovaryans formülü

$$cov(X,Y) = \frac{\sum_{i=1}^{n} \left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{(n-1)} \tag{E.2}$$

şeklinde tanımlanmaktadır. Burada X_i ve Y_i veri kümesindeki her bir verinin iki boyuttaki veri değerlerini, $\overline{X}$ ve $\overline{Y}$ verilerin ilgili boyutlardaki ortalamasını ifade etmektedir [48].

Ek 1'in devamı

Kovaryans Matris

Kovaryans matris, veri kümesindeki verilerin boyutları arasındaki ilişki yumağını bir bütün olarak sunan matristir. Böylelikle sadece iki boyutlu uzayda değil daha yüksek dereceden boyutlara sahip uzaylarda da ilişki yumağını sunulabilmektedir. Örneğin üç boyutlu bir uzaydaki boyutlar arasındaki ilişki yumağı

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix} \quad \text{(E.3)}$$

matrisi ile verilebilir. Burada C kovaryans matrisini ifade etmektedir [48].

Ek 2 (Matris İşlemler)

Ortogonal Matris

Ortogonal matris kolonları ortogonal vektörlerden oluşan matristir. Ortogonal vektörler ise bulunduğu ortogonal vektör uzayında (ortonormal uzay) kendisi dışındaki tüm vektörlerle skaler çarpımı "0" olan vektörler topluluğudur. Bir diğer anlamda ortogonal matris bu ortogonal vektörler topluluğunu (ortonormal uzayı) temsil eder. Özelliği gereği ortogonal matrisin tersi, transpozesine eşittir, bu da aşağıdaki gibi ifade edilmektedir.

$$A^{-1}=A^{T},\ A=[a_1\ a_2\ \ldots\ a_n] \tag{E.4}$$

Burada A, ortogonal matrisi; a, ortogonal kolon vektörü göstermektedir. Ortogonallik özelliği ise

$$A^{T}A=I \text{ veya}$$

$$(A^{T}A)_{ij}=a_i^{T}a_j \text{ ise } (A^{T}A)_{ij}=a_i^{T}a_j=\begin{cases}1 & i=j\\ 0 & i\neq j\end{cases} \tag{E.5}$$

şeklinde gösterilmektedir. Burada I birim vektördür. Ayrıca $A^{-1}A=I$ olduğundan $A^{-1}=A^{T}$ olduğu da görülebilir [44].

Simetrik Matris

Simetrik matris matrisin köşegenine göre terimlerinin simetrik olduğu matristir. Yani transpozesi kendisine eşit olan ($B=B^{T}$, B simetrik matris ise) matristir. Simetrik matris, bir matrisi kendisinin transpozesi ile çarpılmasıyla (AA^{T} veya $A^{T}A$, A her hangi bir matris) elde edilebilmektedir. Simetrik matrisin transpozesi de simetriktir [44].

Simetrik Matrisin Köşegenleştirilmesi

Simetrik bir matris bir ortogonal vektörler uzayında (ortonormal uzayda) köşegen halde bulunur. Yani simetrik bir matris aşağıdaki gibi köşegen hale getirilebilir.

$$A^{T}BA=D \tag{E.6}$$

Ek 2'nin devamı

Burada *A*, ortogonal bir matris; *B*, simetrik bir matris; *D* ise köşegen bir matristir. Yukarıdaki eşitlik ancak özel bir ortogonal matris kullanılması durumunda köşegen matris verir. Eğer o özel ortogonal matris yerine başka bir ortogonal matris kulanılırsa sonuçta köşegen matris yerine köşegen olmayan bir matris elde edilir. O özel ortogonal matris *B* matrisinin özvektörlerinin kolon olduğu matristir. Yukarıdaki eşitliğin doğru olduğu yani *A* matrisinin *B* matrisinin özvektörlerin oluşturduğu ortonormal uzay olduğunu düşünürsek *B* matrisini eğitliğin solunda yalnız bırakacak şekilde matris işlemler uygulanırsa

$$B= ADA^T \tag{E.7}$$

elde edilir. Bu form ise simetrik matrislerin ortogonal ve köşegen matrislerle de ifade edilebileceğini gösterir. Dolayısıyla köşegenleştirme işlemi ancak matrisin kendi özvektörlerinden oluşan bir ortogonal matris ile yapılırsa mümkün olabilir[44].

Karakteristik Matris Dönüşümü

Matris dönüşümü, bir matrisin tanımlı olduğu uzaydan başka bir uzaya taşınması şeklinde tanımlanabilir. Bu amaçla bir dönüşüm matrisi kullanılır. Kullanılan dönüşüm matrisinin ortonormal olması dönüştürülecek matrisi ortonomal bir uzaya taşır. Bu dönüşüm

$$Y=A^TX \tag{E.8}$$

şeklinde ifade edilir. Burada *X*, n x m boyutunda ortonormal uzaya taşınacak matrisi; A^T, n x n boyutunda ortonormal uzayı (dönüşüm matrisi); *Y* ise n x m boyutunda ortonormal uzaya taşınmış *X* matrisini gösterir. Eğer dönüşüm matrisi, dönüştürülecek matrisin karakteristik özelliklerini de ayıştıran bir dönüşüm matrisi ise (yani o dönüşüm matrisi her bir satırında dönüştürülecek matrisin her bir karakterini ayrıklaştırıyor ise) bu dönüşüm sonucunda elde edilen matris sadece bir uzaydan başka bir uzaya dönüştürülmüş olmaz ayrıca karakterlerine de ayrıklaştırılmış olur. Bundan dolayı bu dönüşüm matrisi

Ek 2'nin devamı

"karakteristik dönüşüm matrisi" olarak adlandırılabilir. Ayrıca bu dönüşümle ortaya çıkan *Y* matrisinin her bir satırı ilgili karakterin *X* matrisine ait değişimini verir. Öyle ki,

$$B=XX^T \tag{E.9}$$

olsun. Burada *X* n x m boyutlarında bir matris; *B* ise n x n boyutlarında bir simetrik matristir. Ayrıca

$$D = A^T BA,\ A=\sum_{i=1}^{n} a_i ,\ D = d_i \delta_{ij} \tag{E.10}$$

olsun. Burada *A*, nxn boyutunda *B* matrisinin özvektörler matrisi; a_i, özvektörler matrisinin her bir kolon vektörü (yani ilgili özdeğere karşılık gelen özvektörü); *D* özdeğerleri içeren köşegen matrisi; d_i ise ilgili a_i özvektörüne karşılık gelen özdeğeri; δ_{ij} , Kronecker delta'yı göstermektedir. O zaman

$$d_i \delta_{ij} = a_i^T B a_j = a_i^T (XX^T) a_j = (X^T a_i)^T (X^T a_j) = (X^T a_i)\cdot(X^T a_j) = (a_i^T X)\cdot(a_j^T X) \tag{E.11}$$

şeklinde açılım yapılırsa

$$(a_i^T X)\cdot(a_j^T X) = d_i \delta_{ij} \tag{E.11}$$

elde edilir. Burada

$$(a_i^T X)\cdot(a_j^T X) = \begin{cases} d_i & i = j \\ 0 & i \neq j \end{cases} \tag{E.12}$$

olduğundan

Ek 2'nin devamı

$$d_i = (a_i^T X) \cdot (a_i^T X) = \left\| a_i^T X \right\|^2, \left\| a_i^T X \right\| = \sqrt{d_i} \tag{E.13}$$

eşitliği bulunur. Burada

$$Y = A^T X, \; y_i = a_i^T X \tag{E.14}$$

olduğu hatırlanırsa (burada y_i satır vektördür.)

$$\left\| y_i \right\| = \left\| a_i^T X \right\| = \sqrt{d_i} \tag{E.15}$$

elde edilir. Bu eşitlikte başta *B* matrisinin kovaryans matris olduğu düşünülürse d_i özdeğerinin varyans olmaktadır ve y_i satır vektörü ilgili karakter (özdeğer) için *X* matrisine ait değişimini vermektedir[44].

Ek 3 (YSA Eğitiminde Kullanılan Fonksiyonlar)

```
public void net()
  {
  double ara=0.0;
  int say=0;
  boolean devam,hatavar;
  double toplamhata=0.0;
  double hata,delta;
  hata=0.0;
  hatavar=false;
  double [] f;
  double ornekhatasi;
  int ite,kackezatlama;
  kackezatlama=0;
 do{
  say++;
  toplamhata=0.0;
  ara=0.0;
  for(int i=0;i<os;i++)
  { ornekhatasi=0.0;
  ara=0.0;
  for(int j=0;j<a;j++)
  {
  ara=0.0;
  for(int  k=0;k<g;k++)
  {
  ara+=G[k][j]*Orn[i][k];
  }
  ara+=s1[j];
  Ac[j]=aktivasyon(ara);
  }
  ara=0.0;
  for(int l=0;l<c;l++)
  {
  ara=0.0;
  for(int  m=0;m<a;m++)
  {
  ara+=A[m][l]*Ac[m];
  }
  ara+=s2[l];
  C[l][i]=aktivasyon(ara);
  }
   hatavar=false;
   for(int g=0;g<c;g++)

  hatavar=(hatavar||((Bc[g][i]-
  C[g][i])!=0));
   if(hatavar){
   cikti(i);
   girdi(i);
   }
   for(int b=0;b<c;b++)
  {
   ornekhatasi+=(Bc[b][i]-
  C[b][i])*(Bc[b][i]-C[b][i]);
   }

  toplamhata=toplamhata+ornekhat
  asi/2;
  }
 ite=1000;
 if(say==ite)
 {
 cikisbul();
 say=0;
 kackezatlama++;
 }
 }while(toplamhata>esik);
 kontrol();
 }

  public double aktivasyon(double
  net )
  {
  net=1/(1+Math.exp(-1*net));
  return net;
  }

  public void girdi(int i)
  {
  double topla;
  double [] deltas;
  deltas= new double[500];
  for(int t=0;t<a;t++)
  {topla=0.0;
  for(int f=0;f<c;f++)
  topla+=(cikti_delta[f])*YA[t][f];
  deltas[t]=Ac[t]*(1-Ac[t])*topla;
  }
  for(int r=0;r<a;r++)
  {
  for(int z=0;z<g;z++)
  {
```

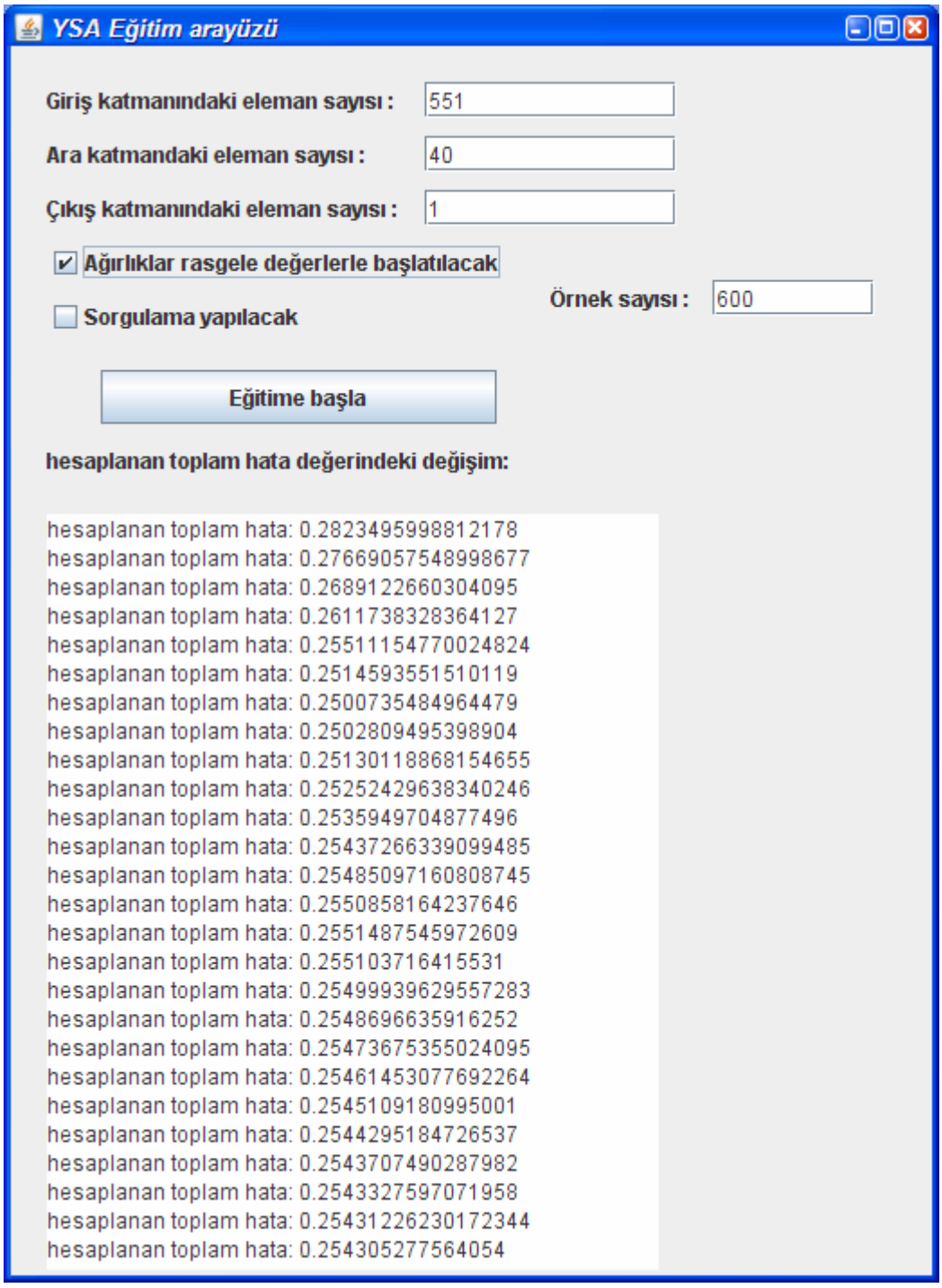
Ek3'ün devamı

```
artiee1[z][r]=ogr*deltas[r]*Orn[i]
[z]+mom*artiee1[z][r];
 }
 }
 for(int r=0;r<a;r++)
 {
 for(int z=0;z<g;z++)
 {
 G[z][r]+=artiee1[z][r];
 }
 }
 for(int r=0;r<a;r++)
 {

artiyy1[r]=ogr*deltas[r]+mom*art
iyy1[r];
 }
 for(int r=0;r<a;r++)
 {
 s1[r]+=artiyy1[r];
 }
 }

 public void cikti(int ko)
 {
 for(int i=0;i<c;i++)
 cikti_delta[i]=C[i][ko]*(1-
C[i][ko])*(Bc[i][ko]-C[i][ko]);
 for(int r=0;r<c;r++)

artiy[r]=ogr*cikti_delta[r]+mom*
artiy[r];
 for(int i=0;i<a;i++)
 for(int h=0;h<c;h++)
 YA[i][h]=A[i][h];
 for(int r=0;r<c;r++)
 s2[r]+=artiy[r];
 for(int j=0;j<a;j++)
 for(int q=0;q<c;q++)
artie[j][q]=ogr*cikti_delta[q]*Ac[
j]+mom*artie[j][q];
 for(int i=0;i<a;i++)
 for(int j=0;j<c;j++)
 {
 A[i][j]+=artie[i][j];
 }
 }

public void cikisbul()
{
double ara;
for(int i=0;i<os;i++)
{
ara=0.0;
for(int j=0;j<a;j++)
{
ara=0.0;
for(int  k=0;k<g;k++)
{
ara+=G[k][j]*Orn[i][k];
}
ara+=s1[j];
Ac[j]=aktivasyon(ara);
}
ara=0.0;
for(int l=0;l<c;l++)
{
ara=0.0;
for(int  m=0;m<a;m++)
{
ara+=A[m][l]*Ac[m];
}
ara+=s2[l];
C[l][i]=aktivasyon(ara);
}
}
for(int m=0;m<os;m++)
{  if((Bc[0][m]-C[0][m])>=0.6 || -
1*(Bc[0][m]-C[0][m])>=0.6)
{
for(int j=m;j<os-1;j++)
for(int k=0;k<g;k++)
Orn[j][k]=Orn[j+1][k];
os--;
}
}
}

public void kontrol()
{
double ara;
for(int i=0;i<os;i++)
{
ara=0.0;
for(int j=0;j<a;j++)
```

Ek3'ün devamı

```
{
ara=0.0;
for(int  k=0;k<g;k++)
{
ara+=G[k][j]*Orn[i][k];
}
ara+=s1[j];
Ac[j]=aktivasyon(ara);
}
ara=0.0;
for(int l=0;l<c;l++)
{
ara=0.0;
for(int  m=0;m<a;m++)
{
ara+=A[m][l]*Ac[m];
}
ara+=s2[l];
C[l][i]=aktivasyon(ara);
}
}
if(sov==1)//sorgulama var
{
double s;
for(int i=0;i<kacsoru;i++)
{
s=0.0;
for(int j=0;j<a;j++)
{
s=0.0;
for(int  k=0;k<g;k++)
{
s+=G[k][j]*Sor[i][k];
}
s+=s1[j];
Ac[j]=aktivasyon(s);
}
s=0.0;
for(int l=0;l<c;l++)
{
s=0.0;
for(int  m=0;m<a;m++)
{
s+=A[m][l]*Ac[m];
}
s+=s2[l];
C[l][i]=aktivasyon(s);
}
}
}
}
 //G[][],giriş sinirlerindeki
//ağırlıklar;
 //A[][],ara sinir ağırlıkları;
 //C[][],bulunan çıkışlar;
 //Orn[][],örnekler;
 //Ac[][],ara sinir çıkışları;
 //Bc[][],beklenen çıkışlardır;
//g, giriş katmanındaki eleman
//sayısı; a, ara katmandaki eleman
//sayısı; c, çıkış katmanındaki
//eleman sayısı, os de örnek
sayısıdır.
```

Ek 4 (YSA Eğitim Arayüzü)



Ek Şekil 1. Yüz tespiti için Ağın Eğitilmesi

Ek 5 (Görüntü Veritabanı 1)

Ek Şekil 2. Görüntü veritabanı 1

Ek 6 (Borda Sayısı Yöntemi)

1700'lü yıllarda Jean Charles de Borda tarafından geliştirilen bir yöntem olan "Borda Sayısı Yöntemi", tercih sırasını tam olarak göz önünde tutan bir puanlama yöntemidir. Bu yönteme göre puanlama; puanlanacak sonuncu adaya, en az puan ("0" değeri) gelecek şekilde ardışık artımsal bir şekilde (0, 1, 2, . . .) ilk adaya kadar yapılmaktadır. Bu puanlama şeklini formülleştirmesi şu şekilde yapılabilir: aday sayısı n ve bir adayın her bir pusuladaki sıralama yerinin m olduğu düşünülürse; her bir aday, bir pusuladan (n-m) puan almaktadır. Bu puanlama yöntemine göre toplamda en çok puan alan aday seçimi kazanmaktadır. Bu yöntem, ayrıca, Yapay Zekâ'da kullanılan puanlamaya dayalı sezgisel değerlendirmeye benzemektedir. Fakat burada puanlama n sayısına bağlı olarak sabittir.

Bu yöntem, üç adaylı ve altı pusulalı bir seçimde aşağıdaki gibi örneklenebilir. Buna göre, bir pusuladaki birinci sıradaki aday "2" puan, ikinci sıradaki aday "1" puan ve üçüncü sıradaki aday "0" puan almaktadır. Ek Tablo 1'de her bir pusuladaki aday sırası ve toplam oy sayısı verilmektedir.

Ek Tablo 1. Bir seçimdeki her bir pusuladaki aday sıralaması ve oy toplamı

	1. Pusula	2. Pusula	3. Pusula	4. Pusula
	A1	A1	A2	A2
	A2	A3	A1	A3
	A3	A2	A3	A1
Toplam Oy	65	78	43	115

Yukarıdaki tabloya göre adayların aldığı toplam puanlar aşağıdaki gibi hesaplanabilir. Buna göre

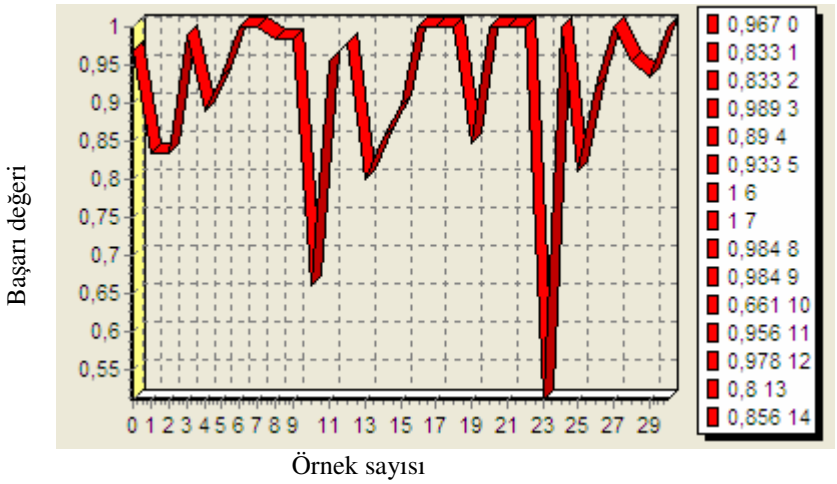
A1 adayı, 65x(2)+78x(2)+43x(1)+115x(0)=459;
A2 adayı, 65x(1)+78x(0)+43x(2)+115x(2)=674;
A3 adayı, 65x(0)+78x(1)+43x(0)+115x(1)=193

Ek 6'nın devamı

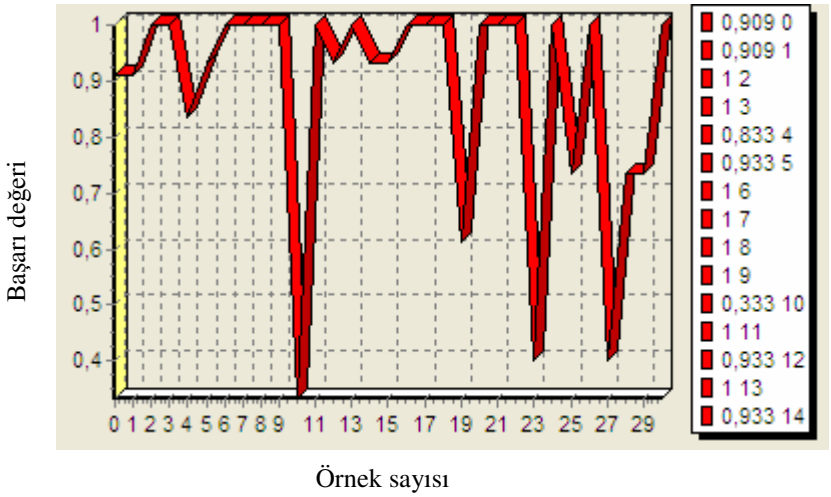
toplam puan almaktadır. Bu puanlama sonucunda en fazla puan alan A_3 adayı kazanan aday olarak belirlenir.

Aslında, seçmenler, ilk birkaç sıradan sonra gelen adayları çok dikkate almayabilirler. Bu durumda, özellikle adayların sayısının artması halinde, önem arz etmeyen tercihler seçimin sonucunu değiştirebilir [58].

Ek 7 (Değerlendirme grafikleri)



Ek Şekil 3. Göz-burun bölgelerinden kıyaslamada *normalize edilmiş Öklit* ile sonuçların yerlerine göre değerlendirme grafiği (% 92 başarı)

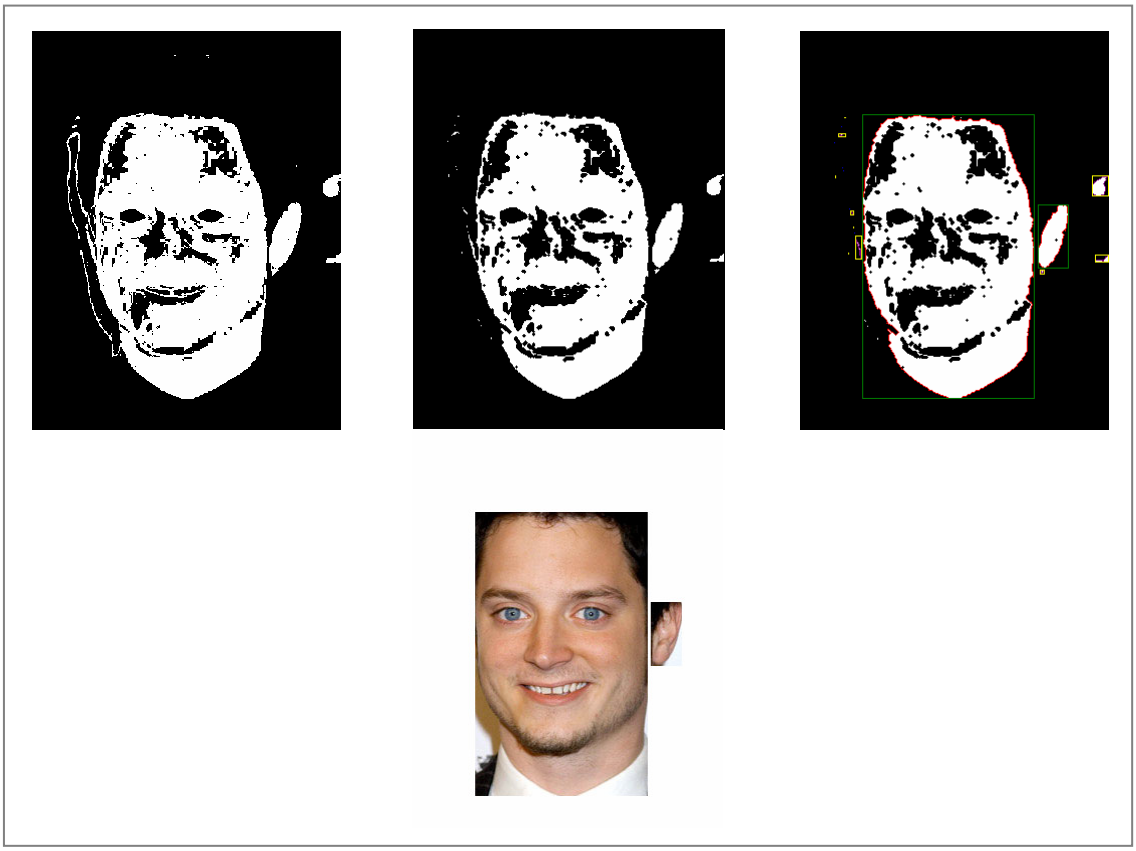


Ek Şekil 4. Göz-burun bölgelerinden kıyaslamada oylama ile sonuçların yerlerine göre değerlendirme grafiği (%88 başarı)

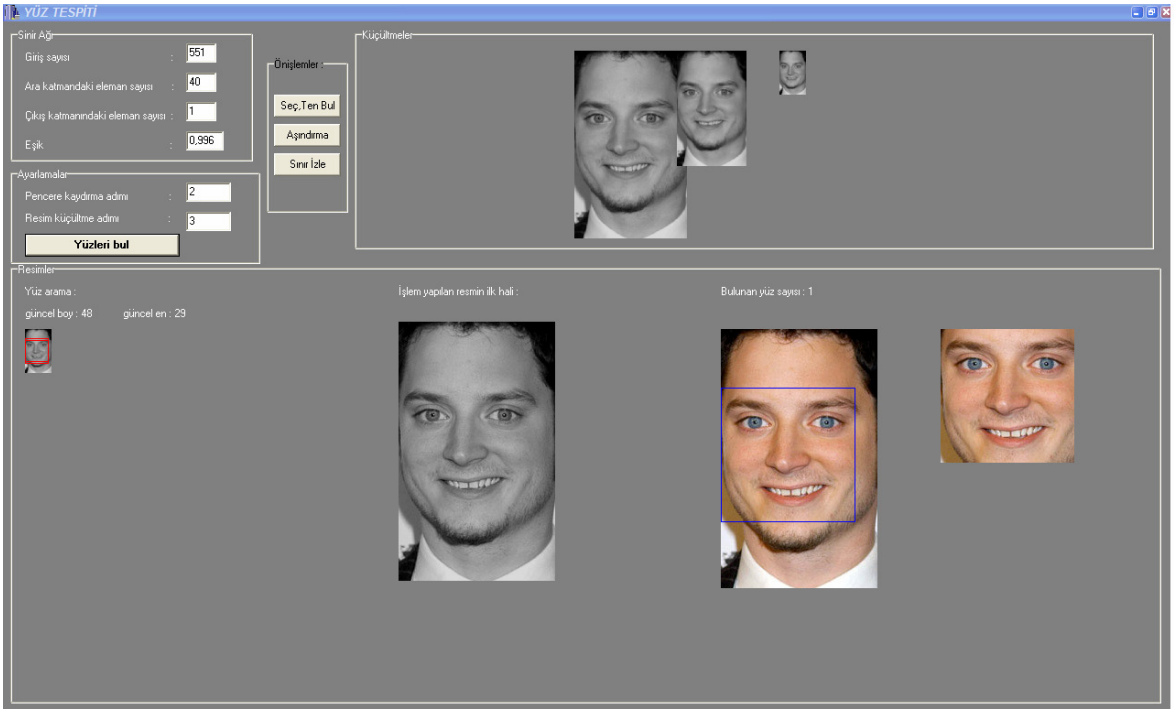
Ek 8 (Görüntü Veritabanı 2)

Ek Şekil 5. Görüntü veritabanı 2

Ek 9 (Örnek Yüz Tespiti)

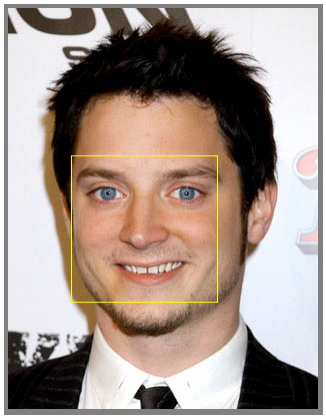


Ek Şekil 6. Örnek üzerinde ten işleme aşamaları



Ek Şekil 7. Örnek üzerinde yüz tespiti aşamaları

Ek 9'un devamı



Ek Şekil 8. Örnek üzerinde yüz tespiti sonucu

ÖZGEÇMİŞ

Yasemin BEKİROĞLU, 1982 Trabzon doğumludur. Karadeniz Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü'nden 2005 yılında üçüncülükle mezun olmuştur. Lisans eğitiminin son yılını Erasmus öğrencisi olarak Danimarka'da Roskilde Üniversitesi Bilgisayar Bilimleri Bölümü'nde tamamlamıştır. 2005 güz yarıyılının sonundan itibaren Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalışmaktadır. Yüksek lisans süresince TÜBİTAK bursu almıştır. İyi derecede İngilizce bilmektedir.