YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİCİK TABANLI İLİŞKİLİ BİLEŞEN ANALİZİYLE EĞİTİCİSİZ YÜZ TANIMA MODELİ

Elektrik ve Elektronik Müh. Bilâl KARADUMAN

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı Haberleşme Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Yrd. Doç. Dr. Lütfiye DURAK

İSTANBUL, 2008

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

KISALTMA LİSTESİ

CMC	Kümülatif Eşleme Karakteristiği (Cumulative Matching Characteristic)
ROC	Alıcı İşletme Karakteristiği (Receiver Operating Charasteristic)
PCA	Temel Bileşen Analizi (Principal Component Analysis)
RCA	İlişkili Bileşen Analizi (Relevant Component Analysis)
LDA	Doğrusal Diskriminant Analizi (Lineer Discriminant Analysis)
ICA	Bağımsız Bileşen Analizi (Independent Component Analysis)
ORL	Olivetti Araştırma Laboratuarı (Olivetti Research Laboratory)
FERET	Yüz Tanıma Teknolojisi Programı (The Face Recognition Technology)
NIST	Amerika Ulusal Teknoloji ve Standartlar Enstitüsü (National Institute of Standards and Technology)

ÖNSÖZ

Son yıllarda güvenlik, dijital ortamda iz sürme ve kimliklendirme gibi sektörlere olan ilgi gelişen teknoloji ile birlikte bu alanın daha da ilgi çekici hale gelmesini sağlamıştır. Bu alanda en çok yatırım yapılan dallardan biri de yüz tanımadır. Yüz tanıma üzerine pek çok algoritma ve metot geliştirilmiştir. Ancak artarak devam eden yatırımlar ve bu sistemlere olan talep bu sistemlerin başarımının henüz istenilen seviyede olmadığını göstermektedir. Bu tezde, yüz tanıma problemi için birim filtre bankaları aracılığıyla türetilen altgörüntüler yan bilgi olarak kullanılmış ve ilişkili bileşen analizi kullanılarak sınıflandırma başarısı yüksek, eğiticisiz yeni bir yüz tanıma modeli geliştirilmiştir.

Bu çalışmanın her noktasında emeğini esirgemeyen sevgili hocam Yrd. Doç. Dr. Lütfiye Durak'a ve Arş. Gör. Ahmet Serbes'e gösterdikleri sonsuz destek ve sabırdan dolayı teşekkürü bir borç bilirim.

Özellikle yüksek lisans eğitimimin ve tez yazım sürecimin her aşamasında her an desteklerini yanında hissettiğim sevgili aileme ve beni yalnız bırakmayan dostlarıma da çok teşekkür ederim.

ÖZET

Bu tezde, yüz tanıma problemi için birim filtre bankaları aracılığıyla türetilen altgörüntüler pozitif yan bilgi olarak kullanılmış ve ilişkili bileşen analizi (RCA) kullanılarak eğiticisiz yeni bir yüz tanıma modeli geliştirilmiştir. Bu geliştirilen model kosinüs, korelasyon, kareli öklit, manhattan ve mahalonobis uzaklık metrikleri kullanılarak ORL ve AR veritabanları üzerinde test edilmiş ve yapılan optimizasyonlarla hem sınıflandırma doğruluğunun yükseltilmesi, hem de çalışma hızının ve performansının azami seviyeye çıkarılması sağlanmıştır.

Altörnekleme ile bir görüntüden dört adet altgörüntü oluşturularak kullanılabilecek özvektör sayısı artırılmıştır. Kullanılabilecek özvektör sayısının artmasıyla birlikte sınıflandırma başarı oranında da belirgin bir şekilde artış gözlenmiştir. Klasik temel bileşen analizinde (PCA'de) en iyi doğru ve yanlış sınıflandırma oranları küçük bir özdeğer aralığında bulunurken, altörnekleme ile özdeğer aralığının geniş bir aralıkta düzgün bir şekilde yayılması sağlanmıştır.

Önerilen bu yeni modelde klasik PCA'i takiben RCA kullanılmış ve ilgisiz bileşenler yok edilerek en iyi doğru sınıflandırma oranında klasik PCA metotuna kıyasla % 12.69'a varan bir yükselme elde edilmiştir.

ABSTRACT

In this thesis, a new unsupervised face recognition model has been proposed in which the face images are transformed through the identity filter banks and the new subimages derived from original image are used as positive side-information in Relevant Component Analysis (RCA). This model has ran over ORL, and AR face databases by using Cosinus, Correlation based, Squared Euclid, Manhattan and Mahalonobis distance metrics and a significant classification improvement is observed.

By subsampling with identity filter banks four new subimages of a quarter size of the original image are obtained and the number of useful eigenvectors are increased which will help to improve the accurate classification rates. While in classical techniques the best accurate recognition rate and incorrect recognition rate are standing in a small eigenvalue space; with the help of big amount of useful eigenvectors, this space is extended to a large scale of surface where the data can be properly spreaded.

In this new model RCA, followed after PCA, has leaded us to reduce the weight of irrelevant components in projection space and improve the accurate classification performance up to 12.69 %.

1. GİRİŞ

Son yıllarda görüntü işleme ve tanıma alanına özel sektör ve enstitüler tarafından olan ilgi daha da artmış ve bu alanda sayısız araştırma yapılmıştır [1] [2]. Araştırmalar sonucu Eyematic [3] ve Identix [4] gibi pek çok ticari yüz tanıma uygulaması geliştirilmiş ve kullanılmaya başlanmıştır [5]. Görüntü tanıma problemlerinin başında yüz tanıma, desen tanıma, biometrik özelliklerin tanınması, görüntü ve iki boyutlu işaretlerin sınıflandırılması bulunmaktadır [6]. Öte yandan güvenlik, dijital ortamda iz sürme ve kimliklendirme gibi sektörlerin bu teknolojiye olan ihtiyacı bu alanın daha da ilgi çekici hale gelmesini sağlamıştır [7].

Biometrik unsurlar, yani insanların birbirlerinden ayırt edilmelerini sağlayacak biyolojik nitelikler içinde yüz tanıma en etkili yöntemlerden biridir. Çeşitli uygulamalarda, uygulamanın niteliğine ve ihtiyacına gore bazı biyometrik unsurlar diğerlerine gore ön plana çıkmaktadır. Bu uygulamalar genelde doğruluk, maliyet ve hassasiyet bazında değerlendirilirler [8]. Mevcut biometrik tanıma sistemleri arasında yapılan bir araştırmada yüz tanımanın diğerlerine göre daha iyi sonuç verdiği ortaya çıkmıştır [9].

1.1 Yüz Tanıma Senaryoları

Yüz tanıma senaryoları üç genel başlık altında değerlendirilebilir [5]:

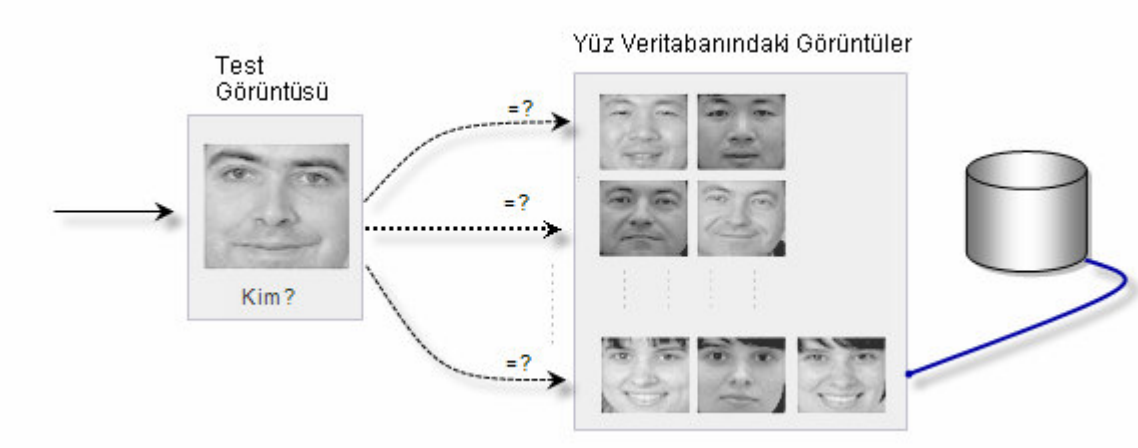
1. Yüz doğrulama
2. Yüz tanıma
3. İzleme Listesi

Yüz doğrulama testi sisteme girdi olarak verilen yüz görüntüsüyle iddia edilen yüz görüntüsünün aynı kişilere ait olup olmadığını kontrol eden birebir eşleme metotudur. Bu testin performansı doğru tanıma oranı ile hatalı tanıma oranının aynı grafik üzerinde üstüste

getirilmesiyle elde edilir. Bu grafik genelde Alıcı İşletme Karakteristiği (ROC) eğrisi olarak bilinir [5].

Yüz tanıma yönteminde, sisteme girdi olarak verilen bir yüz görüntüsü, yüz veritabanındaki fotoğraflarla karşılaştırılarak benzeştirilmeye çalışılır (Şekil 1.3). Girdi olan fotoğraf hangi sınıftaki fotoğrafa en yakınsa, o fotoğrafın o sınıfa ait olduğu varsayılır. Tanıma prosedürü genelde *"kapalı test" (close-universe)* olarak bilinir. Yani test görüntüsünün veritabanındaki sınıflardan kesinlikle biriyle eşleşmesi gerektiği varsayılır. Test görüntüsüne ait özniteliklerle veritabanındaki diğer yüz görüntülerine ait öznitelikler birbirleriyle kıyaslanır ve benzerlik oranına göre her görüntüyle yapılan kıyaslamadan bir skor elde edilir. Bu benzeşme skorları yukarıdan aşağıya sıralanarak en yüksek skoru elde eden fotoğrafların hangi sınıflara ait olduğu yüzdelik bir oran olarak hesaplanır ve test görüntüsünün en yüksek orana sahip olan sınıfa ait olduğu söylenir. Bu hesap Kümülatif Eşleme Karakteristiği (CMC) olarak bilinir [10].

İzleme listesi metotunda**,** test yüz görüntüsü, mevcut bir yüz veritabanındaki insanlara ait diğer görüntülerle karşılaştırılır. Bir önceki metotta olduğu gibi kıyaslama sonuçları en yüksek değerden daha düşük değerlere doğru sıralanarak test görüntüsünün hangi sınıfa ait olduğu kestirilmeye çalışılır. Bir önceki metottan farklı olarak girdi olarak verilen test görüntüsü kıyas için kullanılan yüz veritabanında yer almayabilir. Sistemin bunun ayrımına varabilmesi için deneysel yöntemlerle elde edilecek bir eşik seviyesinin belirlenmesi gerekir. Buna göre eğer en yüksek kıyaslama skoru eşik seviyesini üzerinde olursa sistem uyarı verir. Bu yöntemin performansı, sistemin doğru verdiği uyarı sayısı ile yanlış verdiği uyarı sayısının karşılaştırılmasıyla anlaşılır. Bu oran "bulma ve tanıma oranı" (detection and identification rate) olarak bilinir [5].



Şekil 1.1 Tipik bir yüz tanıma senaryosu

2. YÜZ TANIMA METOTLARI

İnsan yüzü pek çok iç ve dış ortam koşullarının etkisiyle farklı görünümlere bürünebilir. İnsan yüzünün farklı görüntü vermesine neden olan genel etkenler [5]

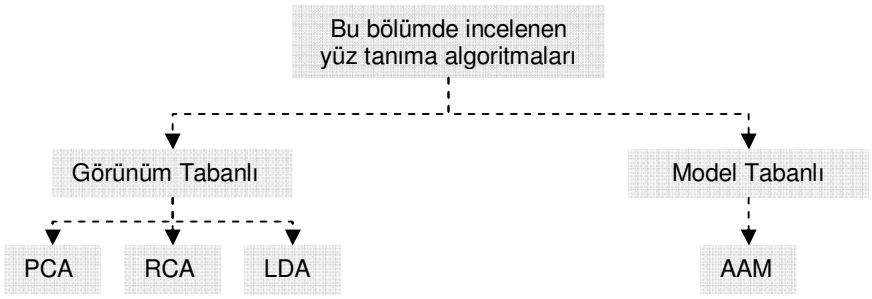
- Kişinin verdiği poz, yani yüzün görünüm açısı,
- Yaşlanma [11],
- Yüzün aldığı ışık oranı, aydınlık (iç ortam, dış ortam) [12] (Şekil 2.1),
- Mimikler,
- Yüzü kapatan aksesuarlar (güneş gözlüğü, eşarp vs.)

olarak düşünülebilir.

Şekil 2.1 ORL veritabanından alınmış bir yüz görüntüsünün farklı ışık kaynakları altındaki değişimi: (a) Soldaki resimde ışık kaynağı yüzü aydınlatacak şekilde ön-üst tarafta yer alırken; (b) Sağdaki resimde ışık kaynağı sağ-üst-arka tarafta yer aldığı için görüntünün karanlık çıkmasına neden olmuştur.

Şekil 2.1'de insan yüz görüntüsünün farklı ışık kaynakları altında ne derece değiştiği gösterilmektedir. [13]'de ışık kaynağının şiddeti ve açısının insan yüzü üzerinde neden olduğu değişikliğin genelde aynı insana ait farklı resimler arasındaki değişimden daha fazla olduğu ortaya koyulmuştur. Bununla birlikte ortam aydınlığının değişiminden etkilenmeyecek çeşitli metotlar da geliştirilmiştir [14].

Bu bölümün devamında görüntü ve model tabanlı algoritmalardan en önemlileri detaylı olarak incelenecektir. Şekil 2.2'de gösterildiği gibi, literatürde genellikle görünüm temelli veya model-tabanlı algoritmalar yoğun bir biçimde kullanılmaktadır [15].



Şekil 2.2 Bölüm 2'de incelenen yüz tanıma algoritmaları

2.1 Görünüm Tabanlı Yüz Tanıma

Bu bölümde 3 temel görünüm tabanlı sınıflandırma algoritması

- Temel Bileşen Analizi (PCA) [16] [17] [18] [19],
- Doğrusal Diskriminant Analizi (LDA) [12][20][21][22],
- İlişkili Bileşen Analizi (RCA) [23] [24] [25] [26]

ele alınacaktır.

Pek çok görünüm tabanlı yaklaşım görüntü vektörlerinin vektör uzayındaki dağılımını analiz etmek için istatistiksel teknikleri kullanır. Girdi olarak verilen bir test görüntüsüyle depo edilmiş görüntü prototipleri arasındaki benzerlik resimlerin öznitelik vektörleri uzayına taşınarak karşılaştırılmasıyla elde edilir.

Yüz görüntüleri yüksek boyutlu bir vektör uzayında bir vektörle gösterilebilir. Örneğin, $p \times q$ ebatlarında 2B bir yüz resmi her bir pikselin leksikografik sıralamasıyla (resmin her bir satır veya sütununun birleştirilmesiyle) vektörel olarak $x \in R^{pq}$ şeklinde ifade edilebilir [5].

$$X = (x_1, x_2, ..., x_i, ..., x_N), \tag{2.1}$$

Buna göre, $n \times N$ eğitim kümesi matrisinin elemanlarıdır. x_i, n boyutlu yüz vektörü ve $n = p \times q$, bir yüz görüntüsündeki piksellerin toplamıdır. *N*, veritabanındaki toplam yüz görüntüsü sayısıdır.

Her bir algoritma kullandıkları istatistiksel hesaplama yöntemlerine göre yüz vektörü uzayını farklı şekillerde ve boyutlarda oluştururlar. Yüz görüntü vektörleri, izdüşümleri (projeksiyon) alınarak temel vektörlere dönüştürülürler. Bu temel vektörler yüz görüntülerinin öznitelikleri olarak da bilinirler.

Test görüntüsü ile eğitim kümesindeki görüntülerin eşleşme skorları, özniteliklerin arasındaki uzaklıklar hesaplanarak bulunur. Bu skor ne kadar yüksek olursa başarım oranı o kadar yüksek olur.

Görünüm temelli PCA, LDA ve RCA algoritmaları yüz görüntüleri arasındaki uzaklığı maksimize edecek doğrusal dönüşüm matrislerini bulmaya çalışırlar. Bu dönüşüm matrisi W olmak üzere,

$$Y = W^T X \qquad (2.2)$$

Y, $d \times N$ öznitelik vektör matrisi; *d* öznitelik vektörünün boyutudur.

$$(d << N) \qquad (2.3)$$

2.1.1 Temel Bileşen Analizi

Özyüzler (eigenfaces) algoritması boyut azaltarak yüz görüntülerinin görüntü uzayında nasıl dağıldığını bulabilmek için PCA kullanır [16]. Yüksek boyutlu görüntülerde öznitelik vektörlerini bulup çıkarmak zor olduğundan PCA bu noktada fazla bilgi kaybına neden olmadan boyut azaltmak için oldukça yardımcı olur. PCA bu yapısından ötürü görüntü sıkıştırma uygulamalarında da kullanılır.

PCA Algoritması:

S, *m* adet elemana ait bir veri kümesi olsun (Eşitlik 2.4).

$$S = \{\Gamma_1, \Gamma_2, \Gamma_3, ..., \Gamma_m \} \tag{2.4}$$

S kümesine PCA uygulamak için,

1. Önişlem olarak kümeyi oluşturan verilerin ortalaması (Eşitlik 2.5)

$$\Psi = \frac{1}{m}\sum_{n=1}^{m}\Gamma_n \tag{2.5}$$

ve S kümesi ile ortalama görüntü ψ arasındaki fark kümesi hesaplanır (Eşitlik 2.6).

$$\Phi_i = \Gamma_i - \Psi \tag{2.6}$$

2. Kovaryans matrisi *C* hesaplanır (Eşitlik 2.7).

$$C = \frac{1}{m}\sum_{n=1}^{m}\Phi_n\Phi_n^T = AA^T, \quad A = \{\Phi_1, \Phi_2, \Phi_3, ..., \Phi_n\} \tag{2.7}$$

3. Kovaryans matrisi *C* kullanılarak özdeğerler ve özvektörler v_l, u_l aşağıdaki gibi hesaplanır.

$$u_l = \sum_{k=1}^{m} v_{lk}\Phi_k \qquad l = 1, ..., m \tag{2.8}$$

4. Özdeğerler en yüksek değerlikli olandan düşük olana göre yeniden sıralanır. En yüksek *R* özdeğere karşılık gelen özvektörler seçilir ve *S* veri matrisi bu özvektörler matrisi üzerine

izdüşürülerek temel bileşen analizi ile boyut azaltma işlemi gerçekleştirilmiş olunur. *U*, yüksek özdeğerlere gelen özvektörler matrisini (Eşitlik 2.9) ve *S* veri matrisini (Eşitlik 2.4) göstermek üzere *W* dönüşüm matrisi Eşitlik 2.10'daki gibi hesaplanır.

$$U = [u_1, u_2, ..., u_r] \tag{2.9}$$

$$W = U^T S \tag{2.10}$$

PCA'i daha iyi anlatabilmek için Tablo 2.1a'da örnek olarak bir veri kümesi verilmiştir. Yukarıda verilen PCA algoritması kullanılarak bu veri kümesi üzerinden işlemlerin nasıl yapıldığı 5 adımda basitçe açıklanacaktır.

Adım 1: PCA'in daha iyi çalışabilmesi için, her veri kümesinden, o kümenin kendi ortalama değerini çıkarmamız gerekir (normalizasyon). Yani aşağıdaki her x değerinden $\bar{x}$ ortalama değerininin; her y değerinden de $\bar{y}$ ortalama değerininin çıkarılması gerekir. Burdaki amaç ortalama değeri sıfır olan bir küme elde etmektir (Kod 2.1). Sonuç Tablo 2.1b'deki gibidir.

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{10} x_i \quad \textbf{ve} \quad \bar{y} = \frac{1}{N}\sum_{i=1}^{10} y_i \quad \textit{(N, her veri kümesindeki eleman sayısıdır.)} \tag{2.11}$$

```
%Normalizasyon
x=[2.5 0.5 2.2 1.9 3.1 2.3 2.0 1.0 1.5 1.1]; %x data seti
y=[2.4 0.7 2.9 2.2 3.0 2.7 1.6 1.1 1.6 0.9]; %y data seti

X=A-mean(x);
Y=B-mean(y);
```

Kod 2.1 Ortalama değeri sıfır olarak elde etmek için (normalizasyon) yazılan matlab kodu

Tablo 2.1 PCA'de kullanılacak örneğe ait (a) veri kümesi; (b) veri kümesinin ortalama değer hesabı yapıldıktan sonraki değerleri

Veri kümesi :

X	*Y*
3	*2.4*
1	*0.7*
2	*2.9*
2	*2.2*
3	*3*
2	*2.7*
2	*1.6*
1	*1.1*
2	*1.6*
1	*0.9*

(a)

Ortalama değeri sıfırlanmış data:

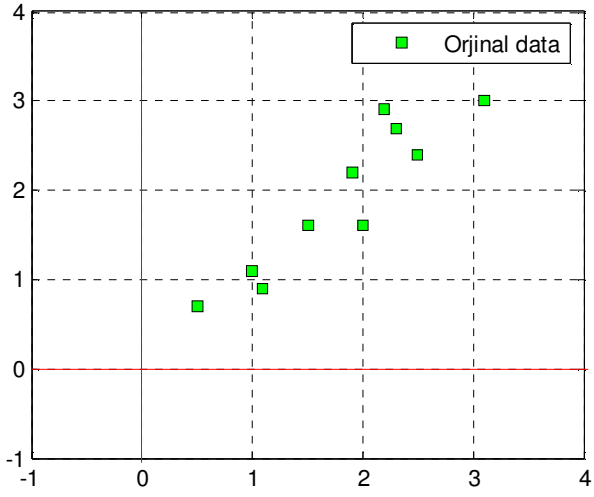
X	*Y*
0.69	*0.49*
-1.31	*-1.21*
0.39	*0.99*
0.09	*0.29*
1.29	*1.09*
0.49	*0.79*
0.19	*-0.31*
-0.81	*-0.81*
-0.31	*-0.31*
-0.71	*-1.01*

(b)

Orjinal verinin izdüşümü Şekil 2.3'te gösterildiği gibidir. (Kod 2.2)

```
%orjinal verinin projeksiyonu
plot(x,y,'rs','LineWidth',1,'MarkerEdgeColor','k','MarkerFaceColor','
g','MarkerSize',5);
grid on;
```

Kod 2.2 Orjinal veriyi kartezyen koordinat düzlemine döken kod



Şekil 2.3 PCA örneğindeki orjinal verinin kartezyen koordinat sistemindeki izdüşümü

Adım 2: Normalize edilmiş verinin kovaryans matrisi *C* hesaplanır. (Kod 2.3)

$$C = \begin{bmatrix} 0.6166 & 0.6154 \\ 0.6154 & 0.7166 \end{bmatrix} \qquad (2.12)$$

```
%Kovaryans matrisi hesapla
covariance=cov(X,Y);
```

Kod 2.3 Kovaryans matrisi hesaplayan kod.

C kovaryans matrisindeki diyagonal olmayan elemanlar pozitif olduğundan *x* ve *y* veri kümesinin birbiriyle doğru orantılı olduğu sonucuna varılabilir.

Adım 3: Kovaryans matrisi kare bir matris olduğundan özvektör (eigenvector) ve özdeğerleri (eigenvalue) hesaplanabilir. Bu hesap oldukça önemlidir. Zira örnekteki verinin biçim için önemli ve kullanılabilir olan yanını ortaya koymaktadır. (Kod 2.4)

```
%Özdeğer ve özvektörleri hesapla
[v d]=eig(covariance) % v= özvektörler, d= özdeğerler
```

Kod 2.4: Özvektör ve özdeğerleri hesaplayan kod.

$$v = \begin{bmatrix} -0.7352 & 0.6779 \\ 0.6779 & 0.7352 \end{bmatrix}, \; d = \begin{bmatrix} 0.0491 \\ 1.2840 \end{bmatrix} \qquad (2.13)$$

Özvektörler *v* ve özdeğerler *d* ile gösterilmiştir. Elde edilen bu özvektörler ile veri kümeleri karakterize edilmeye başlanmış olup her iki veri kümesinin (*x* ve *y*) birbiriyle ne kadar ilişkili olduğu ortaya koyulmaktadır.

Elde edilen özdeğer matrisindeki en yüksek değerli bileşen temel bileşendir. Yani her iki veri kümesi arasındaki en önemli ilişkidir. Genelde izlenilen yöntem özvektörlerin

hesaplanmasının ardından bu değerlerin azalan sırada dizilmesidir. Bu sayede en önemli olandan en az önemli olan bileşene doğru bir sıralama elde edilmiş olur. Bu dizilime göre istenilirse sıfıra yakın veya ihmal edilebilecek daha az önemli olan bileşenler elenerek boyut azaltılmış olunur. Yani *n* boyutlu bir veri kümesinin, *n* adet özvektörü ve *n* adet özdeğeri olacaktır. Bizim için önemli olan *p* adet özdeğer ve özvektörü seçtiğimizde, sonuç kümemiz de *p* boyutuna sahip olacaktır.

Örneğe devam edilirse, sırada öznitelik vektörünü oluşturmak vardır. Öznitelik vektörü, aslında seçilen özvektörlerden oluşturulan yeni vektör matrisidir.

Sadece 2 adet özdeğeri olduğundan, sadece 2 adet tercih seçeneği bulunmaktadır: Öznitelik vektörünü bütün özvektörleri kullanarak oluşturmak ya da sadece küçük ve daha az önemli olan kolonu kullanarak oluşturmak. Bu seçimlerin sonuçları ilerleyen adımlarda açıklanıyor olacaktır.

Adım 4: Bir önceki adımda kullanılacak bileşenler (özvektörleri) seçilip öznitelik vektörü oluşturulduktan sonra seçilen vektör matrisinin tersi (transpozu) alınarak sol tarafta normalize edilmiş veri kümesiyle çarpılır.

$$Z = u \times z \qquad (2.14)$$

u, kolon halinde olan özvektörlerin en önemliden daha öz önemli olana göre sıralandıktan sonra transpoze edilerek satır haline getirilmiş halidir.

z, birinci adımda normalize edilmiş verinin transpoze edilerek satır haline getirilmiş halidir.

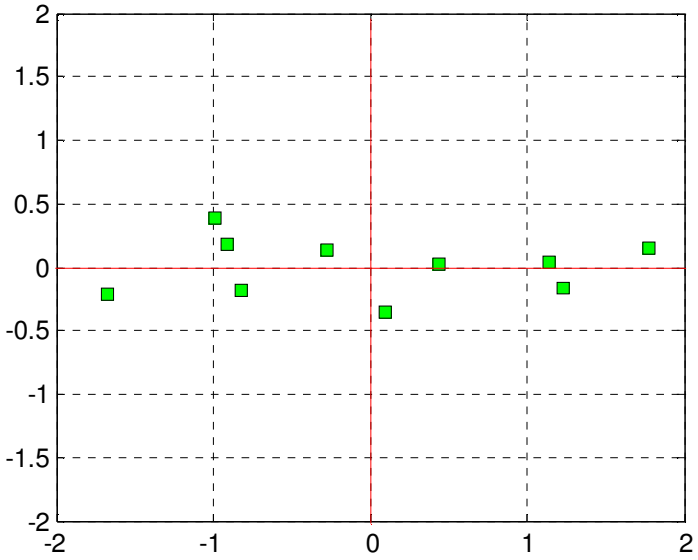
Z, elde edilen yeni veridir.

Son dönüşüm, her iki öznitelik vektörüyle birlikte hesaplandığında sonuç Tablo 2.2'deki, gibi, izdüşümü ise Şekil 2.4'deki gibi olmaktadır

Tablo 2.2 PCA ile iki özvektör kullanılarak dönüştürülmüş veri kümesi değerleri

	X	Y
Dönüştürülmüş veri:	-0.828	-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
	0.438	0.0178
	1.2238	-0.1627

Uygulanabilecek diğer senaryoda ise en büyük özdeğere sahip olan özvektör alınır (yani boyut azaltma uygulanmış olunur) ve aynı hesap yapılır. Bu durumda elde edilen yeni veri kümesi Tablo 2.3'de gösterilmiştir.



Şekil 2.4 PCA ile her iki özvektör kullanılarak elde edilen son veri kümesinin izdüşümü

Tablo 2.3 PCA'de tek özvektör kullanılarak dönüşümü alınmış veri

	X
Dönüştürülmüş veri:	0.828
	1.7776
	-0.9922
	-0.2742
	-1.6758
	-0.9129
	0.0991
	1.1446
	0.438
	1.2238

Adım 5: PCA kullanılan algoritmalarda dönüştürülmüş veriden orjinal veriye geri dönülmesi başlı başına bir sorundur. Eğer öznitelik vektörü hesaplamasında en önemli özdeğerlere sahip olan özvektörler değilde tamamı alınırsa herhangi bir sorun yaşamadan, uygulanan adımlar tersine uygulanarak orjinal veri yeniden elde edilebilir. Ancak, önemli özvektörler alınıp boyut küçültülerek hesaba devam edildiyse, ki genelde böyle yapılır, orjinal verinin tam olarak geri elde edilmesi pek mümkün değildir. Çünkü ihmal edip elenen özvektörlere karşılık gelen veri kaybolmuştur.

$$Z = u \times z \qquad (2.15)$$

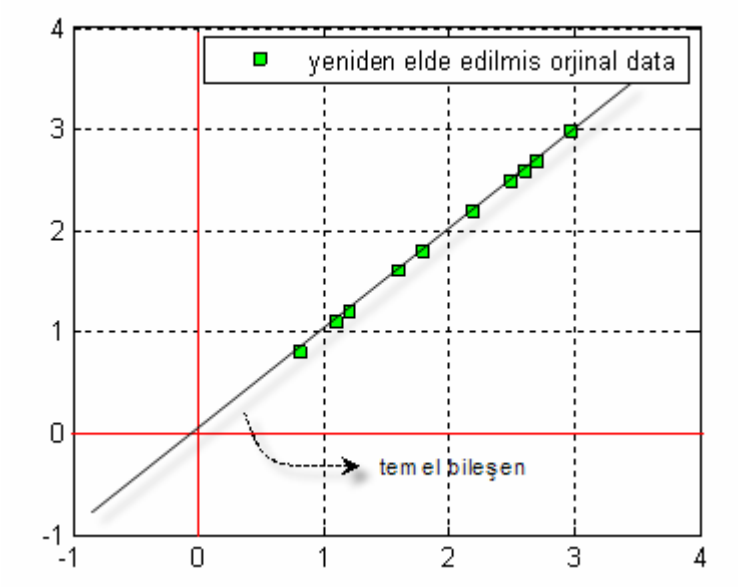
$$z = u^T \times Z \qquad (2.16)$$

Orjinal veriyi tam olarak elde etmek için normalizasyon için çıkarılan ortalama değerinin de geri ilave edilmesi gerekir.

$$Zorj = (u^T \times Z) + m \qquad (2.17)$$

Zorj, orjinal verinin satır formundaki halini, *m* ortalama değeri göstermektedir.

Bu hesap boyut azaltmayla elde edilen vektör kullanılarak yapıldığında geri elde edilen orjinal verinin izdüşümü Şekil 2.5'de gösterildiği gibi olmaktadır.



Şekil 2.5 PCA ile boyut azaltıp orjinal veriyi geri elde ettikten sonraki verinin izdüşümü

Bu noktaya dek basit bir örnek üzerinden gidilerek PCA'in temelinde yatan mantık ve uygulama biçimiyle ilgili genel bir bilgi verilmiştir. Önerilen modelde PCA boyut azaltma ve izdüşüm matrisini oluşturmakta kullanılmaktadır. Geliştirilen bu model klasik PCA ile kıyaslanmış ve başarım sonuçları Bölüm 6 ve Bölüm 7'de verilmiştir.

2.1.2 Doğrusal Diskriminant Analizi

Diskriminant analizi, istatistiksel anlamda nesneleri spesifik özelliklerini baz alarak belli gruplar altında sınıflandırma tekniğidir. Bu teknik pek çok alanda kullanılmakla beraber bu çalışmada yüz tanımayla ilgili olan kısmı üzerinde durulacaktır. Aşağıda Doğrusal Diskriminant Analizi (LDA) üzerine genel bir bilgi verildikten sonra, verilecek örneklerle betimlenip, nasıl uygulandığı nümerik bir analizle açıklanmaya çalışılacaktır

Diskriminant analizinin amacı nesneleri (insanlar, müşteriler, eşyalar vs.), onları tanımlayan özniteliklerini (yaş, gelir, eğitim vs.) baz alarak iki veya daha fazla grup altında

toplayıp, sınıflandırmaktır. Genelde nesneler, onlar üzerinde yapılan ön çalışmalara ve gözlemlere dayanılarak önceden belirlenen gruplara yerleştirilirler.

O halde LDA'daki öncelikli amaç, öznitelik seçimi; daha sonraki amaç ise sınıflandırmadır. Bu bölümün devamında nesnelerin nasıl sınıflandırılacağı üzerinde durulacaktır.

Sınıflandırma söz konusu olduğunda, LDA sınıflar arası saçılma matrisi S_A ile sınıf içi saçılma matrisi S_I arasındaki oranı maksimum yapacak dönüşümü bulmaya çalışır [6] [12] [27]. S_A 'nın büyük olması sınıflar arasındaki mesafenin artmasını ve sınıfların birbirlerinden uzaklaşarak daha rahat seçilebilmesini sağlayacaktır. S_I'nın küçük olması ise sınıfı oluşturan elemanlar arasındaki uzaklığın azaltılmasını ve diğer sınıflara ait elemanlarla örtüşme ihtimalini azaltacaktır.

$$S_I = \sum_{i=1}^{c} \sum_{x_r \in X} (x_r - m_i)(x_r - m_i)^T \tag{2.18}$$

x_r, X matrisinin her bir özvektörü; c maksimum sınıf sayısı ve m_i, her sınıfın ortalamasıdır.

$$S_A = \sum_{j=1}^{c} N_j (m_j - m)(m_j - m)^T \tag{2.19}$$

N_j, her sınıftaki örnek sayısı ve m bütün verinin ortalamasıdır.

Arasında saçılma matrisi S_A ve içinde saçılma matrisi S_I arasındaki oranı maksimum yapacak dönüşüm Eşitlik 2.20'de gösterilmiştir.

$$W = \arg\max \frac{W^T S_A W}{W^T S_I W} \tag{2.20}$$

Buna göre en optimum dönüşüm sınıfları, onları oluşturan verilerin merkezlerini birleştiren doğru üzerine izdüşürmektir. Bu sayede sınıflandırma işlemi çeşitli uzaklık metrikleri kullanılarak rahatlıkla yapılabilir. Bölüm 4'de sıkça kullanılan uzaklık metrikleri üzerinde durulmuştur.

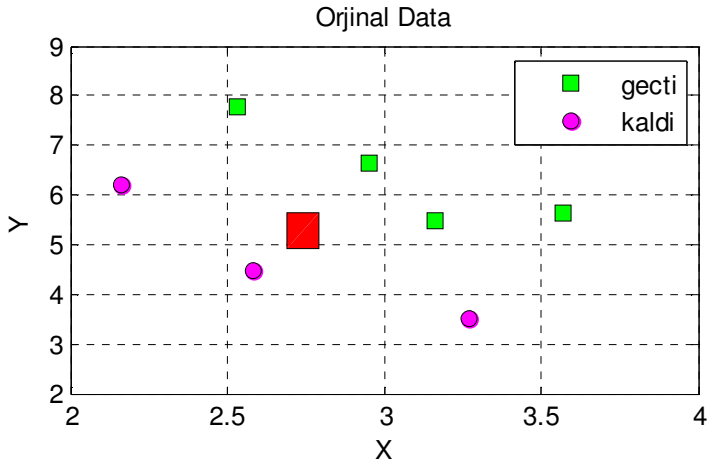
LDA'nın daha iyi anlaşılabilmesini sağlamak için aşağıda analizi yapılmış bir örnek verilmiştir [20].

Örneğin, bir araba lastiği firmasının ürettiği lastiklerin kalitelerini çapına ve kalınlığına göre sınıflandırmak istediği varsayılsın. Tablo 2.4'de kalınlık ve çap değerleri verilmiş lastikler, LDA kullanılarak ilgili sınıflara yerleştirilecektir.

Tablo 2.4 LDA örneğinde kullanılacak orjinal veri

Kalınlık	Çap	Kalite Kontrol Sonucu
2.95	6.63	Geçti
2.53	7.79	Geçti
3.57	5.65	Geçti
3.16	5.47	Geçti
2.58	4.46	Kaldı
2.16	6.22	Kaldı
3.27	3.52	Kaldı

Adım 1: Yukarıdaki 7 lastik verilen öznitelikler baz alınarak koordinat sistemine yerleştirildiğinde, bu verilerin düzlem üzerinde doğrusal olarak yayıldığı görülür. Bu noktada öyle bir çizgi çekilebilmelidir ki, bu iki grup birbirinden net bir şekilde ayrılabilsin. Buradaki asıl problem bu lastiklerin özniteliklerinin izdüşümünün alınıp, gruplar arasındaki arasında dağılma matrisini maksimum yapacak ve aynı zamanda içinde dağılma matrisini de azaltacak çizgiyi bulmaktır.



Şekil 2.6 LDA örneğindeki orjinal verinin izdüşümü

x, bileziklere ait öznitelikler yani bağımsız değişkenlerdir. Her satır (k ile gösterilmiştir) bir lastiği ve her sütun bir özniteliği temsil etmektedir.

y, sınıflandırma grupları yani bağımlı değişkenlerdir. Her satır bir lastiğe karşılık gelmektedir ve onun hangi gruba ait olduğunu göstermektedir.

Örnekteki veriler matris formunda yazılırsa Eşitlik 2.11'deki denklemler elde edilir.

$$x = \begin{bmatrix} 2.95 & 6.63 \\ 2.53 & 7.79 \\ 3.57 & 5.65 \\ 3.16 & 5.47 \\ 2.58 & 4.47 \\ 2.16 & 6.22 \\ 3.27 & 3.52 \end{bmatrix} \quad ve \quad y = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} \qquad (2.21)$$

Burada x_k, k 'ıncı satırdaki veridir. Örnekte,

$$x_3 = \begin{bmatrix} 3.57 & 5.65 \end{bmatrix} ve \; x_7 = \begin{bmatrix} 3.27 & 3.52 \end{bmatrix} \qquad (2.22)$$

g , y matrisindeki grupların sayısıdır. Örnekte, g = *2*'dir ve x_i i 'inci gruba ait özniteliklerdir. Her satır bir nesneyi yani lastiği ve her sütun da bir özniteliği gösterir. O halde;

$$x_1 = \begin{bmatrix} 2.95 & 6.63 \\ 2.53 & 7.79 \\ 3.57 & 5.65 \\ 3.16 & 5.47 \end{bmatrix},\ x_2 = \begin{bmatrix} 2.58 & 4.46 \\ 2.16 & 6.22 \\ 3.27 & 3.52 \end{bmatrix} \qquad (2.23)$$

μ_i , i 'inci gruptaki özniteliklerin ortalaması, yani x_i grubunun averaj ortalamasıdır.

$$\mu_1 = [3.05 \quad 6.38],\ \mu_2 = [2.67 \quad 4.73] \qquad (2.24)$$

μ , global ortalama vektörü yani bütün gruplara ait verilerin averaj ortalamasıdır. Örnekte,

$$\mu = [2.88 \quad 5.67] \qquad (2.25)$$

x_i^0 , normalize edilmiş datadır. Yani x_i grubundan global ortalama vektörü μ 'nün çıkarılmasıyla elde edilen değerdir.

$$x_1^0 = \begin{bmatrix} 0.060 & 0.951 \\ -0.357 & 2.109 \\ 0.679 & -0.025 \\ 0.269 & -0.209 \end{bmatrix},\ x_2^0 = \begin{bmatrix} -0.305 & -1.218 \\ -0.732 & 0.547 \\ 0.386 & -2.155 \end{bmatrix} \qquad (2.26)$$

$$c_i = \frac{\left(x_i^0\right)^T x_i^0}{n_i} \text{ , } i \text{ 'inci gruba ait kovaryans matristir.} \qquad (2.27)$$

$$c_1 = \begin{bmatrix} 0.166 & -0.192 \\ -0.192 & 1.349 \end{bmatrix},\ c_2 = \begin{bmatrix} 0.259 & -0.286 \\ -0.286 & 2.142 \end{bmatrix} \qquad (2.28)$$

$$C(r,s) = \frac{1}{n}\sum_{i=1}^{g} n_i c_i(r,s), \qquad (2.29)$$

$C(r,s)$ global kovaryans matrisidir. Her (r,s) değişkeni için aşağıdaki gibi hesaplanır.

$$\frac{3}{7}0.166+\frac{4}{7}0.259=0.206, \quad (2.30)$$

$$\frac{3}{7}(-0.192)+\frac{4}{7}(-0.286)=-0.233, \quad (2.31)$$

$$\frac{3}{7}0.1349+\frac{4}{7}2.142=1.689 \quad (2.32)$$

Kovaryans matrisi ve kovaryans matrisinin transpozu ise Eşitlik 2.33 ve Eşitlik 2.34'deki gibi bulunur..

$$C=\begin{bmatrix} 0.206 & -0.233 \\ -0.233 & 1.689 \end{bmatrix} \quad (2.33)$$

$$C^{-1}=\begin{bmatrix} 5.745 & 0.791 \\ 0.791 & 0.701 \end{bmatrix} \quad (2.34)$$

p , öncelikli olasılık vektörünü göstermek üzere eğer öncelikli olasılığı bilinmiyor ise bunun gruptaki toplam nesne sayısının, global nesne sayısına bölümü olduğu varsayılabilir. Bu durumda

$$p_i=\frac{n_i}{N} \; ve \; p=\begin{bmatrix} 0.571 \\ 0.429 \end{bmatrix}=\begin{bmatrix} 4/7 \\ 3/7 \end{bmatrix} \quad (2.35)$$

olur ve diskriminant fonksiyonu f_i Eşitlik 2.36'daki gibi hesaplanır

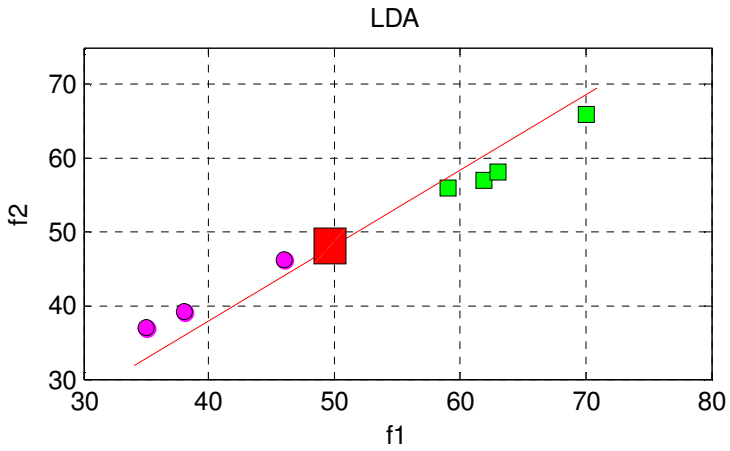
$$f_i=\mu_i C^{-1} x_k^T-\frac{1}{2}\mu_i C^{-1}\mu_i^T+\ln(p_i) \quad (2.36)$$

Bu durumda rastgele seçilen bir k nesnesi maksimum f_i'e sahip olan i'inci gruba yerleştirilmelidir.

Yukarıdaki diskriminant fonksiyonu nesne sınıflandırma ve herhangi bir gruba dahil etme kurallarının matematiksel ifade biçimidir.

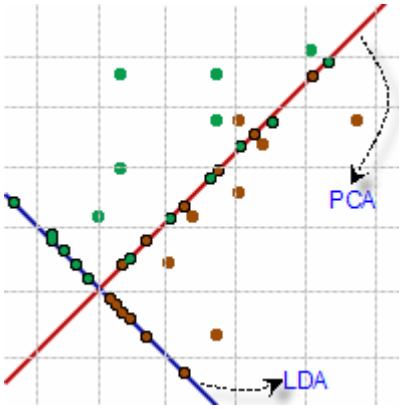
Örnekteki tüm lastikler diskriminant fonksiyonundan (f_1, f_2) gecirilip, değerleri kartezyen koordinat düzlemine yerleştirildiğinde sonuç Şekil 2.7'deki gibi olmaktadır.

Buradaki diskriminant çizgisi $X = \{f_1, f_2\}$ ve $Y = \{f_1, f_2\}$diskriminant fonksiyonuna ait tüm verinin izdüşümünü içerir.



Şekil 2.7 *X* ve *Y* doğrusal diskriminant fonksiyonlarına ait izdüşümler.

Çeşitli yüz tanıma problemlerinde, problemin niteliğine göre bazen LDA, bazen de PCA sınıflandırma metotu olarak tercih edilmektedir. Bu iki metotun da birbirine üstünlük sağladıkları çeşitli problemler olmakla birlikte sınıflandırma anlamında LDA, PCA'e göre daha başarılıdır. Şekil 2.8'de örnek olarak verilen iki sınıfa ait PCA ve LDA izdüşüm grafikleri gösterilmiştir. Buradan anlaşılmaktadır ki, LDA farklı sınıflara ait verilerin birbirinden ayrıştırılmasında PCA'e göre daha iyi sonuç vermektedir.



Şekil 2.8 Örnek olarak verilen iki sınıfa ait verilerin PCA ve LDA kullanılarak elde edilen izdüşümleri

LDA pek çok noktada PCA'e tercih edilse bile bu çalışmada önerilen modelin klasik PCA ile olan karşılaştırılması ortaya koyulduğundan geliştirilen modelde LDA yer almamaktadır. Ancak literatürde oldukça önemli bir yere sahiptir.

2.1.3 İlişkili Bileşen Analizi

İlişkili bileşen analizi (RCA) veri matrisi içindeki değişiklik arz eden istenmeyen kısımların belirlenip yok edilmesi esasına dayanan bir metottur. İstatistikte her bir bileşen için o bileşenin değişkenliğini de işin içine katarak bir uzaklık ölçütü oluşturmak gerekir.Yüksek değişkenliği olan bileşenlerle düşük değişkenliği olan bileşenler için farklı ağırlık belirlemek gerekir. Bu metot doğrusal dönüşüm ile ağırlığı yüksek olan bileşenleri "ilişkili bileşenler"; ağırlığı düşük olan bileşenleri ise "alakasız bileşenler" olarak niteleyerek öznitelik matrisinin yeni bir hal almasını sağlar.

Örneğin, bir yüz tanıma probleminde;

- Yüzün gülen birine ait olması önemliyse
    - Yüksek ağırlıklı bileşen (ilişkili değişken) yüzde gülme ifadesi olması,
    - Düşük ağırlıklı bileşen (ilişkisiz değişken) kişinin kim olduğudur.
- Yüzün kim olduğu önemliyse

- Yüksek ağırlıklı bileşen yüzün kime ait olduğu,
- Düşük ağırlıklı bileşen yüzdeki gülme ifadesidir.

Birbirleriyle ilişkili olan bileşenler *bilgicikler* kullanılarak tahmin edilirler. Bilgicikler, hangi sınıfa ait olduğu bilinmeyen ancak aynı sınıfa ait olduğu bilinen veri kümeleridir. Aşağıda basit bir RCA implementasyonu için ilgili algoritma verilmiştir. Bölüm 6'da bu tezde önerilen model içinde RCA'nın nasıl kullanıldığı açıklanmaktadır.

RCA Algoritması [26]

$X = \{x_i\}_{i=1}^{N}$ data matrisi ve $C_j = \{x_{ji}\}_{i=1}^{n}, (j = 1,..,n)$ n adet bilgicikten oluşan bilgicik matrisi olsun.

1. Bilgicik kümesine ait kovaryans matrisi aşağıdaki gibi hesaplanır:

$$\hat{C} = \frac{1}{N}\sum_{j=1}^{n}\sum_{i=1}^{n_j}(x_{ji} - m_j)(x_{ji} - m_j)^T, \qquad (2.37)$$

m_j, j'inci bilgiciğin ortalamasıdır.

2. Gerekiyorsa, boyut azaltma yoluna gidilebilir.

3. Beyazlatıcı dönüşüm (whitening transform)

$$W = \hat{C}^{-\frac{1}{2}}, \qquad (2.38)$$

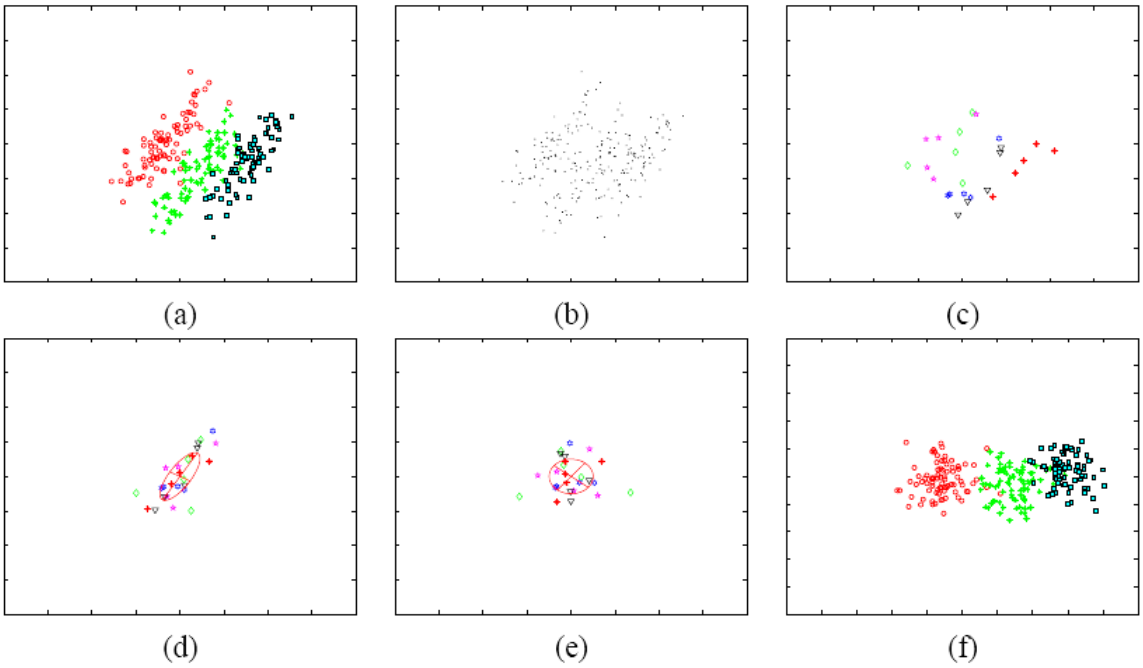
X veri kümesine uygulanır:

$$X_{yeni} = WX \qquad (2.39)$$

Buradaki X, boyut azaltma uygulandıktan sonraki veri kümesini gösterir. Alternatif olarak Mahalanobis uzaklık metrikleri kullanılacaksa $\hat{C}$'nin tersi de kullanılabilir:

$$d(x_1, x_2) = (x_1 - x_2)^T \hat{C}^{-1} (x_1 - x_2) \qquad (2.40)$$

Spesifik olarak açıklanacak olursa, x_1 ve x_2 noktalarının bilinmeyen bir sınıfın üyeleri olduğu biliniyorsa, bunların birbirleriyle pozitif yan bilgilerle ilişkili olduğu söylenir. Eğer x_1 ve x_2 pozitif yan bilgilerle birbirleriyle ilişkiliyseler ve x_2 ile x_3 de yine aynı şekilde birbirleriyle pozitif olarak ilişkili iseler, x_1 , x_2 ve x_3 biraraya gelerek bir bilgicik oluştururlar $\{x_1, x_2, x_3\}$. Genelde bilgicikler tüm veri kümesi üzerinden geçilip pozitif yan bilgilerle birbiriyle ilişkili olan noktalar ortaya çıkarıldıktan sonra oluşturulurlar. Şekil 2.9'da yukarıda anlatılan algoritmayı açıklayan bir örnek verilmiştir [23].



Şekil 2.9 RCA algoritması uygulanmış bir gauss datasına ait örnek. (a) 3 ayrı sınıfa ait işaretlenmiş veri kümesi. (b) Aynı grafiğin işaretler kaldırılmış hali. Sınıf sınırları neredeyse hiç belirgin değil. (c) RCA algoritmasına verilen bilgicik kümesi (aynı renkte olan noktalar bir bilgicik oluştururlar). (d) Ortalanmış bilgicik değerleri ve onların kovaryans matrisi. (e) Beyazlatıcı dönüşümü uygulanmış bilgicik kümesi. (f) RCA dönüşümü uygulandıktan sonraki orjinal veri [23].

RCA, pozitif yan bilgilerin yanı sıra negatif yan bilgiler de kullanılabilir. Negatif sınırlamalı bileşenler işe yarayabilecek bilgi muhteva edebilirler ancak bu bilgiler pozitif

limitli olanlar kadar kullanışlı değildir. Bu görüşü destekleyecek çeşitli deliller de mevcuttur. Örneğin, M adet sınıf içeren bir data kümesi içinden seçilen rastgele iki noktanın aynı sınıfa ait olma olasılığı M^2'dir. Eger bu iki nokta pozitif sınırlar içinde birbirleriyle ilişkiliyseler bu olasılık M 'e iner. Şayet negatif sınırlar içinde birbirleriyle ilişkiliyseler bu ihtimal $M(M-1)$ olacaktır. Bunun dışında, ampirik deneylerin sonuçları pozitif limitler içinde birbiriyle ilişkili olan bileşenlerin negatif sınırlar içinde olanlara göre daha yüksek performans sergilediğini göstermektedir [28] [29].

Yukarıda bahsedilen özelliklerine rağmen negatif yan bilgiler kullanılarak yapılmış uygulamalar da mevcuttur [24][25]. Bölüm 6'da önerilen model eğiticisiz bir algoritma olduğundan sadece pozitif yan bilgiler kullanılmıştır.

RCA algoritmasının ikinci adımında gerektiğinde boyut azaltması uygulanabileceğinden bahsedilmiştir. Yüksek boyutlu görüntü uzaylarında boyut azaltılması durumu algoritmanın hızı ve başarısı açısından neredeyse kaçınılmazdır. Boyut azaltma genelde PCA (temel bileşen analizi) ile en başta başlar.

Önerilen modelde, klasik PCA algoritmasını takiben, boyutu azaltılan matrise RCA uygulanarak ilişkili bileşenler ortaya çıkarılmakta, ilişkisiz bileşenler ise bastırılmaktadır.

2.2 Model Tabanlı Yüz Tanıma

Model tabanlı yüz tanıma tekniği insan yüzüne ait mimik ve hareketleri yakalayıp bu değerlere göre tanıma işlemini gerçekleştiren bir modeldir. Yüzdeki ayırt edici özniteliklerin yüz üzerindeki yerleşim ve birbirlerine göre olan konumuna ve uzaklığına bağlı olarak çalışan algoritmalar bu model altında sınıflandırılabilirler. Kanade, otomatik öznitelik tanıma alanındaki ilk algoritmayı geliştirenlerden biridir [30]. Göz köşeleri, burun, göz gibi biometrik unsurları pozisyonlarına ve birbirlerine olan uzaklıklarına göre yüz veritabanındaki mevcut resimlerdeki karşılık gelen parametrelerle kıyaslayarak test görüntüsünün hangi sınıfa ait olduğunu bulmaya çalışmıştır. [31]'de gösterildiği gibi daha

sonraları yüzün hem şekli hem de dokusunu baz alan ve yüzdeki değişiklikleri öğrenebilen 2B modeller geliştirilmiştir.

Model tabanlı tanıma algoritmaları temelde 3 adımdan oluşurlar:

1- Yüz modelini oluşturma
2- Oluşturulan yüz modelini girdi olarak verilen yüz görüntüsüne uydurma
3- Uydurulan modeli öznitelik vektörü olarak kullanarak veritabanındaki yüz resimleriyle kıyaslayıp benzerlik skorlarına göre sınıflandırma

2.2.1 Aktif Görünüm Modeli

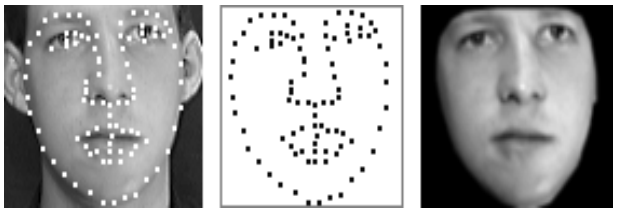
Aktif Görünüm Modeli (AAM), modeldeki şekil değişimlerinin ve görsel degişimlerin her ikisini de baz alan istatistiki bir modeldir. Görüntü ile görüntüden sentezlenen model arasındaki farkı minimize edecek model parametreleri bulunup görüntü üzerine izdüşümü alınır.

2.2.1.1 AAM Oluşturma

AAM, eğitim kümesindeki yüz görüntülerinin ana özniteliklerine ait pozisyonlara yer imleri koyup işaretleyerek model oluşturma esasına dayanır (Şekil 2.10). Yüzün şekli, işaretlenen yer imlerine ait koordinatları içeren vektörlerle gösterilir.

$$s = (x_1, y_1, ..., x_n, y_n)^T \quad . \quad (x_j, y_j), \qquad (2.41)$$

s, görüntünün *j*'inci yer imine karşılık gelen koordinatları işaret eder.



Şekil 2.10 Yer imleri yerleştirilip elde edilen AAM model parametreleriyle yeniden sentezlenen görüntü

Bütün şekil vektörleri normalize edilerek ortak bir koordinat sistemine dönüştürülür. Bu şekil vektörlerine temel bileşen analizi uygulanarak *s* yüz şekil modeli elde edilir,

$$s = \bar{s} + P_s b_s . \qquad (2.42)$$

s, yüz şekil modelini, $\bar{s}$, şekil vektörlerinin ortalamasını, P_s, şekillerdeki değişimlere ait ortagonal değer kümesini, b_s, model parametrelerini gösterir.

2.2.1.2 AAM Uydurma

Girdi olan görüntü ele alınıp model oluşturulmasının ardından model ile görüntü arasındaki eşleşme kalitesinin sağlanması gerekir $\Delta = |\partial I|^2$. ∂I, görüntünün orjinali ile modelleme sonrası sentezlenen görüntü arasındaki yoğunluk fark vektörüdür. *AAM Uydurma,* sisteme girdi olarak verilen görüntüyü tanımlayabilecek en iyi model parametrelerini elde etmeyi amaçlar. Eğitim fazında, AAM yoğunluk fark vektörünü geri besleme olarak kullanıp model parametrelerine ait değerler daha da iyileştirilmeye çalışılır. Bu sayede görüntü ve sentezlenen görüntü arasındaki uyum kalitesi maksimuma çıkarılmış olunur. (Şekil 2.10)

Şekil 2.11 AAM uydurma iterasyonları

2.2.1.3 AAM ile Yüz Tanıma

Bütün eğitim görüntülerine karşılık gelen model parametre vektörleri öznitelik vektörleri olarak kullanılırlar. Yüz tanıma için Doğrusal Diskriminant Analizi kullanılarak diskriminant uzayı oluşturulup, girdi olarak verilen görüntüye AAM uygulanarak karşılık gelen öznitelik vektörleri hesaplanır. Bu öznitelik vektörleri ve veritabanındaki görüntülere ait öznitelik vektörleri diskriminant uzayına yerleştirilip birbirleriyle karşılaştırılırlar ve sınıflandırma (yüzün hangi insana ait yüz görüntüleriyle aynı grupta olduğu) tamamlanır.

3. K-ORTALAMA KÜMELEMESİ

"K-Ortalama Kümelemesi" [32] [33] *n* adet nesneyi (önerilen modelde nesneler, insanlara ait yüz resimleridir) birtakım özellik ve özniteliklerini baz alarak *k* adet grup altında kümeleme algoritmasıdır.

k pozitif bir sayı olmakla birlikte *k*<*n*'dir. Kümeleme işlemi girdi olan veri ile karşılık gelen küme merkezi arasındaki mesafenin karekökünü minimize ederek yapılır (Eşitlik 3.1).

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left(x_j - \mu_i \right)^2 \qquad (3.1)$$

Eşitlik 3.1'de *k* küme sayısı, $S_i = 1,2,.....,k$ ve μ_i, bütün noktaların ortalamasıdır ($x_j \in S_i$).

K-ortalama kümelemesi uygulama şekli oldukça kolaydır. İlk once *K* küme sayısı belirlenip bu kümelere ait küme merkezleri rastgele seçilir. Uygulamanın başında küme merkezinin nasıl seçileceği bilinemeyeceği için rasgele herhangi bir nesneye ait koordinatlar küme merkezleri olarak seçilebilir.

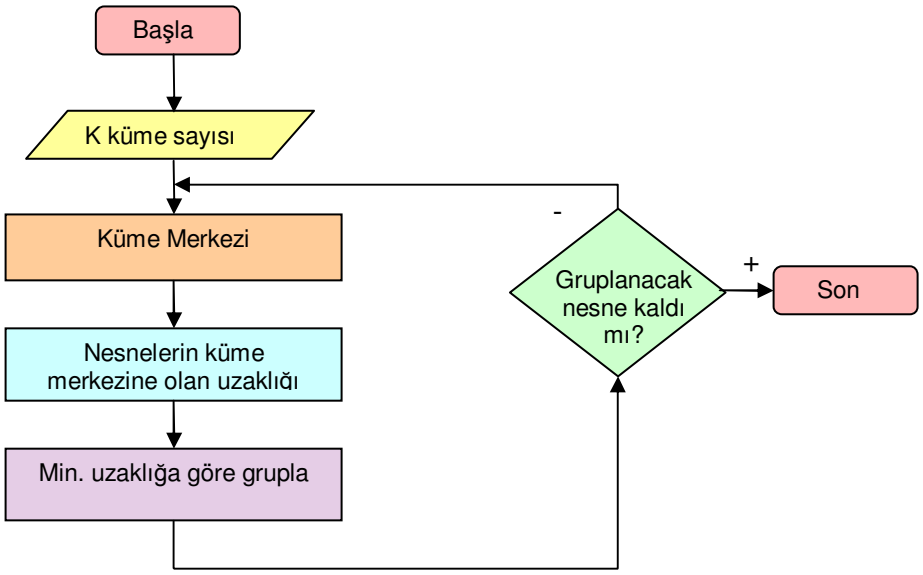
K-ortalama kümelemesi algoritması 3 basit adımdan oluşur ve bütün nesneler kümelere düzgün bir şekilde dağıtılana kadar bu adımlar bir döngü içinde tekrarlanır.

K-Ortalama Kümelemesi Algoritması

1. Küme merkezi belirlenir,

2. Her bir nesnenin küme merkezlerine olan uzaklığı hesaplanır,

3. En kısa mesafe baz alınarak nesneler gruplanır



Şekil 3.1 K-ortalama kümelemesi akış şeması

Tablo 3.1'de örnek olarak 4 adet nesne ve bu nesnelere ait 2'şer adet öznitelik verilmiştir. Buradaki her bir öznitelik ait oldukları nesnelerin kartezyen koordinat düzlemi üzerindeki koordinatlarıdır.

Tablo 3.1 K-ortalama kümeleme örneğinde kullanılacak objeler ve öznitelikleri

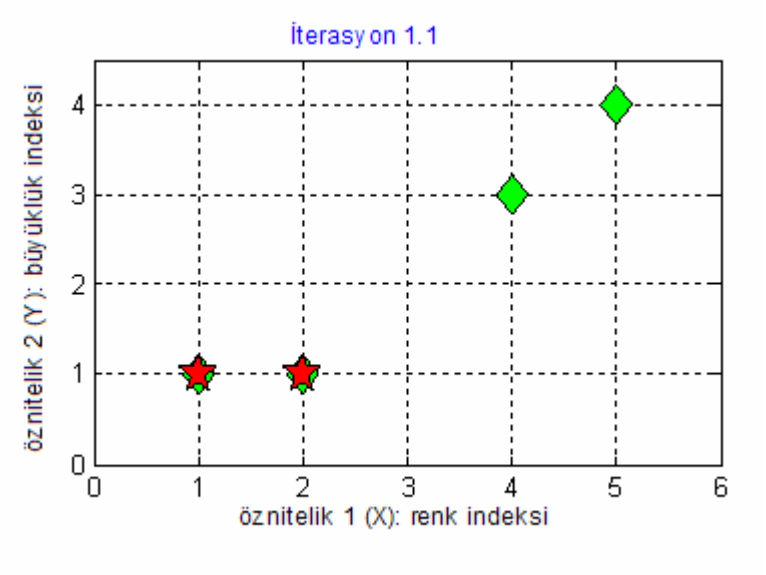
Nesne	***Öznitelik 1 (X): Renk indeksi***	***Öznitelik 2 (Y): Büyüklük indeksi***
A oyuncağı	1	1
B oyuncağı	2	1
C oyuncağı	4	3
D oyuncağı	5	4

Tablo 3.1'de verilen ilaçların iki ayrı oyuncak grubuna ait oldukları bilinmektedir (küme 1 ve küme 2). Bilinmeyen nokta bu oyuncakların hangisinin hangi gruba ait olduklarıdır. Bu problem yukarıda verilen K-ortalama sınıflandırma algoritması ile aşağıdaki gibi çözülebilir.

Tablo 3.1'deki oyuncaklara ait öznitelikler *(x,y)* aslında o nesnelerin öznitelikler uzayındaki koordinatlarını temsil ederler, yani yerlerini gösterirler. Bu nesnelerin koordinat düzlemindeki yerleri aşağıdaki gibidir.

Şekil 3.2 K-ortalama örneğinde verilen nesnelere ait özniteliklerin izdüşümü

İterasyon 1.1 - *İlk küme merkezlerini belirleme:* A ve B oyuncakları küme merkezleri olarak rasgele seçilsin, c_1 ve c_2 küme merkezleridir. O halde $c_1 = (1,1)$ ve $c_2 = (2,1)$ olur.



Şekil 3.3 K-ortalama iterasyon 1.1 sonuçları

İterasyon 1.2 - *Nesnelerin küme merkezlerine olan uzaklığı:* Bütün nesnelerin küme merkezlerine olan uzaklığı Öklit kullanılarak hesaplanır. Bu durumda iterasyon 1.1'de nesnelerin gruplara dağılımı aşağıdaki gibi olur.

$$D^1 = \begin{bmatrix} 0 & 1.0 & 3.6 & 5.0 \\ 1.0 & 0 & 2.8 & 4.3 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & grup-1 \\ c_2 = (2,1) & grup-2 \end{matrix} \qquad (3.2)$$

$$\begin{matrix} A & B & C & D \\ \begin{bmatrix} 1.0 & 2.0 & 4.0 & 5.0 \\ 1.0 & 1.0 & 3.0 & 4.0 \end{bmatrix} & \begin{matrix} X \\ Y \end{matrix} \end{matrix} \qquad (3.3)$$

Uzaklık matrisindeki her sütun bir nesneye karşılık gelmektedir. İlk satır her bir nesnenin ilk küme merkezine olan uzaklığını, ikinci satır ise ikinci küme merkezine olan uzaklığını gösterir. Örneğin; $c = (4,3)$ oyuncağının ilk küme merkezi $c_1 = (1,1)$'e olan uzaklığı Eşitlik 3.4'de ve $c_2 = (2,1)$'e olan uzaklığı ise Eşitlik 3.5'de verilmiştir

$$\sqrt{(4-1)^2+(3-1)^2}=3.61 \qquad (3.4)$$

$$\sqrt{(4-2)^2+(3-1)^2}=2.83 \qquad (3.5)$$

İterasyon 1.3 – *Nesneleri kümeleme:* Her nesne uzaklık matrisine göre hangi küme merkezine daha yakınsa o kümeye yerleştirilir. Nitekim, A oyuncağı 1. gruba; B, C ve D oyuncakları ise 2. gruba girmektedir. O halde grup matrisi aşağıdaki gibi hesaplanır.

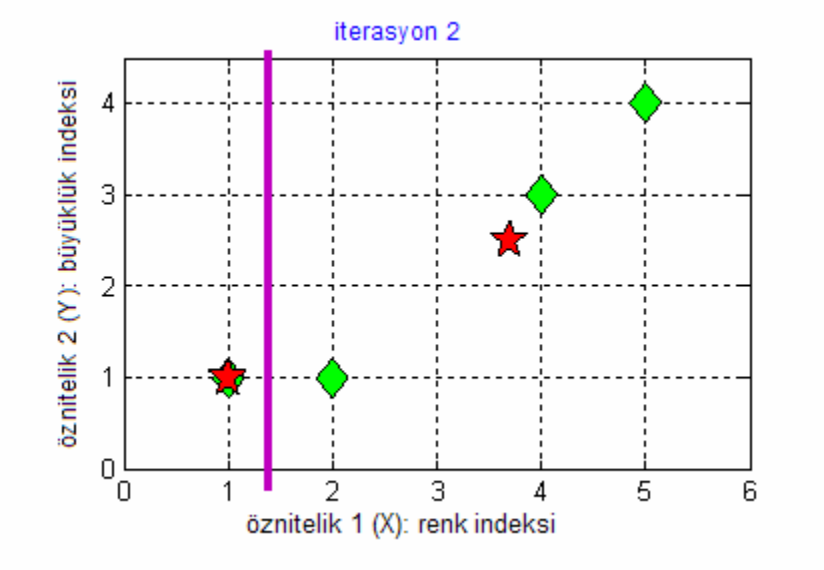
$$G^1=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 1 & 1\end{bmatrix}\begin{matrix}grup-1\\ grup-2\end{matrix} \qquad (3.6)$$

İterasyon 2.1 - *Küme merkezlerini belirleme:* Bütün nesneler gruplara yerleştirildiğine göre yeni küme merkezleri yeniden hesaplanır.

Birinci grubun bir üyesi olduğundan küme merkezi değişmeyecek ve olduğu gibi kalacaktır, $c_1=(1,1)$. Ancak ikinci grubun 3 üyesi olduğundan yeni küme merkezi

$$c_2=\left(\frac{2+4+5}{3},\frac{1+3+4}{3}\right)=\left(\frac{11}{3},\frac{8}{3}\right) \qquad (3.7)$$

olacaktır.



Şekil 3.4 K-ortalama iterasyon 2 sonuçları

İterasyon 2.2 – Nesnelerin *küme merkezlerine olan uzaklığı:* Nesnelerin yeni küme merkezlerine olan uzaklığı ikinci adımda gösterildiği gibi yeniden hesaplanır. O halde iterasyon 2'e ait uzaklık matrisi aşağıdaki gibi olacaktır.

$$D^2 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & grup-1 \\ c_2 = \left(\frac{11}{3}, \frac{8}{3}\right) & grup-2 \end{matrix} \tag{3.8}$$

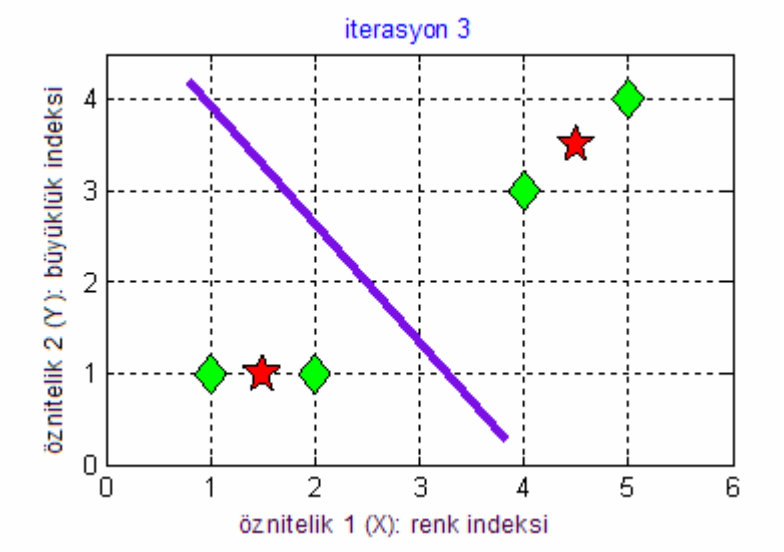
$$\begin{matrix} A & B & C & D \\ \begin{bmatrix} 1.0 & 2.0 & 4.0 & 5.0 \\ 1.0 & 1.0 & 3.0 & 4.0 \end{bmatrix} & \begin{matrix} X \\ Y \end{matrix} \end{matrix} \tag{3.9}$$

İterasyon 2.3 – *Nesneleri kümeleme:* Üçüncü adımda olduğu gibi objeler uzaklık matrisinde gösterilen yeni değerlerine göre yeniden gruplara yerleştirilir. İterasyon 2'e ait grup matrisi aşağıdaki gibi olacaktır.

$$G^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} grup-1 \\ grup-2 \end{matrix} \tag{3.10}$$

İterasyon 3.1 – *Küme merkezlerini belirleme:* Dördüncü adım bir önceki adımdaki yeni küme merkezleri ve grup dağılımları baz alınarak yeniden hesaplanır. Her iki grubun da ikişer adet üyesi olacaktır ve yeni küme merkezleri aşağıdaki gibi hesaplanacaktır.

$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = \left(1\frac{1}{2}, 1\right)$$
$$c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = \left(4\frac{1}{2}, 3\frac{1}{2}\right) \qquad (3.11)$$



Şekil 3.5 K-ortalama iterasyon 3 sonuçları

İterasyon 3.2 – *Objelerin küme merkezlerine olan uzaklığı:* İterasyon 1.2 tekrar edilerek yeni uzaklık matrisi aşağıdaki gibi hesaplanır.

$$D^3 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.3 & 3.54 & 0.71 & 0.71 \end{bmatrix} \begin{matrix} c_1 = (1\frac{1}{2}, 1) & grup-1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & grup-2 \end{matrix} \qquad (3.12)$$

$$\begin{matrix} A & B & C & D \\ \begin{bmatrix} 1.0 & 2.0 & 4.0 & 5.0 \\ 1.0 & 1.0 & 3.0 & 4.0 \end{bmatrix} & \begin{matrix} X \\ Y \end{matrix} \end{matrix}. \qquad (3.13)$$

İterasyon 3.3 – *Nesne kümeleme:* Bütün nesneler, uzaklık matrisi baz alınarak yeniden gruplara yerleştirilir. G^3 grup matrisi aşağıdaki gibi hesaplanır.

$$G^3 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{matrix} grup-1 \\ grup-2 \end{matrix} \qquad (3.14)$$

Son iki iterasyondaki grup matrisleri karşılaştırıldığında $G^3 = G^2$ olduğu ve nesnelerin daha fazla yer değiştiremeyeceği görülüp, iterasyonlar daha fazla tekrar edilmez. Nitekim k-ortalama kümeleme algoritması da zaten kararlı bir seviyeye ulaşmış durumdadır. Sonuç olarak nesnelerin gruplara dağılımı Tablo 3.2'deki gibi olur.

Tablo 3.2 K-ortalama kümelemesi ile nesnelerin gruplara dağılımı

Nesne	**Öznitelik 1 (X): renk indeksi**	**Öznitelik 2 (Y): büyüklük indeksi**	**Grup**
A oyuncağı	1	1	1
B oyuncağı	2	1	1
C oyuncağı	4	3	2
D oyuncağı	5	4	2

```
function y=kMeansCluster(m,k,isRand)
%input matrisi
m = [1 1;2 1;4 3; 5 4];
k = 2;
isRand=0;
[maxRow, maxCol]=size(m);
if maxRow<=k,
    y=[m, 1:maxRow]
else

    % küme merkezinin ilk değeri
    if isRand,
        p = randperm(size(m,1));      % rasgele secim
        for i=1:k
            c(i,:)=m(p(i),:)
        end
    else
        for i=1:k
           c(i,:)=m(i,:)        % artan secim
        end
    end

    temp=zeros(maxRow,1);

    while 1,
        d=DistMatrix(m,c)  % obje-küme merkezi uzakligi hesapla
        [z,g]=min(d,[],2);  % g grup matrisini hesapla
        if g==temp,
            break;          % iterasyonu durdur
        else
            temp=g;         % grup matrisi temp degiskene aktar
        end
        for i=1:k
            f=find(g==i);
            if f
                c(i,:)=mean(m(find(g==i),:),1);
            end
        end
    end

    y=[m,g];
end
```

Kod 3.1 K-Ortalama kümeleme fonksiyonuna ait matlab kodu (kMeansCluster( ))

```
function d=DistMatrix(A,B)
             [hA,wA]=size(A);
             [hB,wB]=size(B);
             if wA ~= wB,  error(' a ve b matrislerinin 2.
elemanlarının büyüklükleri eşit olmalı'); end
             for k=1:wA
                  C{k}= repmat(A(:,k),1,hB);
                  D{k}= repmat(B(:,k),1,hA);
             end
             S=zeros(hA,hB);
             for k=1:wA
                  S=S+(C{k}-D{k}').^2;
             end
             d=sqrt(S)
```

Kod 3.2 DistMatrix( ): Nesneler arasındaki uzaklıkları hesaplayan fonksiyondur ve kMeansCluster( ) fonksiyonu tarafından çağrılır.

4. YÜZ TANIMADA KULLANILAN UZAKLIK METRİKLERİ

PCA ile eğitim için kullanılan yüz görüntülerine ait temel bileşenler bulunur ve bu yüz görüntüleri temel bileşenler uzayına izdüşürülerek öznitelik (özellik) vektörleri elde edilir. Karşılaştırma ve sınıflandırma işlemi bu vektörler arasındaki uzaklıklar hesap edilerek yapılır. Genelde karşılaştırma öznitelik vektörleri arasındaki Öklit uzaklığı kullanılarak hesaplanır. Fakat arada diğer uzaklık metriklerinin de kullanıldığı görülmektedir [10].

Bu bölümün devamında, sıkça kullanılan ve önerilen 12 uzaklık ölçüm metotu gösterilecektir. Bölüm 6'da önerilen modelde aşağıda verilen uzaklık metriklerinden 5 tanesi kullanılmıştır ve bu metriklerin yüz tanıma performansını nasıl etkiledikleri gösterilmiştir.

Uzaklık Metrikleri Algoritması

1. PCA ile elde edilmiş öznitelik uzayındaki özvektörler (Z) aşağıdaki gibi ifade edilsin. Buradaki n öznitelik sayısıdır.

$$Z = (z_1, z_2, ..., z_n)^T = (y_1, y_2, ..., y_n)^T \qquad (4.1)$$

2. Yeni bir yüz görüntüsü bu öznitelik uzayına izdüşürüldüğünde bu görüntüye ait Z_{yeni} öznitelik vektörü elde edilir.

3. Z_{yeni} öznitelik vektörüyle, bilinen yüz görüntülerine ait öznitelik vektörleri arasındaki öklit uzaklıkları hesaplanarak, bu yeni yüz görüntüsünün hangi kişiye ait olduğu söylenebilir.

$$s = \arg\min_i [\varepsilon_i]. \qquad (4.2)$$

4. Bilinmeyen kişilere ait yüz görüntülerinin reddedilmesi için bir eşik değeri τ seçilir. Eğer $\varepsilon_r \geq \tau$ ise izdüşüme ait olan bu yeni görüntünün tanınamadığı söylenebilir.

4.1 Uzaklık Metrikleri

X ve *Y, n* uzunluğunda öznitelik vektörleri olsunlar. Bu vektörler arasındaki uzaklıklar aşağıdaki gibi hesaplanırlar.

a. Minkowski uzaklığı (Lp metrik)

$$d(X,Y) = L_p(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}, p > 0 \qquad (4.3)$$

b. Manhattan uzaklığı (L1 metrik, cityblock uzaklığı)

$$d(X,Y) = L_{p=1}(X,Y) = \sum_{i=1}^{n} |x_i - y_i| \; ; \qquad (4.4)$$

c. Öklit uzaklığı (L2 metrik) [34]

$$d(X,Y) = L_{p=2}(X,Y) = \|X - Y\| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \; ; \qquad (4.5)$$

d. Kareli Öklit uzaklığı (Squared Euclid - SSE)
Toplam kare hatası – sum square error, SSE ve Ortalama kare hatası – mean square error, MSE

$$d(X,Y) = L^2{}_{p=2}(X,Y) = SSE = \|X - Y\|^2 = \sum_{i=1}^{n} (x_i - y_i)^2 \qquad (4.6)$$

$$d(X,Y) = \frac{1}{n} L^2{}_{p=2}(X,Y) = MSE = \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2 \qquad (4.7)$$

e. Kosinüs uzaklığı [35]

$$d(X,Y) = -\cos(X,Y), \qquad (4.8)$$

$$\cos(X,Y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{m} x_i{}^2 \sum_{i=1}^{m} y_i{}^2}} \; ; \qquad (4.9)$$

f. Korelasyon katsayısı tabanlı uzaklık

$$d(X,Y) = -r(X,Y), \tag{4.10}$$

$$r(X,Y) = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{\left(n\sum_{i=1}^{n} {x_i}^2 - \left(\sum_{i=1}^{n} x_i\right)^2\right)\left(n\sum_{i=1}^{n} {y_i}^2 - \left(\sum_{i=1}^{n} y_i\right)^2\right)}}; \tag{4.11}$$

g. Mahalonobis uzaklığı[36]

$$d(X,Y) = -\sum_{i=1}^{n} z_i x_i y_i, \tag{4.12}$$

$$d(X,Y) = -\frac{1}{\sqrt{\sum_{i=1}^{n} {x_i}^2}\sqrt{\sum_{i=1}^{n} {y_i}^2}} \sum_{i=1}^{n} z_i x_i y_i \tag{4.13}$$

h Ağırlıklı Manhattan uzaklığı

$$d(X,Y) = \sum_{i=1}^{n} z_i |x_i - y_i|, z_i = \sqrt{1/\lambda_i}; \tag{4.14}$$

i Ağırlıklı SSE uzaklığı

$$d(X,Y) = \sum_{i=1}^{n} z_i (x_i - y_i)^2, z_i = \sqrt{1/\lambda_i}; \tag{4.15}$$

j. Ağırlıklı Kosinüs uzaklığı

$$d(X,Y) = -\frac{\sum_{i=1}^{n} z_i x_i y_i}{\sqrt{\sum_{i=1}^{m} {x_i}^2 \sum_{i=1}^{m} {y_i}^2}}, z_i = \sqrt{1/\lambda_i}; \tag{4.16}$$

k. Ki Kare uzaklığı

$$d(X,Y) = \chi^2 = \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{x_i + y_i} \quad ; \qquad (4.17)$$

l. Canberra uzaklığı [37]

$$d(X,Y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|} \; ; \qquad (4.18)$$

5. YÜZ VERİTABANLARI

Geliştirilen algoritmaların mevcut diğer algoritmalarla kıyasının yapılabilmesi için standart bazı test verilerinin kullanılması gerekir. Bu maksatla geliştirilmiş spesifik özelliklere sahip onlarca yüz veritabanı vardır. Bu noktada hangi veritabanının seçileceği doğrudan geliştirilecek olan algoritmanın hedefleriyle ilgilidir. Örneğin, algoritmanın hedefi yaşlanma, mimiklerdeki değişme ya da ışık oranındaki değişme gibi durumlardan etkilenmeyen bir yüz tanıma sisteminin geliştirilmesiyse, bu durumda geliştirilen algoritmanın testi için bu özelliklere sahip yüz resimlerinin olduğu bir veritabanı seçilmelidir [38].

Önerilen modelde çeşitli aşamalarda pek çok veritabanı kullanılmıştır. Ancak spesifik olarak ikisi üzerine yoğunlaşılmıştır. Bölüm 7'de bu veritabanları üzerinde yapılan testlere ait karşılaştırmalı performans sonuçları verilmiştir.

5.1 ORL Yüz Veritabanı

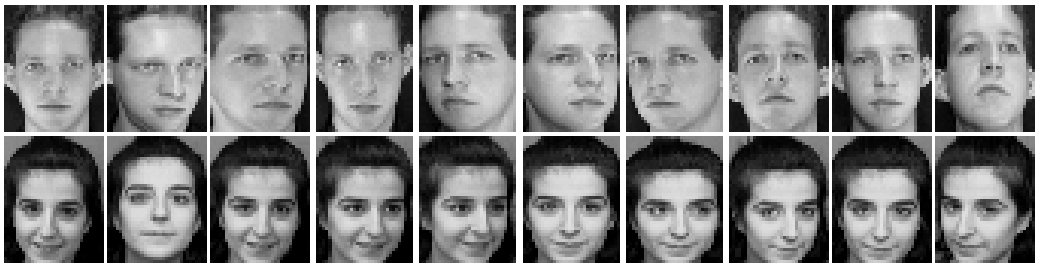
ORL yüz veritabanı Nisan 1992 - Nisan 1994 yılları arasında AT&T laboratuvarlarında çekilmiş yüz fotoğraflarından oluşmaktadır. Bu veritabanı Cambridge Üniversitesi Mühendislik Bölümü Ses, Görüntü ve Robot Grubunun yürüttüğü yüz tanıma projelerinde kullanılmıştır [39].

Veritabanında 40 farklı kişiye ait 10 ayrı resim bulunmaktadır. Bazı kişilere ait fotoğraflar farklı zamanlarda, farklı ışık ve farklı mimiklerle (gözlerin açık/kapalı olması, gülümseme / somurtma vs.) çekilmiştir. Fotoğrafların tamamında denekler önden görüntülenmiş ve arkaplanda koyu ve homojen bir renk kullanılmıştır.

Veritabanındaki resimler PGM formatındadır ve UNIX işletim sisteminde 'xv' gibi programlar kullanılarak görüntülenebilir. Şekil 5.1'de ORL veritabanından iki sınıfa ait yüz görüntüleri verilmiştir.

5.1.1 Teknik Özellikleri

- **Renkli:** Hayır
- **Resim boyutları:** 92x112 px
- **Renk derinliği:** 8 bit, yani piksel başına 256 gri renk seviyesi
- **Denek sayısı**: 40
- **Her deneğe ait resim sayısı:** 10
- **Çekim koşulları:** Tamamı önden çekilmiştir



Şekil 5.1 ORL Yüz veritabanından iki sınıfa ait örnek resimler [39]

5.2 AR Yüz veritabanı

Bu veritabanı Aleix Martinez ve Robert Benavente tarafından Purdue Üniversitesi (USA) Görüntü İşleme Laboratuvarında oluşturulmuştur. 70'i erkek 56'sı bayan olmak üzere 126 deneğe ait 4000'in üzerinde fotoğraftan oluşmaktadır. Ticari kullanımı serbest olmamakla beraber akademik araştırmalar için kullanımı serbesttir. Fotoğrafların çekimi için iki oturum yapılmıştır. İlk oturumda çekilen fotoğrafların aynısı iki hafta (14 gün) arayla tekrar çekilmiştir [40].

Veritabanındaki resimlere ait dosya isimleri aşağıdaki gibi bir formata sahiptir.

Erkek resimleri: *M-xx-yy.raw*
Bayan resimleri: *F-xx-yy.raw*

'xx': fotografin hangi kişiye ait olduğunu gösteren eşsiz bir numaradır. Erkekler için 00'dan 70'e, bayanlar içinse 00'dan 56'ya kadar değer alabilir.
'yy': o fotoğrafin hangi spesifik özelliğe ait olduğunu gösteren koddur. Bu kodların karşılığı şöyledir:

1. Nötr yüz ifadesi
2. Gülümseme
3. Sinirli
4. Bağıran
5. Soldan ışık vuran
6. Sağdan ışık vuran
7. Her iki taraftan da ışık vuran
8. Güneş gözlüğü takan
9. Güneş gözlüğü giyen ve soldan ışık vuran
10. Güneş gözlüğü giyen ve sağdan ışık vuran
11. Kaşkol veya eşarp bağlayan
12. Kaşkol veya eşarp bağlayan ve soldan ışık vuran
13. Kaşkol veya eşarp bağlayan ve sağdan ışık vuran

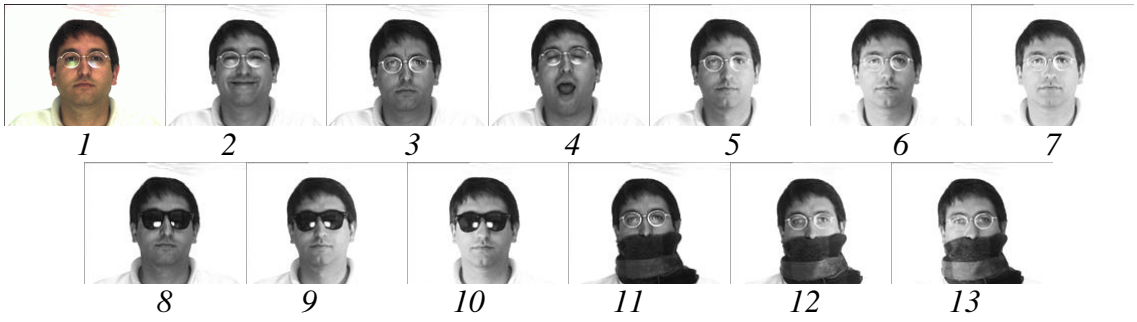
14-26. İkinci oturuma ait resimler (1-13 arasındaki koşullarda çekimler tekrarlanmıştır.)

Aşağıda AR veritabanına ait teknik özellikler verilmiştir. Şekil 5.2 ve Şekil 5.3'de her koda karşılık gelen örnek resimler gösterilmiştir.

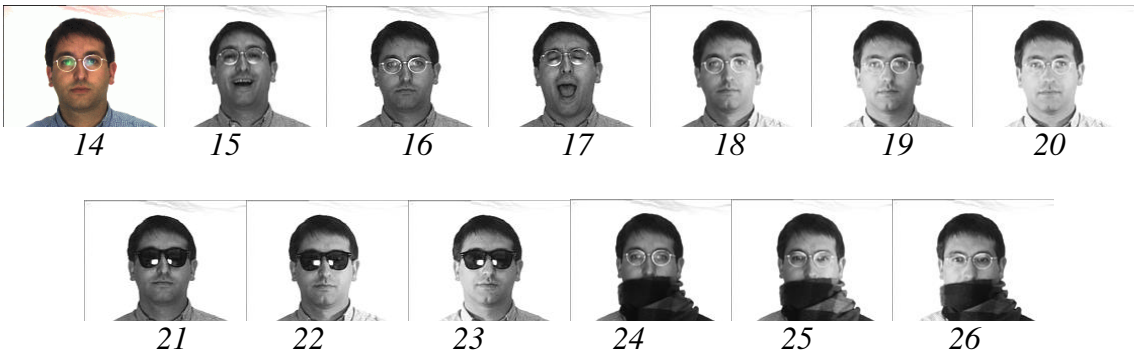
5.2.1 Teknik Özellikleri

- **Renkli:** Evet
- **Resim boyutları:** 576x768 px
- **Renk derinliği:** 24 bit
- **Denek sayısı**: 126; 70 erkek, 56 bayan
- **Her deneğe ait resim sayısı:** 26

- **Çekim koşulları:** İki oturum halinde çeşitli yüz ifadeleriyle tamamı önden çekilmiştir.



Şekil 5.2 AR yüz veritabanından ilk oturuma ait örnek resimler [40]



Şekil 5.3 AR yüz veritabanından ikinci oturuma ait örnek resimler [40]

6. TASARLANAN YÜZ TANIMA MODELİ

Bu bölümde önceki bölümlerde verilen algoritmalar ve teknolojiler ışığında geliştirilen modelin detayları yer almaktadır.

Önerilen modele ait algoritma şu şekildedir:

1. Görüntülerin veritabanından okunarak bir matrise alınması ve sütun matris haline getirilerek yüzler matrisinin oluşturulması
2. Birim filtre bankalarıyla altörnekleme yaparak her yüz görüntüsünden 4 alt görüntü yaratılması ve yüzler matrisinin yeniden elde edilmesi
3. Temel bileşen analizi uygulanması (PCA)
4. Boyut azaltma yapılarak projeksiyon matrisinin oluşturulması
5. İlişkisel bileşen analizi uygulanması (RCA)
6. Beyazlatıcı dönüşüm kullanılması
7. Çeşitli uzaklık metrikleriyle projeksiyon matrisindeki vektörler arasındaki uzaklıkların hesaplanması
8. K-ortalama kümeleme algoritması uygulanması

Bu algoritma, çeşitli uzaklık metrikleri ve değişik öznitelik boyutlarıyla ORL ve AR yüz veritabanlarındaki görüntülere uygulanmış ve benzetimler Bölüm 7'de verilmiştir. Ek 2'de tasarlanan modele ait blok diyagram verilmiştir.

6.1 Veritabanından Görüntülerin Okunması

Önerilen modelin bu kısmı temel bir görüntü okuma algoritmasıdır. Veritabanındaki her bir resim (örneğin ORL veritabanında 40 kişiye ait 10'ar resim, toplam 400 resim bulunur) bir döngüyle okunur ve daha sonraki adımlarda kullanılmak üzere sütun matrise çevrilir. Örneğin, *L=NxM* boyutlarındaki bir yüz görüntüsü, *Lx1*'lik x_k sütun matrisi haline dönüştürülür.

Şekil 6.1'de örnek bir yüz görüntüsünün sütun matrise nasıl dönüştürüldüğü gösterilmektedir. Bu yöntemle veritabanındaki bütün resimler sütun matrisi haline getirilir ve yanyana dizilerek yüzler matrisini oluştururlar. Örneğin, veritabanındaki *k*'ıncı yüz görüntüsünün vektöre dönüştürülmüş hali x_k ise yüzler matrisi

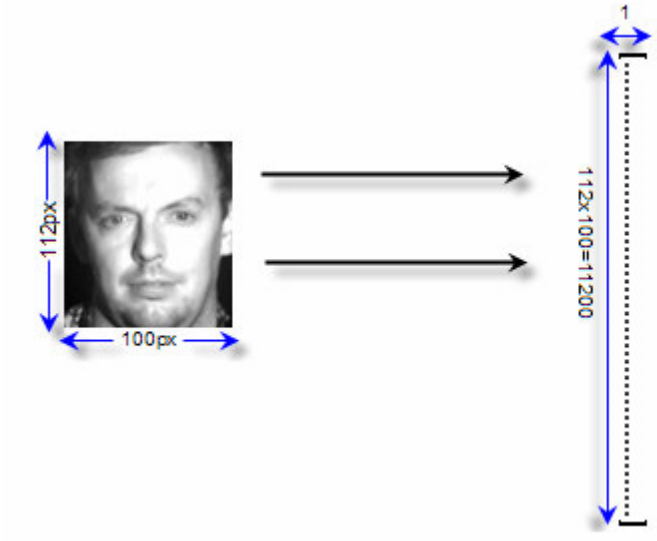
$$X = [x_1, x_2, ..., x_3] \qquad (6.1)$$

olur. Örneğin, ORL veritabanında *N=92* ve *M=112* boyutlarında *K=400* resim yer almaktadır.

$$L = NxM = 92x112 = 10304 \qquad (6.2)$$

x_k sütun matrisinin boyutu *10304x1* olur. ORL veritabanındaki tüm resimler sütun matrise dönüştürüldüğünde ortaya çıkan *X* yüzler matrisinin boyutu 10304x400 olur. (Eşitlik 6.3)

$$LxK = 10304x400 \qquad (6.3)$$



Şekil 6.1 *112x100* boyutundaki yüz resminin veritabanından okunup *11200x1* boyutunda bir sütun matrise dönüşümünü gösteren grafik.

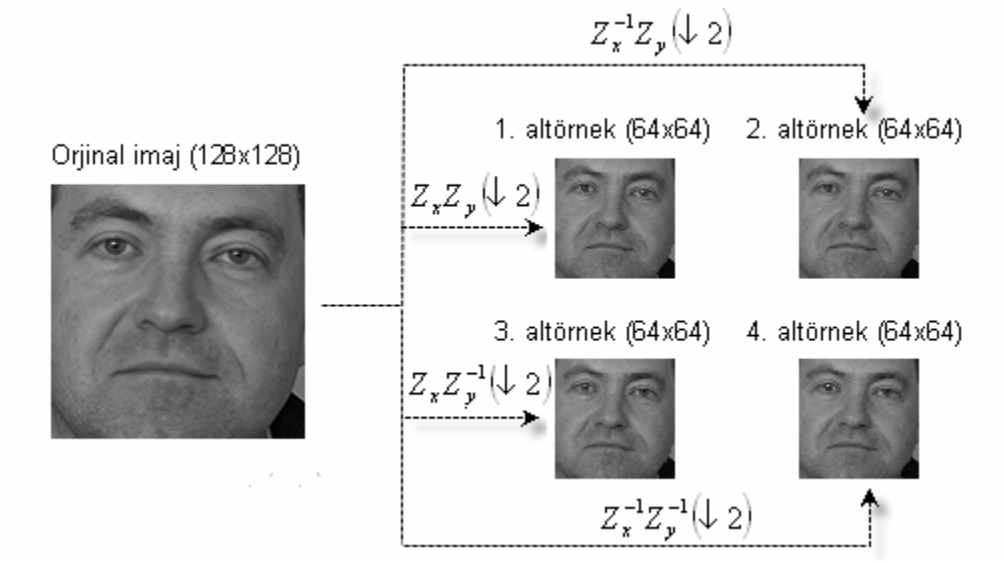
6.2 Birim Filtre Bankasıyla Görüntü Türetme

Yüzler matrisinin kovaryans matrisinin ve dolayısıyla da içinde saçılma matrisinin *(K-1)* adet (*K*, veritabanındaki toplam resim sayısı) sıfırdan farklı özdeğeri bulunmaktadır. (örn. ORL yüz veritabanı için en fazla 399) Bu sorun, literatürde küçük örnek büyüklüğü problemi olarak tanımlanmıştır [41].

Yüz görüntülerinde, bir görüntünün Fourier dönüşümü sonucu frekans gösterimi $0 \le (w_x, w_y) < \pi/2$ 'lik bir bölgeyi kapsıyorsa, bu görüntünün altörneklenmiş görüntülerinden orjinali bozulmadan geri elde edilebildiği söylenebilir. Bir görüntünün her iki yönde de altörneklenmesi sonucu 4 adet yeni görüntü elde edilebilir. Bu dört adet yeni görüntü istatiksel olarak birbirine benzeseler bile birbirlerinin tamamen aynısı değildir [6]. İki boyutlu bir görüntüden dört adet altörneklenmiş görüntü elde etmek, bu görüntüyü birim filtre bankasından geçirmekle eşdeğerdir. Birim filtre bankası mükemmel geri çatım koşulları sağlar [42].

Önerilen bu yönteme göre görüntüler hem *x* hem de *y* yönünde örneklenmekte, sonuçta elde edilen dört adet görüntü sınıflandırma işlemiyle ilişkili yüksek ağırlıklı bileşenleri barındırmaktadır. Şekil 6.2'de bir yüz görüntüsünden türetilen 4 alt örnek görülmektedir.

Önerilen yöntemde ORL veritabanı kullanılmış ve bir önceki adımda *X* yüzler matrisi 10304x400 olarak oluşturulmuştur. *X* yüzler matrisi birim filtre bankasıyla altörneklerine ayrılarak 2576x1600 olacak hale getirilmiş ve veritabanındaki 40 kişiye ait 40'ar adet alt görüntü oluşturulmuştur.



Şekil 6.2 Birim filtre bankası ile orjinal yüz görüntüsünden türetilen 4 alt görüntü. (örnekteki yüz görüntüsü FERET veritabanından [43]alınmıştır)

6.3 Temel Bileşen Analizi ve Boyut Azaltma

Yüz tanıma probleminde temel önişleme metotlarından sonraki ilk işlem temel bileşen analiziyle projeksiyon matrisinin oluşturulması ve boyut azaltma yoluna gidilmesidir. Bunun için literatürde özyüzler metotu olarak da bilinen yaklaşım kullanılır.

X yüzler matrisinin oluşturulması ve altörneklerek yeniden elde edilmesi sonrasındaki ilk iş *X* matrisinden ortalama değerini çıkartmak ve saçılma matrisi S_z hesaplamaktır [6].

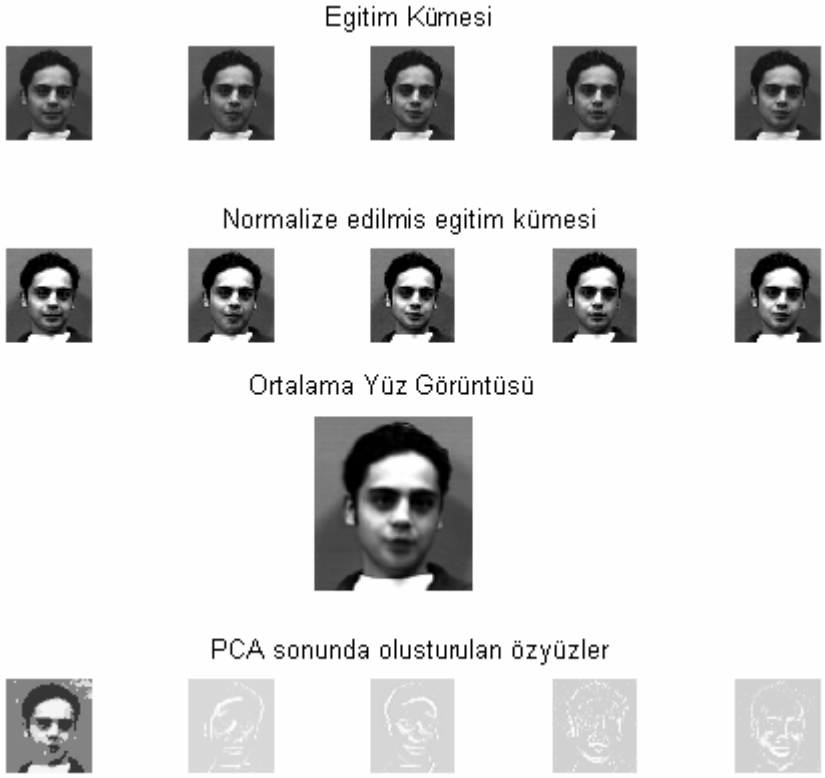
$$S_z = \frac{1}{K}\left(X - \overline{X}\right)\left(X - \overline{X}\right)^T \qquad (6.4)$$

Daha sonra *X* yüzler matrisi S_z saçılma matrisinin en yüksek özdeğerlerine karşılık gelen özvektörler üzerine izdüşürülür.

$$W = \psi^T X \qquad (6.5)$$

Φ_k, *k*'inci en yüksek özdeğere sahip özvektörü göstermek üzere $\psi = [\phi_1, \phi_2, ...\phi_P]$ şeklindedir ve boyutu *KxP* olmaktadır. *P<<L* olduğundan dolayı elde edilen *W* matrisinin boyutları azalmıştır ve böylelikle temel bileşen analizi ile boyut azaltma işlemi gerçekleşmiştir.

Örneğin, en yüksek değere sahip *300* özvektörü aldığımızı varsayarsak projeksiyon matrisi *W*'nun boyutu *300x1600* olacaktır. Seçilen özvektör sayısı değiştirildiğinde sınıflandırma performansı da değişmektedir. İlgili grafikler bir sonraki bölümde ele alınacaktır. Şekil 6.3'de adım adım PCA uygulanmış bir fotoğrafa ait görüntüler verilmiştir.



Şekil 6.3 PCA uygulanmış bir eğitim kümesine ait görüntüler. Sırasıyla en üstte PCA uygulanacak olan eğitim kümesi. İkinci sırada, ortalaması sıfırlanarak (normalize edilerek) elde edilen yüz görüntüleri. Üçüncü sırada, normalizasyon sırasında orjinal görüntülerden çıkarılan ortalama yüz görüntüsü. En altta, PCA sonunda elde edilen özyüzlere ait görüntüler.

6.4 İlişkili Bileşen Analizi

Bir bilgicik, hangi sınıfa ait olduğu bilinmeyen ancak aynı sınıfa ait olduğu bilinen noktalar kümesidir. Altörneklemeyle her bir yüz görüntüsünden dört alt görüntü elde edilmiştir. O halde bu dört görüntünün hangi sınıfa ait olduğunu bilinmese bile aynı kişiye ait olduğunu yani aynı sınıfa ait olduğu biliniyor demektir. Alt örneklemeyle elde edilen bu görüntüler RCA'da bilgicik olarak kullanılacaktır. PCA ve boyut azaltmayla elde edilen projeksiyon matrisi eldeki bilgicikler kullanılarak RCA'dan geçirilir. RCA spesifik olarak şu adımları takip eder [[26]] :

1. Her bir bilgicikten o bilgiciğin ortalamasını çıkarır. Bu şekilde bilgiciklerin ortalaması sıfırlanmıştır.
2. Ortalaması sıfırlanan bilgiciklerin kovaryans matrisi hesaplanır. Toplam p noktadan oluşan k adet bilgicikten j'incisi $\{x_{ji}\}_{i=1}^{n_j}$ şeklinde gösterilir ve ortalaması $\hat{m}_j$ 'dir. RCA kovaryans matrisi $\hat{C}$'i aşağıdaki gibi hesaplar.

$$\hat{C} = \frac{1}{p}\sum_{j=1}^{k}\sum_{i=1}^{n_j}\left(x_{ji} - \hat{m}_j\right)\left(x_{ji} - \hat{m}_j\right)^T \tag{6.6}$$

6.5 Beyazlatıcı Dönüşüm

Özdeğerlerin büyüklüğüyle özniteliklerin dağılımı arasında bir ilişki bulunmaktadır. Doğrusal bir dönüşüm kullanarak yüz görüntüsünün kovaryans matrisinin beklenen değerini birim matris haline dönüştürmek mümkündür. Eldeki verilerin Gauss dağılımlı olduğu varsayılırsa, yüzler matrisini kovaryans matrisinin tersinin köküyle çarpmak kovaryans matrisinin beklenen değerini birim matris haline dönüştürecektir. C yüzler matrisinin beklenen değerini, $E(.)$ beklenen değer operatörünü ve I birim matrisi göstermektedir.

$$E\left(\left(C^{-0.5}X\right)\left(C^{-0.5}X\right)^T\right) = I \tag{6.7}$$

Literatürde değişik beyazlatıcı dönüşümler de mevcuttur [26] [41].

Beyazlatıcı dönüşümü W eşitlik 6.8'deki gibi hesaplanır. Formüldeki $\hat{C}$, RCA'da hesaplanan kovaryans matrisidir.

$$W = \hat{C}^{-\frac{1}{2}} \qquad (6.8)$$

Elde edilen beyazlatıcı dönüşüm matrisi mevcut izdüşüm matrisine uygulanarak X_{yeni} yeni izdüşüm matrisi hesaplanır.

$$X_{yeni} = WX \qquad (6.9)$$

6.6 Kullanılan Uzaklık Metrikleri

Önerilen modelde bölüm 4'de anlatılan uzaklık metriklerinden 5 tanesini kullanılmıştır. Bu metrikler çeşitli öznitelik boyutlarında tekrar tekrar denenmiş ve deneme sonuçları bir sonraki bölümde sunulmuştur.

Kullanılan metrikler şöyledir:

1. Kareli Öklit (Squared Euclid - SSE),
2. Kosinüs,
3. Mahalonobis,
4. Manhattan,
5. Korelasyon katsayı tabanlı

6.7 Yüz Görüntülerinin Sınıflandırılması

Önerilen modelde sınıflandırma için K-ortalama kümeleme algoritması kullanılmaktadır. ORL veritabanında 40 kişiye ait 10'ar yüz görüntüsü bulunmaktadır. Altörneklemeyle birlikte her yüz görüntüsünden 4 adet alt görüntü türetilmekte, dolayısıyla her kişiye ait 40 yüz görüntüsü elde edilmektedir. Sınıflandırma algoritması aşağıdaki gibi uygulanmaktadır.

1. K-ortalama kümeleme algoritmasının ilk aşaması k küme sayısının belirlenmesidir. Toplam 40 ayrı kişi olduğu için k küme sayısı 40'tır.
2. Her kümenin ortalaması, kümeyi oluşturan 40 adet yüz görüntüsünün merkez noktası yani ortalaması alınarak belirlenmektedir.
3. Her yüz görüntüsünün belirlenen merkez noktalarına olan uzaklığı hesaplanır. Elde edilen sonuçlara göre tüm yüz görüntüleri $k = 40$ adet kümeden kendilerine en yakın olan kümeye yerleştirilmektedir.
4. Oluşan kümelerin yeni merkez noktaları, kümeye yerleştirilen tüm yüz görüntülerinin ortalaması alınarak yeniden hesaplanır ve bu adımlar küme merkez noktaları değişmeyene kadar tekrar edilir.

K-ortalama algoritmasında her bir yüz görüntüsünün küme merkez noktalarına olan uzaklığını hesaplamak için kullanılan uzaklık metrikleri bir önceki adımda verilmiştir. Sınıflandırma işlemleri bu uzaklık metriklerinin her biri için defalarca tekrarlanmış ve sonuçları Bölüm 7'de verilmiştir.

7. BENZETİMLER

Önerilen modelde, öncelikle hem ORL hem de AR veritabanlarındaki yüz görüntüleri birim filtre bankalarıyla altörneklerine ayrılmıştır. Altörnekleme öncesi yüz görüntüleri ORL veritabanı için 92x112 ve AR veritabanı için 576x768 iken altörnekleme sonrası ORL veritabanı için 46x56 ve AR veritabanı için 288x384 olmuştur. Bölüm 6'da açıklanan ve Ek 2'de blok diagramı verilen adımlar izlenerek önce normalizasyon işlemi gerçekleştirilmiş ve Temel Bileşen Analiziyle boyut azaltılmıştır. Daha sonra İlişkili Bileşen Analizi ve beyazlatma işlemi yapılmış ve kareli öklit, manhattan, kosinüs, korelasyon tabanlı ve mahalonobis uzaklık metrikleri kullanılarak sınıflandırma işlemleri gerçekleştirilmiştir.

Önerilen modelde, hem ORL hem de AR veritabanı üzerinde klasik PCA ve PCA ile RCA'nın birlikte kullanıldığı benzetimler yapılmış; bu benzetimler her bir uzaklık metriği için 10'ar defa tekrarlanmıştır. Benzetimlerin aynı değerlerle defalarca tekrarlanmasının nedeni daha kararlı ve net sonuçlara ulaşabilmektir.

Değişen özvektör sayılarıyla, PCA ve PCA+RCA algoritmaları kullanılarak AR ve ORL veritabanları üzerinde 5 ayrı uzaklık metriğiyle yapılan testlere ait en iyi sınıflandırma başarım oranları Tablo 7.1'de verilmiştir. Klasik PCA'in en iyi doğru tanıma oranı %79.65 iken, RCA ile yapılan benzetimlerde en iyi doğru tanıma oranı %92.34'e kadar çıkmıştır.

Ek 3'de en iyi başarım ve hata oranlarının kullanılan özvektör sayısıyla değişimini gösteren ayrıntılı bir tablo verilmiştir.

Her bir veritabanı ve algoritma kombinasyonu için başarı ve hata oranlarının özvektör sayılarıyla nasıl değiştiğini gösteren ve uzaklık metriklerinin birbiri üzerine örtüştürülüp kıyaslandığı grafikler aşağıda verilmiştir (Şekil 7.1-7.8).

Tablo 7.1 En iyi doğru tanıma oranları

Uzaklık Metrikleri	ORL Veritabanı		AR Veritabanı	
	Klasik PCA	PCA+RCA	Klasik PCA	PCA+RCA
	Başarı Oranı (%)	Başarı Oranı (%)	Başarı Oranı (%)	Başarı Oranı (%)
Kosinüs	79.65	92.34	73.82	86
Mahalonobis	73.29	88.61	65.83	81.68
SSE	77.46	87.98	70.51	84.83
Manhattan	74.44	86.73	71.31	82.01
Korrelasyon	78.7	88.32	68.14	84.45

ORL veritabanına ait

- başarı oranı grafikleri Şekil 7.1 ve Şekil 7.2'de;
- hata oranı grafikleri Şekil 7.5 ve Şekil 7.6'da

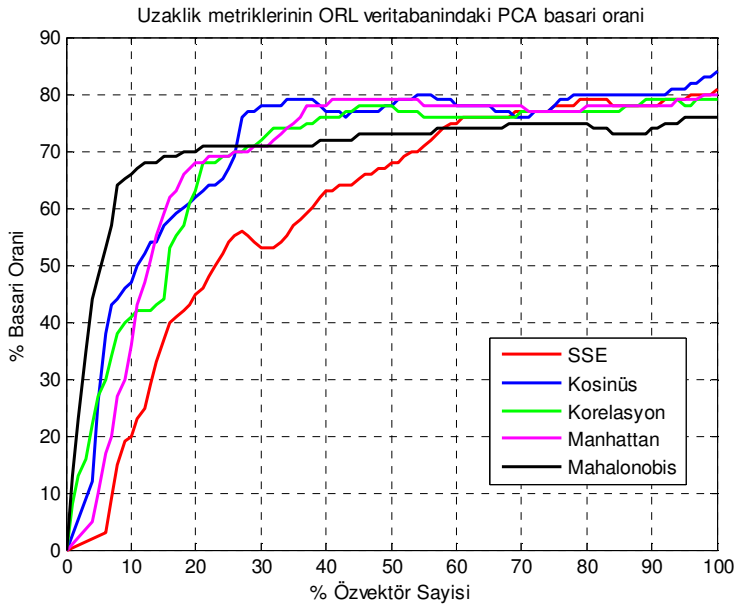
verilmiştir.

AR veritabanına ait

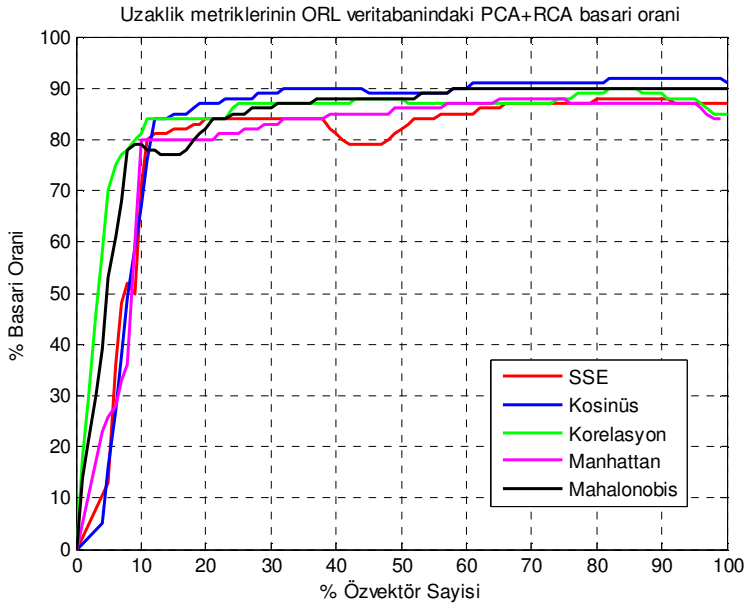
- başarı oranı grafikleri Şekil 7.3 ve Şekil 7.4'de;
- hata oranı grafikleri Şekil 7.7 ve Şekil 7.8'de

verilmiştir.

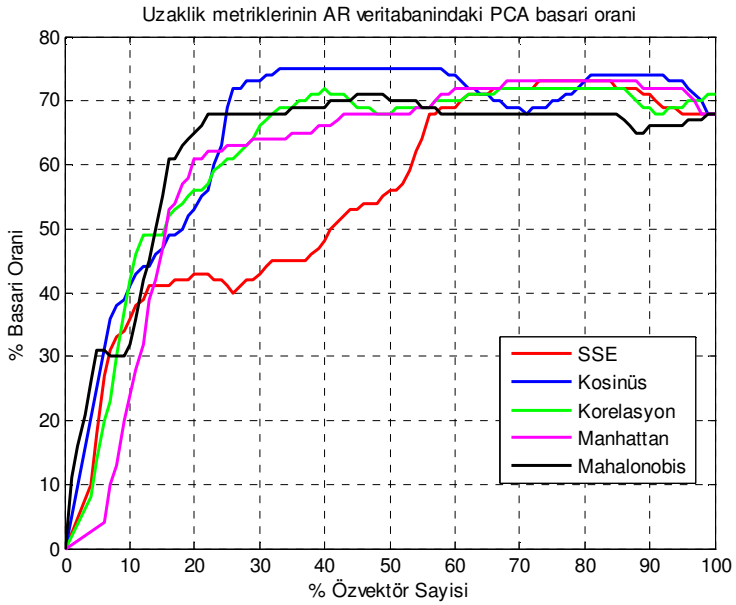
AR veritabanındaki çekim koşullarının ORL veritabanındaki çekim koşullarından daha farklı olduğunu, özellikle AR veritabanındaki yüz görüntülerinde çeşitli aksesuarların da yer aldığını hesaba kattığımızda bu etkenlerin AR veritabanındaki tanıma başarı oranının daha düşük olmasına neden olduğu çıkarımı yapılabilir. Tüm algoritmalar için tüm testlerde Kosinüs uzaklık metriği diğerlerine göre daha başarılı sonuç vermiştir.



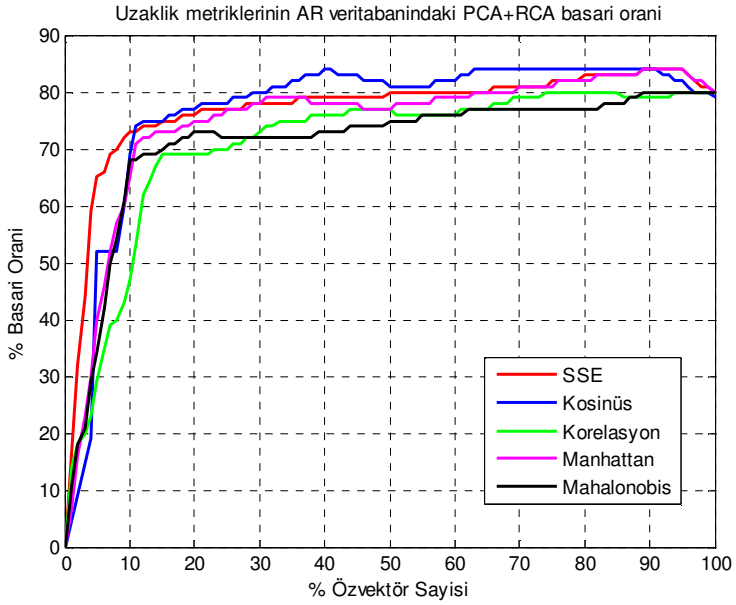
Şekil 7.1 ORL veritabanında PCA algoritmasıyla yapılan testlere ait başarı oranının özvektör sayısıyla değişimini gösteren grafik



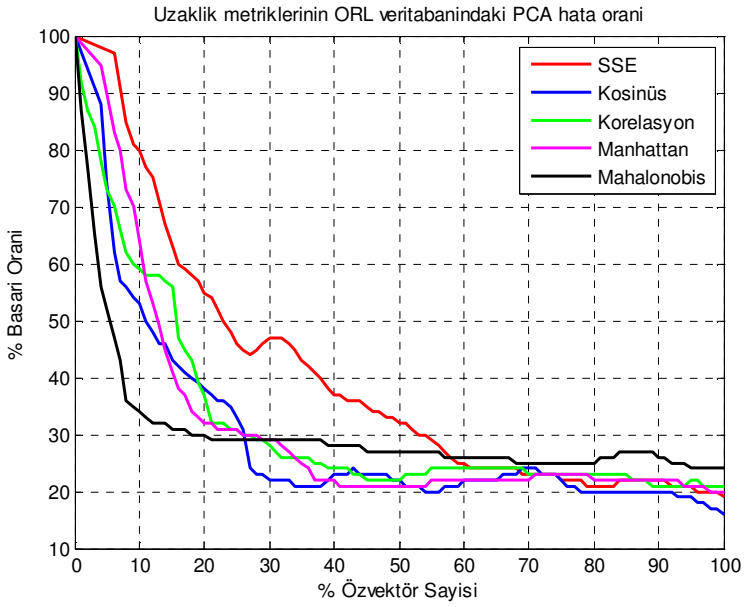
Şekil 7.2 ORL veritabanında PCA + RCA algoritmasıyla yapılan testlere ait başarı oranının özvektör sayısıyla değişimini gösteren grafik



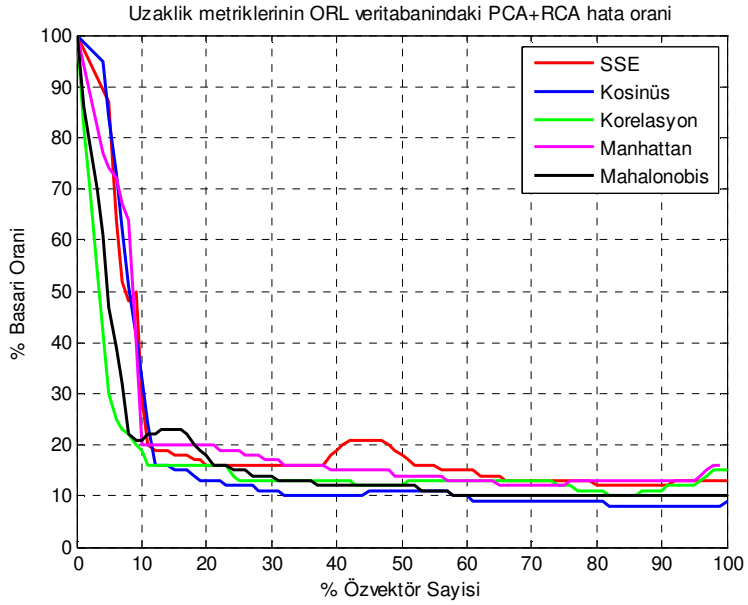
Şekil 7.3 AR veritabanında PCA algoritmasıyla yapılan testlere ait başarı oranının özvektör sayısıyla değişimini gösteren grafik



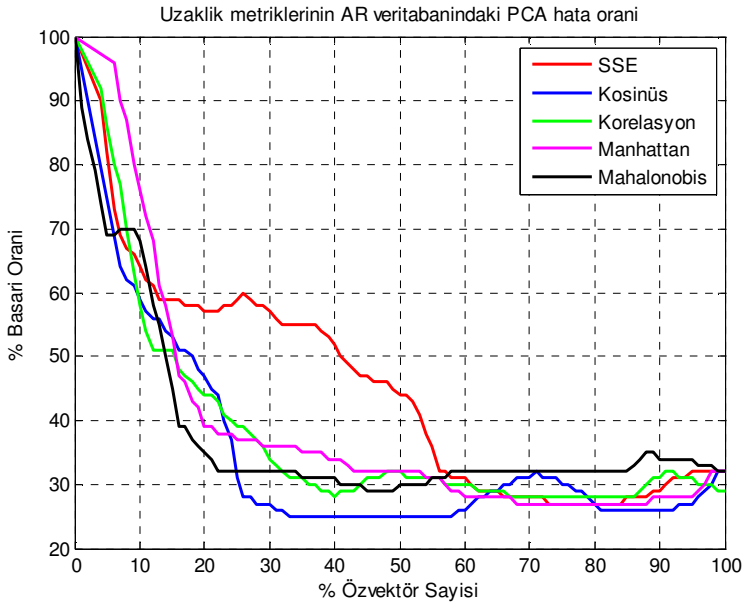
Şekil 7.4 AR veritabanında PCA + RCA algoritmasıyla yapılan testlere ait başarı oranının özvektör sayısıyla değişimini gösteren grafik



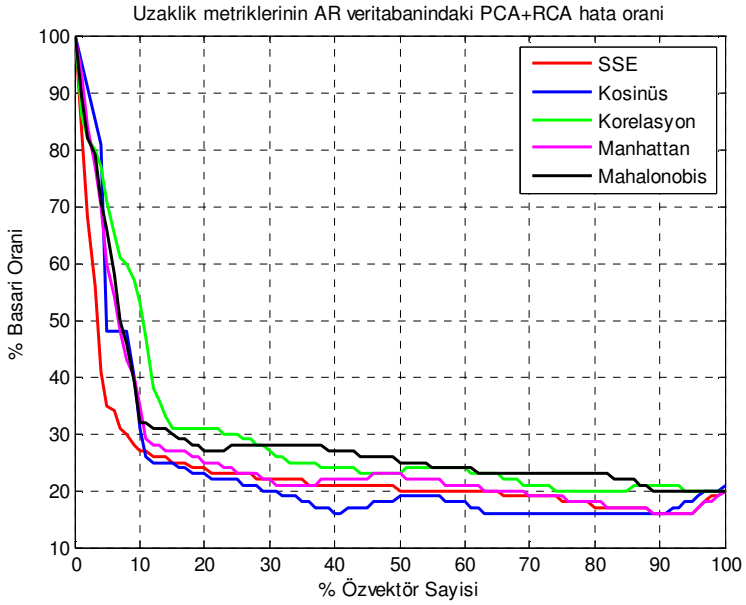
Şekil 7.5 ORL veritabanında PCA algoritmasıyla yapılan testlere ait hata oranının özvektör sayısıyla değişimini gösteren grafik



Şekil 7.6 ORL veritabanında PCA + RCA algoritmasıyla yapılan testlere ait hata oranının özvektör sayısıyla değişimini gösteren grafik



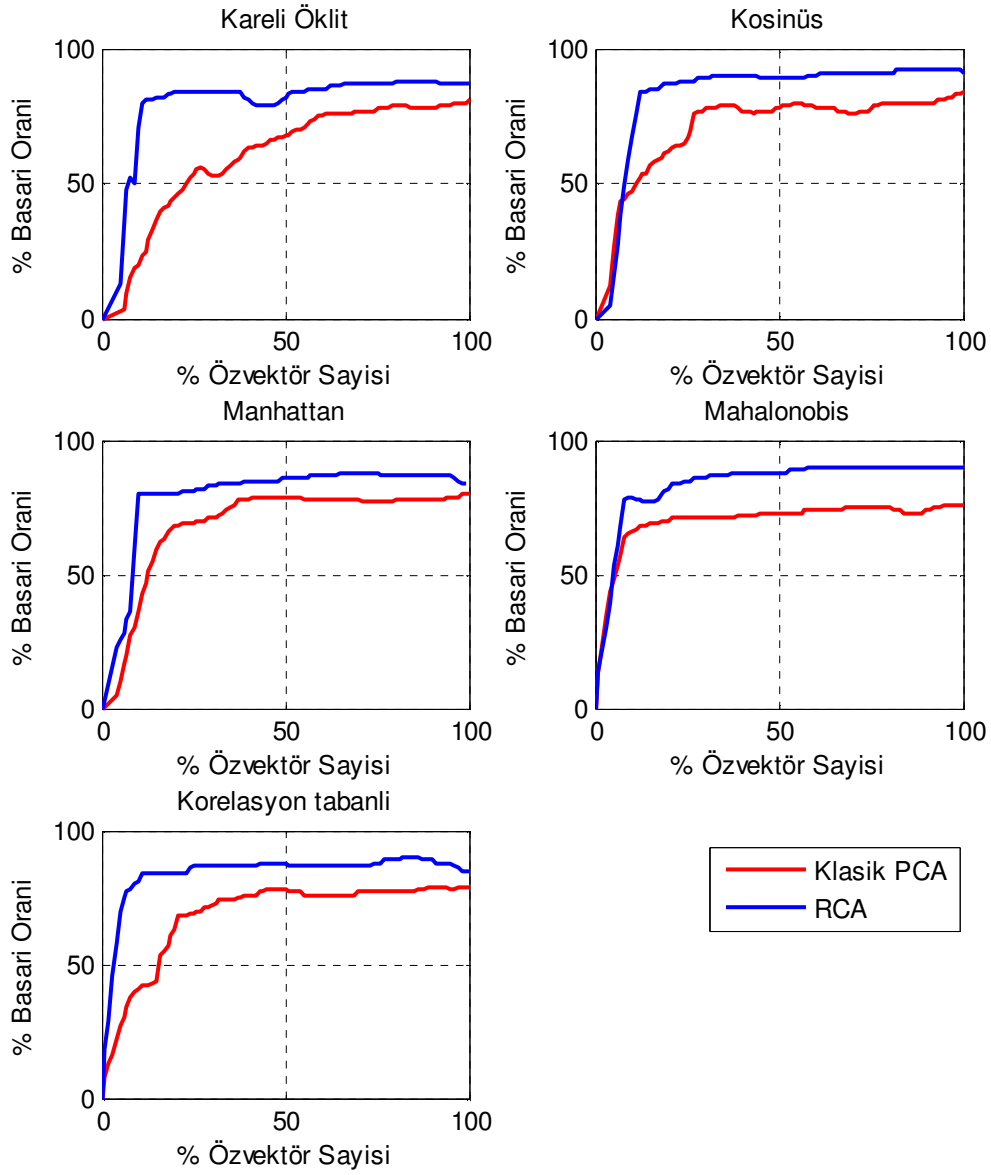
Şekil 7.7 AR veritabanında PCA algoritmasıyla yapılan testlere ait hata oranının özvektör sayısıyla değişimini gösteren grafik



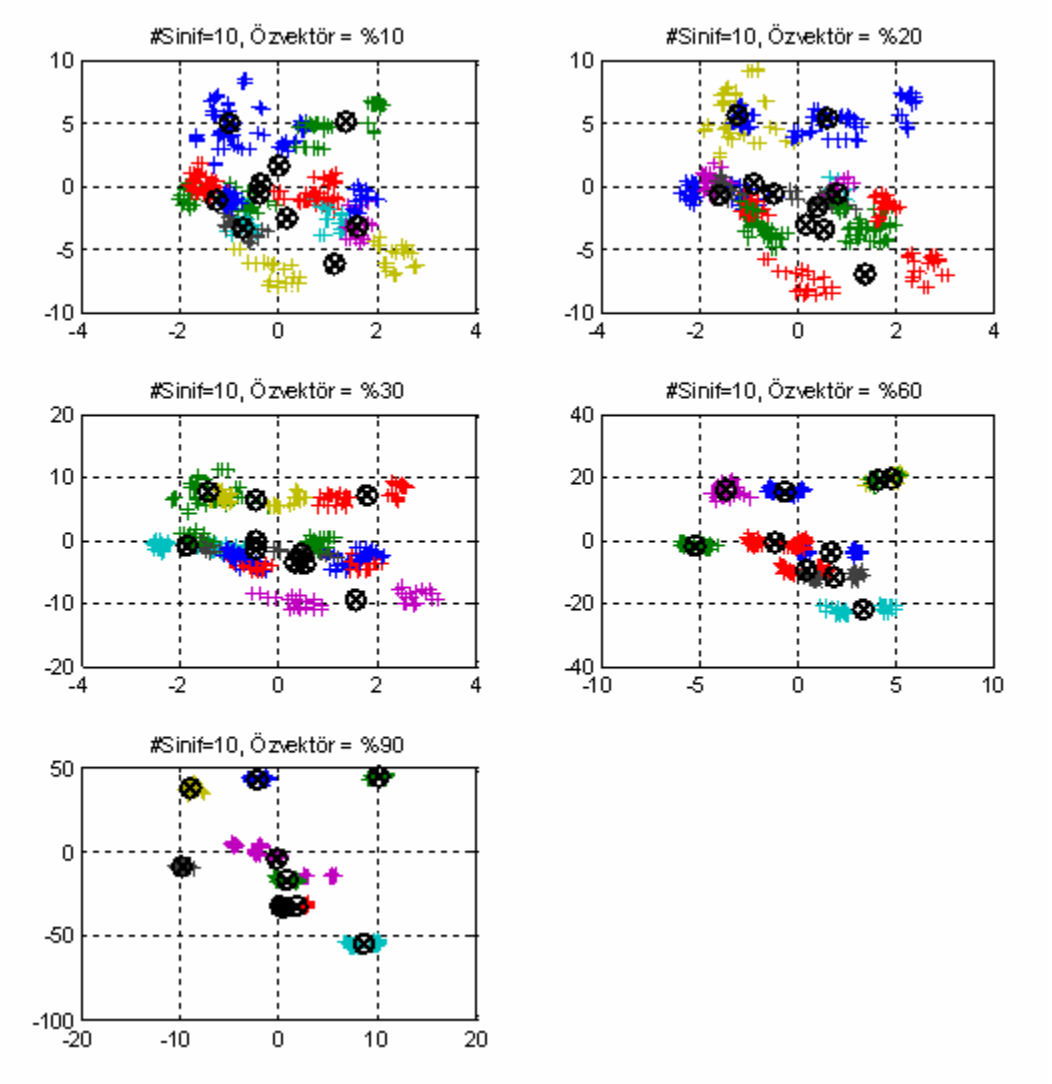
Şekil 7.8 AR veritabanında PCA + RCA algoritmasıyla yapılan testlere ait hata oranının özvektör sayısıyla değişimini gösteren grafik

Şekil 7.9'da uzaklık metrikleri bazında klasik PCA ve RCA algoritmalarının başarı oranlarının karşılaştırılması verilmiştir.

Şekil 7.10'da ORL veritabanında RCA kullanılarak 10 kişilik bir sınıf üzerine yapılan testlere ait sınıflandırma grafikleri verilmiştir. Sınıflandırmada uzaklık metriği olarak kosinüs uzaklığı kullanılmıştır. Şekil 7.10'daki grafikte her renk bir sınıfı temsil etmekle birlikte, kullanılan özvektör sayısı arttıkça içinde saçılma uzaklıklarının azaldığı ve aynı renkli noktaların birbirine daha da yakınlaştığı; aynı şekilde sınıflar arası saçılma uzaklıklarının da arttığı görülmektedir. Kullanılan özvektörlerin % 10 ve % 20 olduğu testlerde iz düşüm matrisinin genişliğinin [-10 10] ve [-4 4] aralığında değiştiği ve sınıf içi dağılımların çok kötü olduğu ve sınıfların neredeyse birbiri içine geçtiği görülmektedir. Buna karşılık, özvektörlerin % 60 ve % 90'ının kullanıldığı testlerde sınıflar arasındaki uzaklıkların iyice arttığı ve izdüşüm matrisinin [-100 50] ve [-20 20]'lik bir düzleme yayıldığı, aynı şekilde sınıflar içindeki noktaların da birbirlerine çok yaklaştığı ve neredeyse birbirleri üzerine bindikleri görülmektedir. Bu sınıflandırma başarısının kullanılan özvektör sayısıyla doğru orantılı olduğunu göstermektedir.



Şekil 7.9 Klasik PCA ve RCA başarı oranlarının her bir uzaklık metriğine göre karşılaştırılmalı grafikleri



Şekil 7.10 ORL veritabanındaki 10 sınıf üzerinde *PCA + RCA* algoritması ve kosinüs uzaklık metriği kullanılarak yapılan benzetimlere ait sınıflandırma grafiği.

8. SONUÇLAR & YORUMLAR

Önerilen model birim filtre bankalarıyla yapılan altörnekleme işlemiyle başlamıştır. Bunun çeşitli avantajları vardır:

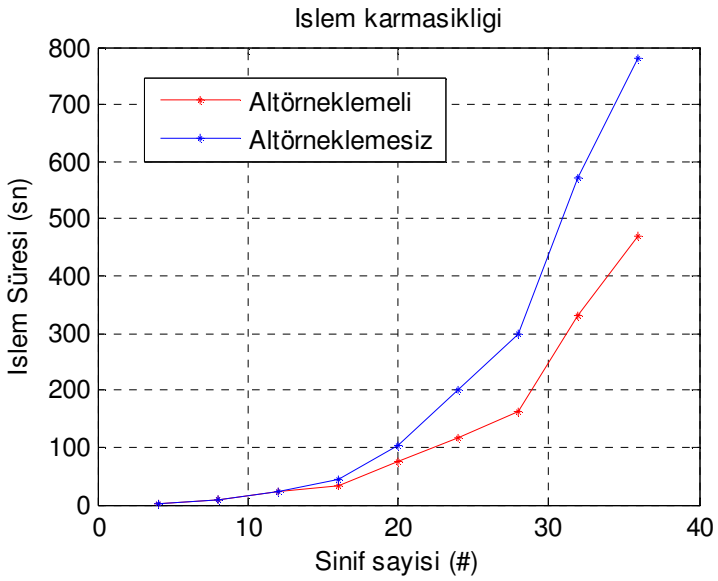
1. Her bir yüz görüntüsünden türetilen 4 ayrı alt görüntü sayesinde özvektör sayısının artırılması sağlanmıştır. Özvektör sayısının artmasıyla birlikte sınıflandırma başarı oranının da belirgin bir şekilde arttığı gösterilmiştir. Klasik yöntemlerde en iyi doğru sınıflandırma oranıyla en iyi yanlış sınıflandırma oranları küçük bir özdeğer aralığında bulunmaktadır. Bu durumda boyult azaltma yapıldığında uygun olan özdeğer aralığını tespit etmek her zaman kolay değildir. Bu modelde en iyi doğru sınıflandırma ve en iyi yanlış sınıflandırma oranları geniş bir özdeğer aralığında kabul edilebilir seviyededir.

2. Türetilen altgörüntülerin pozitif yan bilgi olarak kullanıldığı bu eğiticisiz yeni modelde klasik PCA'i takiben RCA'nın kullanılması alakasız bileşenlerin bastırılmasını ve doğru sınıflandırma oranının da kayda değer bir şekilde yükselmesini sağlamıştır. Klasik PCA'de en iyi doğru tanıma oranı %79.65 seviyelerindeyken, RCA'nın kullanılmasıyla birlikte en iyi doğru tanıma oranı % 92.34'e çıkarılmış ve doğru sınıflandırma oranında % 12.69'lık bir yükseliş elde edilmiştir.

3. İşlem karmaşıklığı açısından değerlendirildiğinde, şayet altörnekleme yapılmamış olsaydı ORL veritabanındaki $N = m \times n = 92 \times 112\, px$ boyutlarındaki $K = 400$ adet yüz görüntüsü için $N \times K = 10304 \times 400$ boyutunda bir yüz matrisi oluşacaktı. Bu büyüklükte bir matrisin kovaryans matrisini eşitlik 2.7'de verilen denkleme göre hesaplamak için milyonlarca çarpma ve toplama işlemi yapılması gerekmektedir. Böylesine bir hesap için 2GB'dan daha yüksek kapasitede bir hafıza kartı olan performansı yüksek bir bilgisayara ihtiyaç vardır. Bu koşullarda dahi bu işlem çok uzun sürmektedir. Hesaplama işlemi bittiğinde kovaryans matrisinin büyüklüğü 10304×10304 'tür. Bu kovaryans matrisini kullanarak özdeğer ve özvektörlerin hesaplanması saatler almaktadır. Buna karşılık

altörnekleme yapıldığında yüz matrisinin boyutu 2576×1600 olmaktadır. 256 Mb üzeri bir hafıza kartına sahip herhangi bir bilgisayarla bu matrisin kovaryans matrisi, özdeğer ve özvektörleri kolaylıkla çok hızlı bir şekilde hesaplanabilmektedir. Yani altörnekleme işlemi hem yüksek teknoloji ürünü sistemlere olan bağımlılığımızı azaltmakta hem de işlem hızını oldukça yükseltmektedir.

Şekil 8.1'de sınıflandırma işlemi sayısına karşılık, altörnekleme yapılan ve yapılmayan durumlardaki işlem hızını gösteren işlem karmaşıklığı grafiği verilmiştir. Bu benzetimde ORL veritabanındaki ait yüz görüntüleri kullanılmıştır. Boyut azaltma işlemi özvektör sayısının % 90'ı kullanılarak yapılmış ve k-ortalama kümelemesinde küme merkezleri 10 tekrarla kosinüs uzaklık metriği kullanılarak hesaplanmıştır.



Şekil 8.1 İşlenen sınıf sayısına karşılık gelen işlem süresinin altörneklemeli ve altörneklemesiz durumlardaki değişim grafikleri.

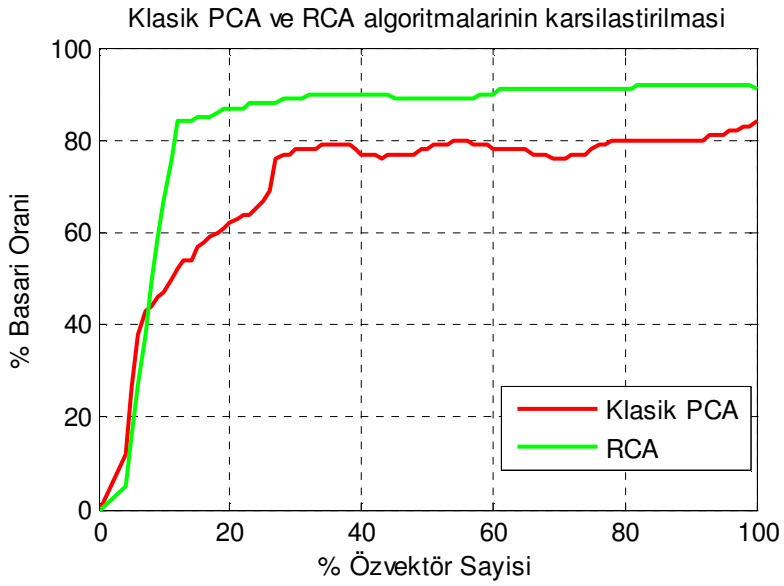
Şu ana dek bu sistemde alt örnekleme yapılmasının önerilen modelin başarısına katkısından bahsedilmiştir. Ancak altörnekleme yapılması bu sistemin başarılı bir şekilde çalışması için mutlak bir zorunluluk teşkil etmez. Örneğin, güvenlik gerektiren sistemlerde çekilen tek bir

resimden bir kaç alt görüntünün türetilmesi yerine, kişinin 4-5 fotoğrafı bir arada seri çekimlerle elde edilip bilgicik kümesi olarak kullanılabilir.

Bu model yüz tanımanın güvenlik maksadıyla kullanıldığı pek çok sistemde, örneğin yetkilendirme ve/veya doğrulama sistemlerinde doğru sınıflandırma oranını yükselttiği ve yanlış sınıflandırma oranını düşürdüğü için klasik sistemlere tercih edilebilir.

Tasarlanan modelde kullanılan görüntüler herhangi bir önişlemden geçirilmemiştir. Önişlemden geçirmek en iyi doğru tanıma oranını yükseltmekte, hata oranını ise düşürmektedir. Burda önerilen modelin klasik PCA'e göre olan avantajlarını ortaya koyabilmek için yüz görüntüleri orjinal haliyle, hiçbir önişlemden geçirilmeden ele alınmıştır.

Şekil 8.2'de klasik PCA ile RCA'ya ait ORL veritabanındaki sınıflandırma başarım grafiği verilmiştir. Bu sınıflandırma sırasında kosinüs uzaklık metriği kullanılmıştır.



Şekil 8.2 Klasik PCA ile RCA'nın karşılaştırıldığı sınıflandırma başarım grafiği

EKLER

Ek 1: Geliştirilen Modele Ait Matlab Kodları

Ek 1.1 PCA() Fonksiyonu

Tezde algoritması verilen temel bileşen analizini uygulayıp boyult azaltma işlemini gerçekleştiren fonksiyondur.

```
%-----------------------------------------------------------
%Tarih:7 Temmuz 2008
%Geliştiren:Bilal Karaduman
%Açıklama: PCA ve boyult azaltma uygulayan kod
%-----------------------------------------------------------

function uNew=pca(subDim,numberOfFolders,boolDownSample)

%% Çalışma ortamını ve ram'i temizle
clc;
%close all;

 %Sabit değişkenler
N=numberOfFolders; % Kullanılacak klasör(sınıf) sayısı
M=10; % Her bir klasördeki yüz görüntüsü sayısı
numIm = N*M;
 save constants N M numIm;

%% 1-imSpace matrisi için RAM'de yer rezerve et
fprintf('1-Memory allocation \n');
fID=1;
folder=strcat('ORL\s',int2str(fID));
```

```
folder=strcat(folder,'\');%dizin ismini oluştur

image=strcat(int2str(1),'.pgm');
image=strcat(folder,image);%dosya ismini oluştur
eval('tmp=imread(image);');

[m n]=size(tmp);    % görüntüye ait satir(m) ve sütun(n) sayısını al
imSpace=uint8(zeros(m*n,numIm)); %RAM'de alan rezerve et

clear tmp; %Hafızayı boşalt

%% 2-DATA Matrisini oluştur
fprintf('2-Creating training images space \n');
tic
ind=1;
for fID=1:N
    folder=strcat('ORL\s',int2str(fID));
    folder=strcat(folder,'\');%dizin ismini üret

    for i=1:M
        image=strcat(int2str(i),'.pgm');
        image=strcat(folder,image);%görüntü ismini türet
        eval('im=imread(image);');

        imSpace(:, ind) = reshape (im, m*n, 1);
        ind=ind+1;
   end
end
toc
size(imSpace)%(1)=10304x40
```

```
%Hafızayı boşalt
clear fID ind i folder image im;

%% 3-Orjinal görüntülerden altörneklemeyle yeni görüntüler türet
% Altörnekleme yapilsin mi?
if(boolDownSample==1)

    fprintf('3-Do subsampling\n');
    tic
    tmpS=[];
    for i=0:(N-1)
        tmpFrom=i*M+1;
        tmpTo=tmpFrom+M-1;
        for j=tmpFrom:tmpTo
            temp=reshape(double(imSpace(:,j)),n,m)';
            tmpS=[tmpS splitMatrix(temp)];
        end
    end
    imSpace=[];
    imSpace=tmpS;
    size(imSpace)%(2)=2576x1600
    %de?i?kenleri diske yaz
    save imSpace imSpace;
    toc

end

%% 4-Egitim görüntülerinden ortalama resmi hesapla
fprintf('4-Calculating mean face from training images\n');
```

```
psi = mean(double(imSpace'))';
save psi psi;

%% 5-Ortalaması sıfır olan matrisi oluştur
fprintf('5-Calculate zero mean space\n');
tic
zeroMeanSpace = zeros(size(imSpace));
for i = 1 : size(imSpace,2)
    zeroMeanSpace(:, i) = double(imSpace(:, i)) - psi;
end;
toc
%temp=ones(size(imspace))*psi;
%değişkenleri diske yaz
save zeroMeanSpace zeroMeanSpace;
%hafızayı bosalt
clear imSpace;

%% 6-PCA uygula
fprintf('6-PCA\n');
tic
L = zeroMeanSpace' * zeroMeanSpace;         % Turk-Pentland PCA implemetasyonu

%++++++++++++++1. alternatif çözüm++++++++++++++++++++++
%save L L;

%[eigVecs, eigVals] = eigs(L,subDim);

%save eigVecs eigVecs;

%pcaEigVals = diag(eigVals);
```

```
%pcaEigVecs = zeroMeanSpace*eigVecs;
%++++++++++++++++++++++++++++++++++++++++++++++++

%++++++++++++++2. alternatif çözüm++++++++++++++++++++++
[eigVecs, eigVals] = eig(L);

diagonal = diag(eigVals);
[diagonal, index] = sort(diagonal);
index = flipud(index);

pcaEigVals = zeros(size(eigVals));
for i = 1 : size(eigVals, 1)
    pcaEigVals(i, i) = eigVals(index(i), index(i));
    pcaEigVecs(:, i) = eigVecs(:, index(i));
end;

pcaEigVals = diag(pcaEigVals);
pcaEigVals = pcaEigVals / (numIm-1);
pcaEigVals = pcaEigVals(1 : subDim);        % Sadece yüksek olanlari al

pcaEigVecs = pcaEigVecs(:,1 : subDim);        % Sadece büyük olanlari al

pcaEigVecs = zeroMeanSpace*pcaEigVecs;
%++++++++++++++++++++++++++++++++++++++++++++++++
toc

fprintf('6.1-eigenvectors size:');
size(pcaEigVecs)
fprintf('6.2-eigenvalues size:');
size(pcaEigVals)
```

```
save pcaEigVecs pcaEigVecs;

%% 7-Normalizasyon islemi
fprintf('7-Normalisation to unit length\n')
tic
for i = 1 : size(pcaEigVecs,2)
    pcaEigVecs(:, i) = pcaEigVecs(:, i) / norm(pcaEigVecs(:, i));
end;
toc

%% 8-Boyult azaltma
fprintf('8-Creating lower dimensional subspace\n');
w = pcaEigVecs'*zeroMeanSpace;
save w w;

fprintf('8.1-Subspace size:');
size(w)

%Hafizayi temizle
%clear all;
uNew=w;
```

Ek 1.2 RCA() Fonksiyonu

PCA projeksiyon matrisini girdi olarak alıp RCA algoritmasını uygulayan ve beyazlatıcı dönüşüm yapan fonksiyondur.

```
%------------------------------------------------------------
%Tarih:21 Temmuz 2008
%Gelistiren:Bilal Karaduman
%Aciklama: RCA uygula
%Referans: Aharon Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall,
%“Learning a Mahalonobis Metric from Equivalance Constraints”, in Journal of Machine
Learning Research 6 (2005), pp. 937-965
%------------------------------------------------------------
%Kullanim sekli
%Girdiler :
% data - N*D boyutundaki izdusum matrisi. N matristeki nokta sayisi,
%        D data'nin boyutudur

% chunks - N*1 boyutundaki bilgicik'i gösteren matris
%          i. noktdaki -1 i'nin hic bir bilgicike ait olmadigini
%          i. noktadaki j tamsayisi bu i indisli yanbilgicik'in j
%          bilgicigine ait oldugunu gösterir.
%          Bilgicik indisleri 1:bilgicik_sayisi seklinde olmalidir

% useD  - opsiyoneldir. Verilmedigi zaman RCA orjinal boyutlariyla yapilir.
%         B full rank'tir. useD verildigi zaman RCA LDA ile useD'da verilen
%         boyutlara indirilir.
```

```
% Ciktilar :

% B - RCA'da önerilen Mahalonobis data matrisi
% RCA - girdi olarak verilen datanin RCA dönüsümü
%           Boyutlari (orjinal data boyutu)*(useD)
% newData - RCA dönüsümünden sonraki islenmemis data

% Görüntülerdeki her bir x1,x2 noktasi için y1,y2:
%  (x2-x1)'*B*(x2-x1) = || (x2-x1)*A ||^2 = || y2-y1||^2
%------------------------------------------------------------
%% RCA fonksiyonu
function [ B, RCA, newData ] =rca(data,chunks,useD)

[ n,d ] = size(data);

if ~exist('useD','var')
   useD=d;
end

% ortalamayi cikar
TM=mean(data);
data=data-ones(n,1)*TM;

% bilgiciklerin ortalamasini hesapla ve datayi ortaya hizala
S= max(chunks);
Cdata=[];
AllInds=[];
for i=1:S
   inds=find(chunks==i);
   M(i,:)=mean(data(inds,:),1);
```

```
    %her bilgicikten ait oldugu bilgicik kümesinin ortalamasini cikar.
    Cdata(end+1:end+length(inds),:)=data(inds,:)-ones(length(inds),1)*M(i,:);
    AllInds=[ AllInds ,inds ];
end

% Kovaryans matrisi hesapla
InnerCov=cov(Cdata,1);

%opsiyonel
if useD<d
    TotalCov=cov(data(AllInds,:));    % Bilgicikle verilen noktalara göre toplam kovaryans
matrisi hesapla
    % TotalCov= cov(data);
    [ V D ]=eig(inv(TotalCov)*InnerCov);
    [ OD inds ]=sort(diag(D));
    V=V(:,inds);    %vektörleri azalan sekilde s?rala
    A=V(:,1:useD);
    InnerCov = A'*InnerCov*A;
else
    A=eye(d);
end

%RCA: Beyazlatici donusum
RCA=A*(InnerCov)^(-0.5);
 newData = ( data+ones(n,1)*TM) * RCA;
B=RCA *RCA' ;
```

Ek 1.3 SplitMatrix() Fonksiyonu

Girdi olarak verilen sütun matris formundaki orjinal görüntüyü dört altörneğe ayıran fonksiyondur.

```
%-------------------------------------------------------
%Tarih:20 Mart, 2008
%Gelistiren:Bilal Karaduman
%Aciklama: Girdi olarak verilen matrisi altörnekleme yaparak 4 parcaya
%ayirir.
%-------------------------------------------------------
function s=splitMatrix(input)
%% girdi matrisinin boyutlarini al
input=input(:,:,1);
[irow icol]=size(input);
% figure(1);
% imshow(input);
% title('Orjinal imaj (128x128)');

%% altörnekleme yap
input=input(:,:,1);
A=[];
for i=0:1
   temp1=reshape(downsample(input,2,i)',irow*icol/2,1);
   for j=0:1
      temp=downsample(temp1(:,1),2,j);
      A=[A temp];
   end
end
```

```
%% altörneklenen imajlari ciz
% figure(2);
% title('Alt örneklenmis imajlar');
% for i=1:4
%    subplot(2,2,i);
%    temp=reshape(A(:,i),irow/2,icol/2)';
%    imshow(temp);
%    ttl=strcat(int2str(i),'. altörnek (64x64)');
%     title(ttl);
%    drawnow;
% end
s=A;
```

Ek 1.4 K-Means() Fonksiyonu

RCA() fonksiyonunun çıktısı olan beyazlatma dönüşümü yapılmış izdüşüm matrisini girdi olarak alıp k-ortalama kümelemesi uygulayan fonksiyondur.

```
%-------------------------------------------------------
%Tarih: 9 Haziran, 2008
%Gelistiren:Bilal Karaduman
%Aciklama: K-Komsuluk Siniflandirma algoritmasi uygulayip siniflandirma
%sonuclarini cizer
%-------------------------------------------------------

function [idx, ctrs]=kMeans(pcaProj,numClass,cntIteration,numCursor,numLoop, ttl)
%% Kolonlari kümelere dagit

opts = statset('Display','final');

pcaProj=pcaProj';
[idx,ctrs] =
kmeans(pcaProj,numClass,'Distance','sqEuc','Replicates',cntIteration,'Options',opts);


%% Siniflari ciz
subplot(ceil(sqrt(numLoop)),ceil(numLoop/ceil(sqrt(numLoop))),numCursor)

for i=1:numClass
   plot(pcaProj(idx==i,1),pcaProj(idx==i,2),'+');
   hold all;
end
```

```
plot(ctrs(:,1),ctrs(:,2),'kx','MarkerSize',9,'LineWidth',2);
plot(ctrs(:,1),ctrs(:,2),'ko','MarkerSize',9,'LineWidth',2);
title(ttl)
grid on;
hold off;

%legend('Cluster 1','Cluster 2','Cluster 3','Centroids','Location','NW')
```

Ek 1.5 ExecFaceReco() Fonksiyonu

Tezde önerilen modelde belirtilen adımları birleştiren ve yardımcı fonksiyonları çağıran ana kod bloğudur.

```
%-----------------------------------------------------------
%Tarih:9 Temmuz 2008
%Gelistiren:Bilal Karaduman
%Aciklama: ?lgili siniflara atifta bulunan ve onlari calistiran ana kod
%blogu
%-----------------------------------------------------------
function [idx,
ctrs]=ExecFaceReco(numClass,numIteration,numSubDim,boolRCA,boolDownSample,nu
mCursor, numLoop)
tic
%% Uygulama girdileri
%numClass=10; % okunacak klasör sayisini gösterir
%numIteration=10; % k-komsuluk fonksiyonundaki tekrar sayisini gösterir
%numSubDim=390; % boyut azaltma yapilacak boyutu gösterir

save params numClass numIteration numSubDim boolRCA boolDownSample numCursor
numLoop;

%% A- Yüz görüntülerini DB'den oku, PCA uygula ve izdü?üm matrisini geri getir
fprintf('Start...\n');
fprintf('A-Apply PCA and do projection\n');
tic;
fprintf('-------------------------------------------------\n');
```

```
pcaProj=pca(numSubDim,numClass,boolDownSample);%kullanim bicimi -->
pca(subDim,noOfFoldersToBeProcessed);
fprintf('A.1-Projection matrix size:\n');
size(pcaProj)

save pcaProj pcaProj;

%hafizayi temizle
clear all;
fprintf('-------------------------------------------------\n');
toc;

%% B- RCA uygula ve uzaklik matrisini hesapla
% Rca uygulanacak mi?
load params;
load pcaProj;

if (boolRCA==1)

   fprintf('\n\nB-Apply RCA and compute distance matrix\n');
   fprintf('-------------------------------------------------\n');

   chunklet=[];
    tic;
   sampleNumber=1;
   if (boolDownSample==1)
      sampleNumber=4;
   end

   for i=1:numClass
```

```
        chunklet=[chunklet; i*ones(sampleNumber*10,1)];
    end
    [b, m, k]=rca(pcaProj',chunklet);%fonksiyonun kullanimi --> rca(rowVectorData,
chunklets)
    toc;
    fprintf('-----------------------------------------------\n');
    save k k;
    inputSet=k;
else
    inputSet=pcaProj';
end

%% C- K-Komsuluk siniflandirmasi uygula ve siniflandirma grafiklerini ciz

fprintf('\n\nC-Do K-Means Clustering and plot the clusters\n');
fprintf('-----------------------------------------------\n');
tic;

    ttl=strcat('#Sinif=',int2str(numClass));
    ttl=strcat(ttl,', Özvektör = % ');
    ttl=strcat(ttl, int2str(numSubDim/(4*numClass)*10));
size(inputSet);
[idx, ctrs]=kMeans(inputSet',numClass,numIteration,numCursor,numLoop, ttl);
toc;
fprintf('-----------------------------------------------\n');

%% Son
toc
fprintf('Completed!\n');
```

Ek 1.6 BulkAnalyser() Fonksiyonu

ExecFaceReco() fonksiyonunu değişen özvektör, algoritma ve uzaklık metrikleri parametreleriyle çağırıp çalıştıran koddur.

```
Function BulkAnalyser()
clear all;
close all;
clc;

numClass=10;
numIteration=10;
boolRCA=1;
boolDownSample=1;

if (boolDownSample~=1)
   numSample=numClass*1;
else
   numSample=numClass*4;
end

numSubDim=[1*numSample 2*numSample 3*numSample 6*numSample 9*numSample ];

numLoop=size(numSubDim,2);
figure(1);

idxPool=[];
for numCursor=1:numLoop
```

```
[idx,ctrs]=ExecFaceReco(numClass,numIteration,numSubDim(1,numCursor),boolRCA,bo
olDownSample,numCursor, numLoop);
    idxPool=[idxPool idx];
end
```

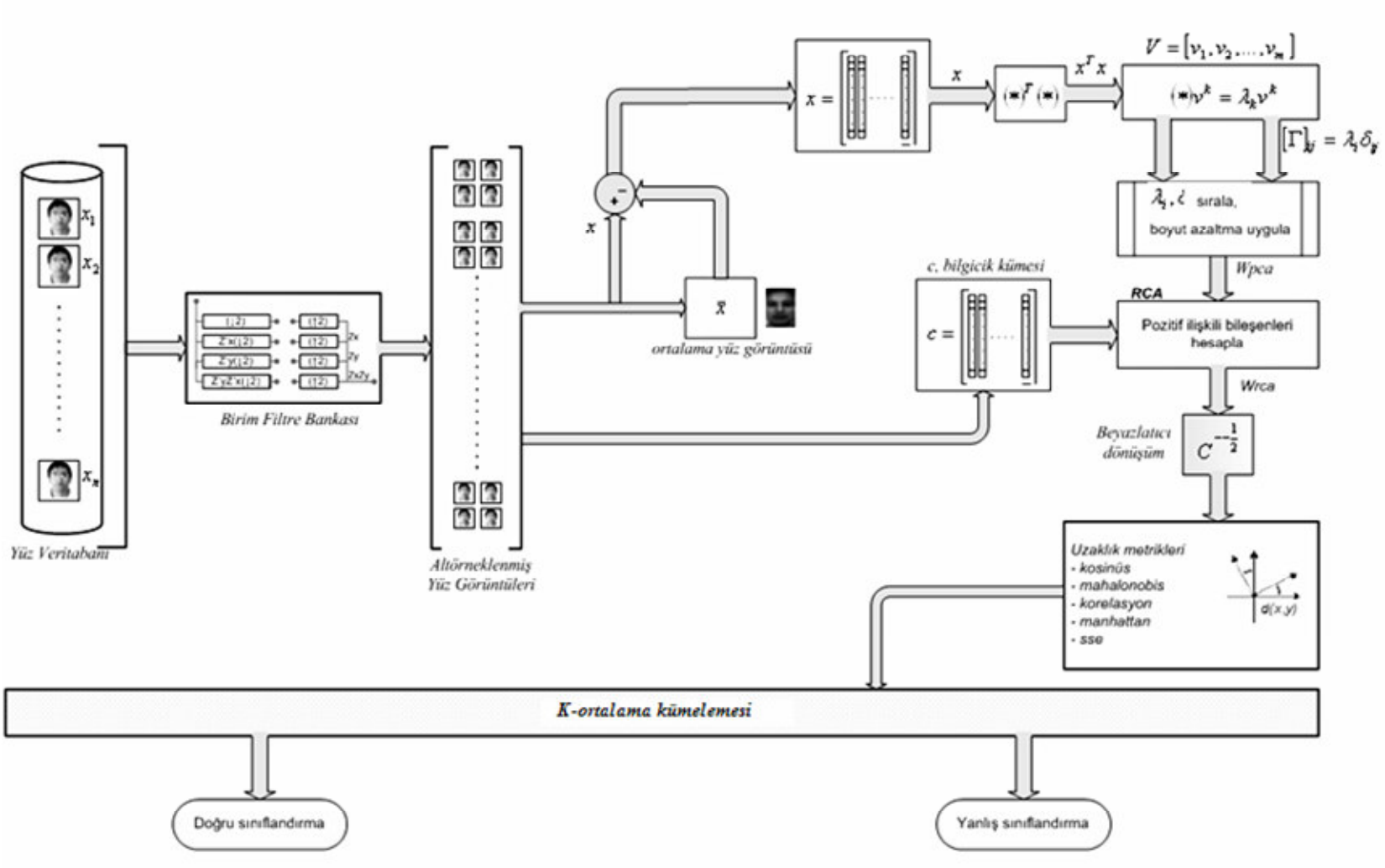
Ek 1.7 AnalyseScope() Fonksiyonu

BulkAnalyser() fonksiyonunun çıktısı olan sınıflandırma başarım matrisini girdi olarak alıp yüz görüntülerinin sınıflar arasındaki dağılımının analizini yapan fonksiyondur.

```
Function analyseScope()
for i=1:numClass
   scope=find(idxPool(:,4)==i);
   for mm=1:numClass
      a=(mm-1)*40+1;
      u=a + 40;
         c=0;
      for j=1:size(scope,1)
        if scope(j,1)>=a && scope(j,1)<u
            c=c+1;
        end
      end

      if (c>0)
        fprintf('class %d- scope %d --> #%d\n',i,mm,c);
      end
   end
end
```

Ek 2: Tasarlanan Modelin Blok Diyagram Gösterimi



Şekil Ek 2.1 Bu tezde önerilen modele ait algoritmanın blok diyagramı

Ek 3: Sınıflandırma Başarı Oranının Kullanılan Özvektör Sayısıyla Değişimi

Aşağıda ORL veritabanındaki yüz görüntüleri kullanılarak yapılan benzetimlere ait doğru tanıma oranları verilmiştir.

Tablo Ek 3.1 Sınıflandırma başarı oranları

% Öznitelik sayısı	*PCA (%)*					*PCA+RCA (%)*				
	Kosinüs	***Mahalonobis***	***SSE***	***Manhattan***	***Korelasyon***	***Kosinüs***	***Mahalonobis***	***SSE***	***Manhattan***	***Korelasyon***
5	26	48	20	10	27	16	53	13	26	32
10	47	66	37	36	41	67	79	71	68	45
15	57	69	45	59	44	85	77	74	71	68
20	62	70	54	68	63	87	82	84	73	74
25	67	71	53	69	69	88	85	84	79	75
30	78	71	57	71	72	89	84	84	83	75
35	79	71	63	75	74	90	85	84	84	81
40	77	72	65	78	76	90	84	81	85	82
45	77	73	68	79	78	89	84	79	85	84
50	78	73	71	79	78	89	86	82	86	85
55	80	73	75	78	76	89	85	84	86	87
60	78	74	76	78	76	90	83	85	87	87
65	78	74	77	78	76	91	84	86	88	87
70	76	75	78	78	77	91	87	87	88	87
75	78	75	79	77	77	91	87	87	88	88
80	80	75	78	78	77	91	87	88	87	89
85	80	73	78	78	77	92	87	88	87	88
90	80	74	79	78	79	92	87	88	87	88
95	81	76	79	79	78	92	87	87	87	86

ÖZGEÇMİŞ

Doğum Yeri:	Ankara	
Doğum Tarihi:	1983	
Lise:	1994 - 2001	Ankara Mamak Çağrıbey Anadolu Lisesi
Lisans:	2001 - 2005	Kırıkkale Üniversitesi Mühendislik Fakültesi Elektrik Elektronik Mühendisliği Bölümü
Yüksek Lisans:	2005 - 2008	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Müh. Ana Bilim Dalı Haberleşme Programı
Çalıştığı Kurum:	2005 -	Sony Europe İstanbul Yazılım Geliştirme Birimi, Yazılım Performansı Uzmanı

TEZ ONAY SAYFASI

YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİCİK TABANLI İLİŞKİLİ BİLEŞEN ANALİZİYLE EĞİTİCİSİZ YÜZ TANIMA MODELİ

Elektrik ve Elektronik Müh. Bilâl KARADUMAN

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı Haberleşme Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Yrd. Doç. Dr. Lütfiye DURAK

İSTANBUL, 2008

Jüri : Yrd. Doç. Dr. Songül Albayrak

Prof. Dr. Vedat Tavşanoğlu

KAYNAKLAR

1 Wechsler, H., Philips, P. et.al., (1996), Face Recognition: From Theory to Applications, Springer-Verlag

2 Chellappa, R., Wilson, C.L. et.al., (1995), *"Human and machine recognition of faces: A survey"*, Proc. IEEE, vol 83, pp. 705-740

3 Eyematic Interfaces Inc., *http://www.eyematic.com*

4 Identix, Minnetonka, MN, *http://www.identix.com*

5 Lu, X., (2004), *"Image Analysis for Face Recognition"*, Michigan State University,Dept. of Computer Science & Engineering

6 Serbes, A. ve Durak, L., (2008), *"Düşük çözünürlüklü görüntüler üzerinden yüz tanıma"*, ASYU 2008 Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu

7 Torres, C., Perrot, P. et.al., (2007),*"Face recognition: From biometrics to forensic applications"*, Biometrical Feature Identification and Analysis Conference

8 International Biometric Group, *http://www.biometricgroup.com*

9 Hietmeyer, R., (2000), *"Biometric identification promises fast and secure processing of airline apssengers",* The Int'l Civil Aviation Organization Journal, vol. 55, no.9, pp. 10-11

10 Perlikabas, V., (2003), *"Distance measures for PCA-based face recognition"*, Image Processing and Multimedia Laboratory, Kaunas University of Technology

11 Lanitis, A., Taylor, C. J.et.al., (2002), *"Towards automatic simulation of ageing effects on face images*", IEEE Trans. Pattern Analysis and machine Intelligence, vol. 24, no. 4, pp. 442-455,

12 Belhumeur, P., Hespanha, J.P. et.al., (1997), *"Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection"*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No 7, pp. 711

13 Moses, Y., Adini, Y. et.al., (1994), *"Face Recognition: The Problem of Compensating for Changes in Illumination Direction"*, European Conf. Computer Vision, pp. 286-296

14 Zhao, W., Chellappa, R., (2000), "IlluminationInsensitive Face Recognition Using Symmetric Shape-from-Shading," IEEE Conference on Computer Vision and Pattern Recognition

15 Chellappa, R., Wilson, C. L. et.al., (1995), "Human and machine recognition of faces: A survey", Proc. of IEEE, 83(5):705-740

16 Turk, M., Pentland, A., (1991), *"Eigenfaces for recognition"*, Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71-86

17 Yambor, W. S., Draper, B. et.al, (2000), "Analyzing PCA-based face recognition algorithms:Eigenvector selection and distance measures", Computer Science Department, Colorado State University

18 Kim, K., (1996), "Face recognition using principal component analysis", Deparment of Computer Science, University of Maryland, USA

19 Smith, L. I., (2002), *"A tutorial on Principal Component Analysis"*, pp. 13-22

20 Teknomo, K., Discriminant Analysis Tutorial,
http://people.revoledu.com/kardi/tutorial/LDA/

21 Decision Trees, Linear Discriminant Analysis, *http://www.dtreg.com/lda.htm*

22 Welling, M., (2006), *"Fisher Linear Discriminant Analysis"*, University of Toronto, Department of Computer Science

23 Bar-Hillel, A., Hertz, T. et.al., (2005), *"Learning a Mahalonobis Metric from Equivalance Constraints"*, Journal of Machine Learning Research 6, pp. 937-965

24 Wu, F., Zhou, Y. et. al., (2004), **"***Self-enhanced relevant component analysis with side-information and unlabeled data"***,** Neural Networks, and Proceedings IEEE International Joint Conference, 25-29, vol. 2, pp. 1347 - 1351

25 Yeung, D., Chang, H., (2005), *"Extending the relevant component analysis for metric learning using both positive and negative equivalence constraints"*, Department of Computer Science, Hong Kong University of Science and Technology

26 Shental, N., Hertz, T. et.al., (2002), *"Adjustment learning and relevant component analysis"*, Lecture notes in Computer Science, 2353:767-792

27 Bow, S. T., (2002), *"Pattern recognition and Image preporecessing (2nd edition)"*, Marcel-Dekker Press, New York

28 Shental, N., Bar-Hillel, A. et.al., (2004), *"Computing Gaussian mixture models with em using equivalence constraints"*, In Sebastian Thrun, Lawrence Saul, and Bernhard

Schölkopf, editors, Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA

29 Wagstaff, K., Cardie, C. et.al., (2001), "Constrained K-means clustering with background knowledge", In *Proc. 18th International Conf. on Machine Learning*, pages 577–584, Morgan Kaufmann, San Francisco, CA

30 Kanade, T., (1973), *"Picture Processing by Computer Complex and Recognition of Human Faces"*, PhD thesis, Kyoto University

31 Cootes, T.F., Edwards, G.J. et.al., (1998), *"Active appereance models",* Proc. European Conference on Computer Vision, vol. 2, pp. 484-498

32 Teknomo, K., K-Means Clustering, *http://people.revoledu.com/kardi/tutorial/kMean/*

33 Wikipedia, K-Means algorithm*, http://en.wikipedia.org/wiki/K-means_algorithm*

34 Jin, Y., Ruan, Q., (2006), *"Face recognition using assembled matrix distance metric based 2DLDA algorithm"*, Beijing Jiatong University, Institute of Information Science

35 Mochihashi, D., Kikui, G. et.al., (2004), *"Learning nonstrutural distance metric by minimum cluster distortions"*, ATR Spoken Language Translation Research Laboratories and Center for Advanced Information Technology, Tokushima University

36 Malisiewicz, T., (2007), *"Supervised Distance Metric Learning"*, CMU's Computer Vision Misc-read Reading Group

37 Emran, S.M., Ye N., (2001), *"Robustness of Canberra Metric in Computer Intrusion Detection"*, Proceedings of the 2001 IEEE, Workshop on Information Assurance and Security

38 Gross, R., Stan, Z. L. et.al,. (2005), *"Face Database*s, *Handbook of Face Recognition",* Springer-Verlag, 22 pages

39 ORL face database official homepage,
http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

40 Martinez ,A.M., Benavente, R.., (1998), *"The AR Face Database. CVC Technical Report #24"*

41 Fukunaga, K., (2002), *"Introduction to statistical pattern recognition (second edition)*", Academic Press, New York

42 Strang, G., Nyugen, T., (1997), *"Wavelets and filter banks"*, Wellesley-Cambridge Press, Massachusetts

43 The Facial Recognition Technology (FERET) official homepage, *http://www.itl.nist.gov/iad/humanid/feret/feret_master.html*