T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK ANABİLİM DALI
2015-YL-044

GÜNCEL METOTLARLA RESİM SINIFLANDIRMA

Ömer KALFA

Danışmanı:
Yrd. Doç. Dr. Rıfat AŞLIYAN

AYDIN

T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Matematik Anabilim Dalı Yüksek Lisans Programı öğrencisi Ömer KALFA tarafından hazırlanan "Güncel Metotlarla Resim Sınıflandırma" başlıklı tez, 15.07.2015 tarihinde yapılan savunma sonucunda aşağıda isimleri bulunan jüri üyelerince kabul edilmiştir.

	Ünvanı, Adı Soyadı	Kurumu	İmzası
Başkan :	Yrd. Doç. Dr. Rıfat AŞLIYAN	ADÜ Fen Edebiyat Fak.	.....
Üye :	Yrd. Doç. Dr. Korhan GÜNEL	ADÜ Fen Edebiyat Fak.	.....
Üye :	Yrd. Doç. Dr. Refet POLAT	Yaşar Üni. Fen Edeb. Fak.	.....

Jüri üyeleri tarafından kabul edilen bu Yüksek Lisans tezi, Enstitü Yönetim Kurulunun …… sayılı kararıyla …… tarihinde onaylanmıştır.

Prof. Dr. Aydın ÜNAY
Enstitü Müdürü

T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Bu tezde sunulan tüm bilgi ve sonuçların, bilimsel yöntemlerle yürütülen gerçek deney ve gözlemler çerçevesinde tarafımdan elde edildiğini, çalışmada bana ait olmayan tüm veri, düşünce, sonuç ve bilgilere bilimsel etik kuralların gereği olarak eksiksiz şekilde uygun atıf yaptığımı ve kaynak göstererek belirttiğimi beyan ederim.

15/07/2015

Ömer KALFA

ÖZET

GÜNCEL METOTLARLA RESİM SINIFLANDIRMA

Ömer KALFA

Yüksek Lisans Tezi, Matematik Anabilim Dalı
Tez Danışmanı: Yrd. Doç. Dr. Rıfat AŞLIYAN
2015, 89 sayfa

Tekstil endüstrisinin çok hızlı ilerlemesiyle birlikte tekstil desenlerini içeren tekstil resimlerinin sayısı büyük miktarlara ulaşmıştır. Bu yüzden, tekstil resimlerinin otomatik olarak sınıflandırılması ihtiyacı vardır. Bu tez çalışmasında, büyük sayıdaki tekstil resimlerinin otomatik olarak sınıflandırılmasını sağlayacak sistemler geliştirilmiştir. Bu sistemler, Destek Vektör Makinesi, Çok Katmanlı Algılayıcı, K-En Yakın Komşu ve Naive Bayes metotlarıyla oluşturulmuştur. Eğitim ve test aşamasında kullanılmak üzere iki veri seti kullanılmıştır.

Eğitim ve test veri setinde "Çiçekli", "Puantiyeli", "Yatay Çizgili", "Dikey Çizgili", "Ekoseli", "45 Derece Çizgili" ve "135 Derece Çizgili" tekstil desenleri olmak üzere yedi tane sınıf vardır. Bu çalışmadaki sistemler, önişleme, öznitelik çıkarımı, sistemin eğitilmesi ve test edilmesi safhalarından oluşmaktadır. Çalışmanın önişleme safhasında, bütün tekstil resimleri siyah-beyaz resimlere dönüştürülür. Sobel, Prewitt, LoG ve Zero-Cross filtreleriyle tekstil resimlerinin kenar tespit işlemi yapılır. Bununla birlikte inceltme (iskeletleme) işlemi de uygulanır. Öznitelik çıkarımı safhasında, her bir tekstil resmi için 2x2, 3x3 ve 4x4 çekirdek matrislerinin resim içindeki frekansları hesaplanır ve resmin toplam frekansına bölünmesiyle normalize edilir. Böylece, her tekstil resminin öznitelik vektörü elde edilmiş olur. Eğitim safhasında ise kullanılan metoda göre her sınıf içindeki öznitelik vektörleri eğitilmek suretiyle, her sınıfı temsil eden modeller oluşturulur. Test safhasında ise test veri setindeki tekstil desenleri ve eğitim safhasında oluşturulan sınıf modelleri kullanılarak sistemin başarısı hesaplanır. Bunun için Doğruluk ve F-Ölçüsü kullanılmıştır. Geliştirilen sistemler karşılaştırılarak en başarılı sistem tespit edilmiştir.

Anahtar Sözcükler: Tekstil Resimleri, K-En Yakın Komşu, Çok Katmanlı Algılayıcı, Destek Vektör Makinesi, Naive Bayes

ABSTRACT

IMAGE CATEGORIZATION USING STATE-OF-THE-ART METHODS

Ömer KALFA

M.Sc. Thesis, Department of Mathematics
Supervisor: Assist. Prof. Dr. Rıfat AŞLIYAN
2015, 89 pages

The number of textile pictures containing textile patterns has increased very much with rapid progression of the textile industry. Thus, the automatic classification of the textile image is necessary. In this study, some systems were developed to provide automatic classification of a large number of textile pictures. These systems were designed with Support Vector Machine, Multilayer Perceptron, K-Nearest Neighbor and Naive Bayes classification methods. We have used two datasets for training and testing stage. There are seven textile design classes as "Flowery", "Spotted", "Horizontal Striped", "Vertical Striped", "Plaided", "45 Degree Striped" and "135 Degrees Striped" in the training and testing datasets. The systems in this study are composed of preprocessing, feature extraction, training and testing of the system phases. All textile pictures are converted to black-and-white images in the preprocessing stage. Edge detection process of textile pictures is made by the edge detection filters as Prewitt, Sobel, LoG and Zero-Cross. In addition, the thinning (skeletonization) process is also applied. In the feature extraction stage, the frequencies of 2x2, 3x3 and 4x4 kernel matrices in the image are calculated for each textile pictures and normalized by dividing the total frequencies of the picture. Thus, the attribute vectors of each textile image are obtained. In the training phase according to the methods, the models representing each classes are composed by training all attribute vectors in each class. In the testing phase, the success of the system is calculated by using textile patterns in testing datasets and the class models developed in the training phase. Accuracy and F-Measure values are used for computing the success of the systems. The most successful system has been determined by comparing the developed systems.

Keywords: Textile Image, K-Nearest Neighbor, Multi-layer Perceptron, Support Vector Machines, Naive Bayes

ÖNSÖZ

Eğitim hayatım boyunca maddi manevi her türlü imkanı sağlayan anneme ve babama, bana her zaman destek olan sevgili eşime ve kardeşime, her yönüyle takdir ettiğim ve örnek aldığım saygıdeğer hocam Yrd. Doç. Dr. Rıfat AŞLIYAN'a ve hiç bir zaman yardımlarını esirgemeyen çok değerli hocam Yrd. Doç. Dr. Korhan GÜNEL'e teşekkür ederim.

Bu tez çalışması, Adnan Menderes Üniversitesi Bilimsel Araştırma Projeleri (BAP) tarafından "Güncel Metotlarla Resim Sınıflandırma" yüksek lisans tez projesi adıyla ve FEF-14030 proje numarasıyla desteklenmiştir. Katkılarından dolayı teşekkür ederiz.

Ömer KALFA

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

EKLER DİZİNİ

1 GİRİŞ

Fiziki ya da sanal ortamlarda her an çok büyük miktarlarda resimler ile karşı karşıyayız. Çevremizdeki zaten çok fazla sayıda olan resimler hızlı bir şekilde artarken, bunları işleyen insan sayısı sınırlı kalmaktadır. Resimlerin insanlar tarafından türlerine göre ayrılması, sınıflanması ve listelenmesi işi yüksek maliyete ve zaman kaybına neden olurken; insanın tabiatı gereği çoğu zaman önemli oranda da hataya sahip olarak gerçekleştirilmektedir. Bu duruma ek olarak zaten çok zor oluşturulan veri tabanından istenilen bir resme ulaşılabilmesi zahmetli bir iş olarak önümüze çıkar. Tam da bu noktada resimleri en az hata ile tanıyan, sınıflandıran ve listeleyerek veri tabanı oluşturan; ihtiyaç duyulduğu anda ise resmin kolayca bulunmasını sağlayan bir otomasyon sistemine ihtiyaç duyulmaktadır. Böyle bir otomasyon sistemi maliyeti de zaman kaybını da en aza indirecektir.

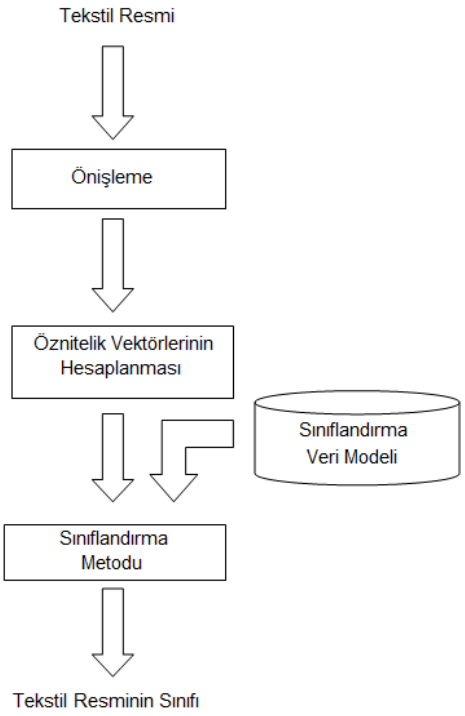
Bu tezde, tekstil desenlerinin sınıflandırılması (Aşlıyan ve Alpkoçak, 2002; Aşlıyan 2010; Ulvi vd., 2013) üzerine çalışmalar yapılmıştır. Tekstil desenlerinin sınıflandırılması, resim sınıflandırmanın bir alt konusudur. Resim sınıflandırma işlemi resimlerin içeriklerine göre yada resmi ifade eden metinlere göre yapılır. Fakat en yaygın resim sınıflandırma resim içeriklerine (Content-Based Image Classification) göre yapılmaktadır (Geyers vd., 2000). Tekstil desenlerinin sınıflandırılması şu şekilde olmaktadır. Bir tekstil resminin içindeki özelliklere bakılarak ne tür bir tekstil resmi olduğuna karar verilir. Yani, çiçekli bir resmin çiçekli sınıfına, puantiyeli bir resmin puantiyeli sınıfına atanması, tekstil desenlerinin sınıflandırılması konusunun amacıdır. Böylece, bütün tekstil desenleri uygun bir şekilde sınıflandırılırsa hem müşteriler hem de desinatörler istedikleri tekstil ürününe en doğru ve en hızlı bir şekilde ulaşmış olurlar.

Tekstil desenlerinin sınıflandırılması işlemi yapılırken resim sınıflandırmada en yaygın ve en güncel sınıflandırma metotlarından Destek Vektör Makinesi (DVM) (Burges, 1998; Sezer vd., 2005), K-En Yakın Komşu (K-NN) (Akkuş ve Güvenir, 1996; Aşlıyan ve Alpkoçak, 2002; Aşlıyan 2010; Özkan, 2013), Naive Bayes (Kim vd., 2002; Özkan, 2013) ve Yapay Sinir Ağlarından Çok Katmanlı Algılayıcı (ÇKA) (Haykin, 1994; Kohonen, 1995; Chtioui vd.,1997; Elmas, 2003; Efe ve Kaynak, 2000; Öztemel, 2006) ağı kullanılmıştır.

Bu tez çalışmasının yapılmasında eğitim ve test aşamasında kullanılmak üzere iki veri seti oluşturulmuştur. Eğitim ve test aşamasında yani eğitim ve test veri setinde kullanılmak üzere "Çiçekli", "Puantiyeli", "Yatay Çizgili", "Dikey Çizgili", "Ekoseli", "45 Derece Çizgili" ve "135 Derece Çizgili" gibi sınıfların her biri için tekstil desenlerinin içeren resimler elde edilmiştir. Eğitim veri seti, oluşturulacak sınıflandırma sisteminin eğitilmesinde kullanıldı. Test veri seti ise eğitilmiş sistemlerin ne kadar başarılı sınıflandırma gerçekleştirdiğinin tespiti ve sistemlerin karşılaştırılması için oluşturulmuştur.

Resim sınıflandırmada (Yılmaz, 2013), sisteme verilen resimlerin sayısının önemli bir rolü vardır. Resimlerin gereğinden fazla olması sistemin öğrenmesini zorlaştırmakta, gereğinden az olması da sistemin hata oranını arttırmaktadır.

Tekstil desenlerinin sınıflandırılmasının genel yapısı Şekil 1'de gösterilmektedir. İlk olarak, tekstil resimleri sisteme alınarak önişlemden geçirilir (Gonzalez ve Woods, 2008; Timur ve Sarı, 2010; Bilgi, 2012; Klette, 2014). Önişleme safhasında, renkli veya gri tonlamalı resimler siyah-beyaz resimlere dönüştürülür. Ardından, Sobel (Sobel ve Feldman, 1968; Boyle ve Thomas, 1988; Anonim, 2004; Dong vd., 2005; Bilgi, 2012), Prewitt (Prewitt, 1970; Heijden, 1995; Konishi vd., 2003; Bilgi, 2012), LoG (Özkul, 1995; Anonim, 2006; Bilgi, 2012) ve Zero-Cross (Clark, 1989; Anonim, 2004; Bilgi, 2012) kenar tespit filtreleriyle tekstil desenlerinin kenarları tespit edilir. Aynı zamanda görüntüyü inceltmek için iskeletleştirme (skeletonization) (Blum, 1967; Aşlıyan, 2002; Dimitrov vd., 2000; Atay, 2014) işlemi de uygulanır. 2x2, 3x3 ve 4x4 çekirdek matrislerinin her bir resim içindeki frekansları hesaplanıp normalize edilerek resim öznitelik vektörleri oluşturulur.



Şekil 1.1. Tekstil desenlerinin sınıflandırılmasının genel yapısı

Eğitim veri setindeki resimlerin çekirdek matrisi frekansları hesaplandıktan sonra her sınıf için öznitelik vektör veri seti oluşturulur. Daha sonra sınıfı belli olmayan tekstil resimleri sınıflandırma metotları kullanılarak resmin sınıfı belirlenir.

Bu zamana kadar resim sınıflandırma konusunda pek çok proje ve çalışmalar yapılmıştır. Genel haliyle bu projeler bilgisayar görüntü teknolojisi ile ilgilidir. Resim sınıflandırma alanındaki çalışmaların sonucunda meydana getirilen sistemlerin bazıları sağlık alanında kanser gibi hastalıkların belirlenmesinde, askeri alanda silah yerlerinin tespiti gibi konularda, otomobillerde sürücüye yardımcı olması için kullanılır.

Bu çalışmamız hem tekstil alanındaki üretici ve tüketicilere fayda sağlayacak hem de bu alandaki diğer çalışmalara yardımcı olacaktır. Bu çalışma sayesinde tekstil mühendisleri yada tasarımcıları ihtiyaç duydukları desenlere daha rahat ulaşabilecek, daha verimli çalışabilecek ve daha hızlı sınıflandırma yaparak veritabanları oluşturabileceklerdir. Tüketiciler ise istedikleri desenlerdeki ürünlere çok daha kolay bir şekilde ulaşabileceklerdir.

İkinci bölümde kullanılan sınıflandırma metotları ile dijital görüntü işleme ve görüntülerdeki nesnelerin kenar çıkarım yöntemleri açıklanmıştır. Üçüncü bölümde sistemlerin uygulanması sonucunda çıkan Doğruluk ve F-Ölçüsü değerleri tablolar şeklinde verilmiş ve en başarılı yöntemler tespit edilmiştir. En son bölümde, genel olarak test sonuçlarına göre kullanılan metotların kıyaslaması yapılmıştır.

2. MATERYAL VE YÖNTEM

2.1. Sınıflandırma

Farklı sınıflar ve her sınıfa ait farklı gözlemlerimizin olduğunu varsayalım. İlk defa karşılaştığımız yeni bir gözlemin özelliklerini önceki sınıfını bildiğimiz gözlemlerin özellikleri ile kıyaslayarak bir sınıfa atama işlemine sınıflandırma işlemi denir (Yılmaz, 2013).

Matematiksel ifadeyle; (+1) ve (-1) sınıflar olmak üzere $y_i \in \{-1, +1\}$ sınıf etiketi ve $\boldsymbol{x_i} \in X$ kümesine ait bir gözlem olsun. Denklem 2.1'deki çıktıların elde edildiğini varsayalım.

$$(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \dots, (\boldsymbol{x_n}, y_n) \in X \times \{-1, +1\} \tag{2.1}$$

Denklem 2.1'de $\boldsymbol{x_i}$ her bir gözleminin y_i sınıfına ait olduğu $(\boldsymbol{x_i}, y_i)$ sıralı ikilileri ile gösterilmiştir. Bu sınıflandırma türüne ikili sınıflandırma denilmektedir.

Sınıflandırma probleminde amacımız yeni bir x gözlemi X kümesine eklendiğinde $\{-1, +1\}$ kümesindeki uygun y etiketini belirleyebilmektir. Bu işlem için ise X ve $\{-1, +1\}$ kümeleri üzerinde benzerlik kriterlerine gereksinim duyulur.

$\{-1, +1\}$ kümesi üzerindeki benzerlik durumu için herhangi bir işleme gerek yoktur. Zaten burada iki tane eleman vardır. X gözlem kümesi üzerindeki benzerliği belirleyebilmek için ise Denklem 2.2'deki f fonksiyonu tanımlanır.

$$f: X \times X \to \mathbb{R}, \quad \forall(\boldsymbol{x_1}, \boldsymbol{x_2}) \in X; f(x_1, x_2) \in \mathbb{R} \tag{2.2}$$

Denklem 2.2'deki bu f fonksiyonuna çekirdek fonksiyonu denir.

İki vektörün birbirine uzaklığı bir benzerlik kriteri olarak alınabilir. Bunun için X kümesi bir vektör uzayının alt kümesi ve x_1 ve x_2 gözlemleri de buradan alınan iki vektör olsun. Bu durumda bu iki vektör arasındaki uzaklık Denklem 2.3'teki gibi fark vektörünün normu olarak alınabilir.

$$\|\boldsymbol{x_1} - \boldsymbol{x_2}\| = \sqrt{\langle(\boldsymbol{x_1} - \boldsymbol{x_2}), (\boldsymbol{x_1} - \boldsymbol{x_2})\rangle} \tag{2.3}$$

Denklem 2.3'ten yola çıkarak bu iki vektörün iç çarpımları da bir benzerlik kriteri olarak alınabilir. $\boldsymbol{x_1}$ve $\boldsymbol{x_2}$ vektörlerinin iç çarpımları Denklem 2.4'te gösterilmiştir.

$$\langle \boldsymbol{x_1}, \boldsymbol{x_2} \rangle = \sum_{i=1}^{n} x_{1_i} x_{2_i} \tag{2.4}$$

Burada x_{1_i}, $\boldsymbol{x_1}$ vektörünün i. elemanıdır.

Eğer ki gözlemlerin bulunduğu X kümesi bir iç çarpım uzayına ait değilse Denklem 2.4'teki iç çarpım kriterine ulaşabilmek için önce X kümesinden iç çarpım uzayının bir H alt kümesine Denklem 2.5'teki gibi bir φ fonksiyonu tanımlanır.

$$\varphi: X \to H, \quad \forall x: \varphi(x) \in H \tag{2.5}$$

Ardından φ fonksiyonu yardımıyla Denklem 2.6'daki iç çarpıma ulaşılabilir.

$$f(\boldsymbol{x_1}, \boldsymbol{x_2}) = \langle \boldsymbol{x_1}, \boldsymbol{x_2} \rangle = \langle \varphi(\boldsymbol{x_1}), \varphi(\boldsymbol{x_2}) \rangle \tag{2.6}$$

2.2. Bazı Doküman Sınıflandırma Metotları

2.2.1. Çok Katmanlı Algılayıcılar (ÇKA)

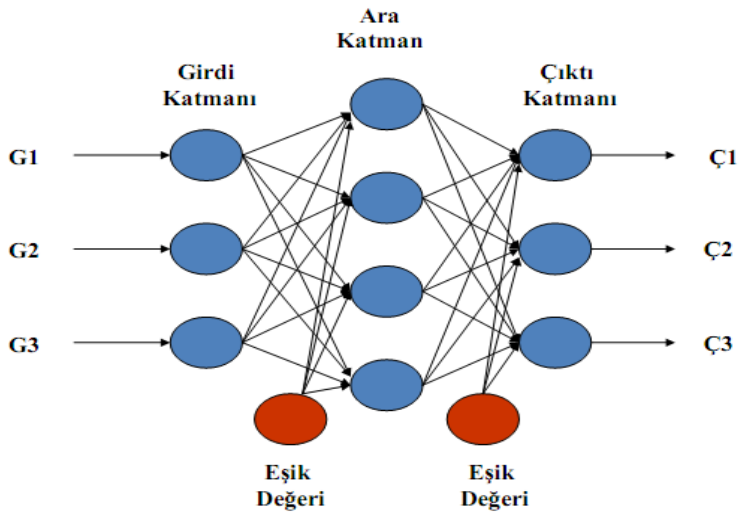
Çok sayıda ve büyük boyutlardaki verileri sınıflandırmada önemli başarılara imza atmış sınıflandırma modellemelerinden biri de Çok Katmanlı Algılayıcılardır (Multilayer Perceptron).

Çok Katmanlı Algılayıcılar, Tek Katmanlı Algılayıcılar'dan geliştirilmiş modellemelerdir. Tek Katmanlı Algılayıcıların yaptığı doğrusal sınıflandırmadan kaynaklı eksiklikler Çok Katmanlı Algılayıcılarda giderilmiş ve bu yöntem ile doğrusal olmayan veri grupları doğru sınıflandırılabilmiştir.

Çok Katmanlı Algılayıcıların çalışma prensibi ise şu şekildedir (Öztemel, 2012):

Öncelikle, tüm Yapay Sinir Ağlarında (YSA) olduğu gibi ağın eğitilmesi işlemi vardır. YSA, "makine öğrenmesi" denilen örneklerden öğrenme yaptığı için örnekler seçilir ve ağa gösterilir. Ağ, bu örneklerle ile kendini eğitme işlemini yapacaktır. Girdi katmanından alınan bu bilgiler Çok Katmanlı Algılayıcının öğrenme ve momentum katsayıları, aktivasyon fonksiyonları, ağırlıklar, eşik değerler ve proses elemanları ile işleme girerek eğitim işlemi gerçekleşir. Çok Katmanlı Algılayıcılar "Delta Öğrenme" kuralını kullanır. Bu kurala "öğretmenli öğrenme" de denmektedir. Delta Öğrenme kuralı beklenen çıktı ile gerçekleşen

çıktının farkını en aza indirgemek esasına dayanır. Yani hem girdiler hem çıktılar sisteme verilir. Sistem ise hatayı bulur ve kabul edilebilir düzeye indirmeye çalışır. Hatayı en aza indirgemek için ağ sürekli olarak ağırlıkları değiştirir. Beklenen çıktı ile gerçekleşen çıktı arasındaki fark kabul edilebilir düzeye geldiğinde ise ağ eğitilmiş olur. Eğitime başlarken ağırlıklara rastgele değerler verilir. Optimum değere ise ağ eğitim sırasında kendisi ulaşır. Eğitimin ardından ağın performansı test edilmelidir. Ağa bu amaçla daha önce görmediği çıktılarını bizim bildiğimiz örnekler, çıktıları olmadan gösterilir. Ağın yeni girdi bilgilerini kullanarak aynı işlemler sonucunda sınıflandırmasını yapar. Kabul edilebilir performans için belli bir standart yoktur. Bu eşik, kullanıcının inisiyatifindedir. Kullanıcı, hata toleransına göre ağın istediği gibi çalışıp çalışmadığına kendisi karar verir.



Şekil 2.1. Çok Katmanlı Algılayıcı Modeli (ÇKA)

2.2.1.1. Çok Katmanlı Algılayıcının Öğrenme Kuralı

ÇKA öğrenmesi 2 aşamada gerçekleşir. Bunlar:

- İleri Doğru Hesaplama
- Geriye Doğru Hesaplama

İleri Doğru Hesaplama, örneğin girdi katmanından gösterilmesi ile başlayıp çıktı katmanından bir çıktı elde edilmesine kadar gerçekleşen işlemleri kapsar. Elde edilen çıktı ile beklenen çıktı arasındaki fark hesaplanır. Bu farka hata denir. Eğer

hata kabul edilebilir seviyede değilse Geriye Doğru Hesaplama başlar. Amaç, hatayı yayarak minimize etmektir (Rummelhart vd., 1986). Bunun için hata geriye doğru ağırlıklara dağıtılır. Yani ağırlıklar değiştirilmiş olur. Yeni ağırlıklar elde edilmesi ile tekrar ileri doğru hesaplama başlar ve sonraki iterasyona geçilir. Her bir iterasyonda aynı işlemler yapılır ve istenen hata seviyesine ulaşıldığında iterasyon durdurulur. Böylece ağ eğitilmiş olur.

İleri Doğru Hesaplama: Her bir iterasyonda ağın çıktısının hesaplandığı safhadır. Bu safhada girdi katmanına gelen girdiler hiçbir değişikliğe uğramadan ara katmana iletilir. Yani girdi katmanındaki *k.* elemanının çıktısı Denklem 2.7'deki gibi belirlenir.

$$Ç_k^g = G_k \tag{2.7}$$

Her bir ara katman elemanına gelen net girdilerin bulunması gerekir. *j.* ara katman elemanına gelen tüm girdiler ağırlıkları ile çarpılarak toplanır.

$$Net_j^a = \sum_{k=1}^{n} A_{kj}^g Ç_k^g \tag{2.8}$$

Denklem 2.8 ile gelen net girdi hesaplanır. Burada A_{kj}^g *k.* girdi katmanı elemanını *j.* ara katman elemanına bağlayan bağlantının ağırlığıdır. *k.* girdi katmanı elemanının çıktısı olan $Ç_k^g$ ise *j.* ara katmanın girdisidir. Örneğin, 1. ara katman elemanına gelen net girdi Denklem 2.9'daki gibi bulunabilir.

$$Net_1^a = \sum_{k=1}^{n} A_{k1}^g Ç_k^g = A_{11}^g \cdot Ç_1^g + A_{21}^g \cdot Ç_2^g + \cdots + A_{n1}^g \cdot Ç_n^g \tag{2.9}$$

Her bir ara katman elemanına gelen net girdi aktivasyon fonksiyonundan geçirilmesi ile çıktılar elde edilir. Burada genelde aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanılmakla birlikte başka fonksiyonlar da kullanılabilir. Önemli olan kullanılan bu fonksiyonun türevlenebilir bir fonksiyon olmasıdır. Sigmoid fonksiyonu ile *j.* ara katman elemanının çıktısı Denklem 2.10'da görüldüğü gibi hesaplanır.

$$Ç_j^a = \frac{1}{1 + e^{-\left(Net_j^a + \beta_j^a\right)}} \tag{2.10}$$

Burada β_j^a, j. ara katmanına bağlanan eşik değer elemanının ağırlığıdır.

Ara katmanın çıktıları, çıktı katmanı elemanları için girdi olur. O halde yapılan işlemlerin aynısı çıktı katmanı için sadece girilen değerler değiştirilerek yapılırsa ağın çıktıları elde edilmiş olur. Çıktıların elde edilmesi ile ileri doğru hesaplama safhası da biter. Hata hesaplaması ve hatanın geriye yayılımı için bir sonraki safha olan geriye doğru hesaplamaya geçilir.

Geriye Doğru Hesaplama: Beklenen çıktı ile gerçekleşen çıktı arasındaki fark bize hatayı verir. Eğer bu fark tolore edilemeyecek derecede ise geriye yayılarak azaltılır. Her bir iterasyonda bu hata azaltılarak minimuma indirilmeye çalışılır. *m.* çıktı elemanının hatası Denklem 2.11'deki gibi olacaktır.

$$E_m = B_m - Ç_m \tag{2.11}$$

Bu sadece bir çıktı elemanı için gerçekleşen hatadır. O halde ağın toplam hatası (TH) Denklem 2.12 ile bulunur.

$$TH = \frac{1}{2}\sum_m {E_m}^2 \tag{2.12}$$

Hatayı en aza indirmek için bu gerideki ağırlıklara dağıtmak gerekir. Geriye yaymanın iki durumu vardır. İlk durum ara katman-çıktı katmanı arasındaki ağırlıkların değiştirilmesi, ikinci durum ise ara katmanların kendi ve ara katman-girdi katmanı arasındaki ağırlıkların değiştirilmesidir.

2.2.1.2. Ara Katman ile Çıktı Katmanı Arasındaki Ağırlıkların Değiştirilmesi

ΔA_{jm}^a, j. ara katman elemanı ile m. çıktı elemanını birbirine bağlayan bağlantının ağırlığındaki değişim miktarı Denklem 2.13'de gösterilmiştir.

$$\Delta A_{jm}^a(t) = \lambda \delta_m^{ç} Ç_j^a + \alpha \Delta A_{jm}^a(t-1) \tag{2.13}$$

Burada λ öğrenme, α ise momentum katsayısıdır. Bunlara öğrenme parametreleri denmektedir ve öğrenme hızının arttırılmasında kullanılır. Dolayısıyla başlangıçta uygun parametreyi seçmek ağın eğitilmesi açısından önemlidir. t ise hangi

iterasyonda olduğumuzu gösterir. Formülden de anlaşılacağı üzere *t*. iterasyondaki ağırlıkların değişimi bir önceki iterasyondaki değişimlerden doğrudan etkilenmektedir. $\delta_m^{ç}$ geriye doğru hesaplamada ise *m*. çıktı elemanının hatasını gösterir ve Denklem 2.14'teki gibi hesaplanır:

$$\delta_m^{ç} = f'(Net) \tag{2.14}$$

f'ile kullandığımız aktivasyon fonksiyonun türevi belirtilmektedir. Biz sigmoid fonksiyonunu kullandığımız için hata değeri Denklem 2.15'teki gibi olacaktır.

$$\delta_m^{ç} = Ç_m(1 - Ç_m)E_m \tag{2.15}$$

Denklem 2.15'i düzenlersek değişimlerden sonraki yeni ağırlıklar Denklem 2.16'da gösterildiği gibi olacaktır.

$$A_{jm}^{a}(t) = A_{jm}^{a}(t-1) + \Delta A_{jm}^{a}(t) \tag{2.16}$$

Benzer olarak ara katman ile çıktı katmanı arasındaki eşik değer ünitesinin de ağırlıkları değiştirilmelidir. Eşik değer elemanının ağırlıkları değiştirilirken ise Denklem 2.15 ve 2.16'da kullanılan ara katman elemanların çıktılarını eşik değer elemanının çıktısı ve ağırlıkları eşik değerlerin ağırlıkları ile değiştirmek yeterli olacaktır. Eşik değer elemanının çıktısı her zaman 1'e eşittir. *m*. çıktı elemanına giden ağırlığını da $\beta_m^{ç}$ ile gösterirsek ağılıklardaki değişim aşağıdaki Denklem 2.17 ile bulunabilir.

$$\Delta\beta_m^{ç}(t) = \lambda\delta_m^{ç} \cdot 1 + \alpha\Delta\beta_m^{ç}(t-1) \tag{2.17}$$

Ağırlıklar değiştikten sonraki eşik değer ünitesinin çıktı elemanlarına olan yeni ağırlıkları Denklem 2.18'deki gibi hesaplanır.

$$\beta_m^{ç} = \beta_m^{ç}(t-1) + \Delta\beta_m^{ç}(t) \tag{2.18}$$

2.2.1.3. Ara Katmanlar Arası veya Ara Katman Girdi Katmanı Arasındaki Ağırlıkların Değiştirilmesi

Hatanın yayılması sadece ara katman ile çıktı katmanı arasındaki ağırlıklara olmamalıdır. Çünkü, hata sadece oradan kaynaklanmamaktadır. Hata tüm katmanlar arasındaki ağırlıklara dağıtılmalıdır. İki ara katman yada ara katman

girdi katmanı arasındaki ağırlıkların değişimini ΔA^g ile gösterirsek bu değişim Denklem 2.19 ile ifade edilir.

$$\Delta A_{kj}^{g}(t) = \lambda \delta_j^a Ç_k^g + \alpha \Delta A_{kj}^{g} \tag{2.19}$$

j. ara katman elemanının hatası ise Denklem 2.20'deki gibi bulunur.

$$\delta_j^a = Ç_j^a \left(1 - Ç_j^a\right) \sum_m A_{jm}^a \, \delta_m^ç \tag{2.20}$$

Burada f' ile belirtilenin aktivasyon fonksiyonun türevi olduğunu ve aktivasyon fonksiyonu olarak da sigmoid fonksiyonu alındığını hatırlayalım. Hata ve değişim miktarı bulunduktan sonra artık yeni ağırlık değerleri Denklem 2.21'deki gibi olur.

$$A_{kj}^{g} = \Delta A_{kj}^{g}(t) + A_{kj}^{g}(t-1) \tag{2.21}$$

Denklem 2.21'de, ara katman ile çıktı katmanı arasındaki işlemlerin aynısı bu kez iki ara katman yada ara katman girdi katmanı arasındaki eşik değer elemanının ağırlıklarının değişimi için yapılırsa *t*. iterasyondaki yeni ağırlıklar Denklem 2.22' deki gibi hesaplanır.

$$\beta_j^a(t) = \beta_j^a(t-1) + \Delta \beta_j^a(t) \tag{2.22}$$

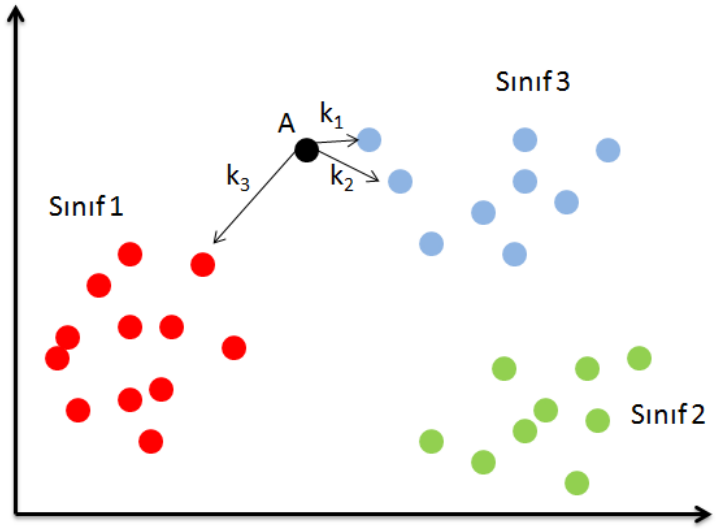
Böylece geriye doğru hesaplama da bitirilmiş olur ve sonraki iterasyona geçilir.

Bir sonraki iterasyon tekrardan ileri doğru hesaplama ile başlar. İleri doğru hesaplamada elde edilen çıktı ile beklenen çıktı arasındaki fark istenen seviyede ise sınıflandırma işlemi bitmiş olur; eğer istenen seviyede değilse tekrardan geriye doğru hesaplama yöntemi ile hata geriye doğru yayılır. Hata istenen seviyeye gelene kadar iterasyona devam edilir

2.2.2. K-En Yakın Komşu Algoritması (K-NN)

Geniş bir uygulama alanı bulan sınıflandırma metotlarından biri olan K En Yakın Komşu (K-NN) sınıfları belli olan bir örnek kümesindeki gözlem değerlerinden faydalanılarak sınıfları belli olmayan yeni öğelerin hangi sınıflara ait olduğunu belirlemede kullanılır.

Bu gözlemde sınıfları bilinen her bir örneğin sonradan belirlenen ve sınıfı belli olmayan bir gözleme uzaklıkları bulunur. Ardından bu örneklerden en yakın k tanesi ve yeni öğe ile bir küme oluşturulur. Bu kümedeki ağırlıklı sınıf yeni gözlemimizin sınıfı olarak atanır (Özkan, 2013).



Şekil 2.2. k=3 için A noktasının sınıfının belirlenmesi

Uzaklıkların hesaplanmasında çeşitli formüller vardır. Fakat en fazla Öklid uzaklık formülü kullanılır.

N boyutlu bir küme Öklid uzayında $\boldsymbol{P} = (p_1, p_2, p_3, \dots, p_n)$ ve $\boldsymbol{Q} = (q_1, q_2, q_3, \dots, q_n)$ noktaları arasındaki Öklid uzaklığı Denklem 2.23'teki eşitlik kullanılarak bulunur.

$$\sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.23}$$

Yöntemin uygulanışını kısaca özetlenecek olursa, önce k parametresi belirlenmelidir. Bu k parametresi sınıfı bulunmak istenen yeni gözleme en yakın olan örnek adedidir. Bu durumda tüm örneklerin uzaklıkları bulunur. Ardından seçilen en yakın k nokta ve yeni öğeden oluşan kümenin ağırlıklı sınıfı bulunur. Bu sınıf ise yeni gözlemimizin sınıfı olarak atanır.

Ağırlıklı sınıfın belirlenmesinde iki yöntem kullanılır. Birincisi en çok tekrarlanan sınıfın ağırlıklı sınıf olarak seçilmesidir. İkinci yöntem ise ağırlıklı oylama (weighted voting) yöntemidir.

$$d'(\boldsymbol{P},\boldsymbol{Q}) = \frac{1}{[d(\boldsymbol{P},\boldsymbol{Q})]^2} \tag{2.24}$$

Ağırlıklı oylama yönteminde Denklem 2.24. kullanılarak her bir örneğin yeni gözleme olan Öklid uzaklıklarının bir dönüşümü elde edilir. Ardından her bir sınıf için bu dönüşümlerin toplamları alınarak sınıfların kümedeki ağırlıklı oylama değerlerine ulaşılır. En büyük değere sahip olan sınıf ağırlıklı sınıf olarak atanır. Fakat ağırlıklı oylama yöntemi tüm örneklem için kullanıldığında daha doğru sonuç verebilir.

2.2.3. Naive-Bayes Sınıflandırma

Koşullu olasılıklar dikkate alınarak, özellikleri bilinen bir verinin önceden belirli sınıf etiketlerine ait olma olasılığını hesaplayan istatistiksel bir sınıflandırma modelidir. Bu durumda öncelikle koşullu olasılıktan biraz bahsetmek gerekirse (Özkan, 2013):

P(A|B): *A*'nın *B*'ye koşullu olasılığıdır. Yani *B* olayının olduğu bilindiği durumda *A* olayının meydana gelme olasılığıdır. *A*' nın *B*'ye koşullu olasılığı veren eşitlik Denklem 2.25'te ifade edilmiştir.

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{A'nın\ B\ içindeki\ olma\ olasılığı}{B'nin\ olma\ olasılığı} \tag{2.25}$$

Denklem 2.25'teki eşitlik $(A \cap B)$' nin olasılığını Denklem 2.26'daki gibi bulabiliriz:

$$P(A \cap B) = P(A|B)\,.\,P(B) \tag{2.26}$$

Denklem 2.25'te verilen koşullu olasılık tanımını kullanarak *B*'nin *A*'ya koşullu olasılığı Denklem 2.27'de verilmiştir.

$$P(B|A) = \frac{P(B \cap A)}{P(A)} \tag{2.27}$$

Denklem 2.27'de $P(A \cap B)$ yerine Denklem 2.25'teki eşitliği uygularsak Denklem 2.28'e ulaşırız.

$$P(B|A) = \frac{P(A|B).P(B)}{P(A)} \tag{2.28}$$

Bayes Teoremi: $B_1 \cap B_2 = \emptyset$ ve $B_1 \cup B_2 = B$ olacak şekilde B_1ve B_2 kümelerini ele alalım. A ise B içerisinde bir olay olsun. Bu durumda A'nın B_1 yada B_2 içerisinde olma durumunu bulmak için öncelikle A olayını B_1 ve B_2 cinsinden eşitini Denklem 2.29'daki gibi bulalım:

$$A = (B_1 \cap A) \cup (B_2 \cap A) \tag{2.29}$$

Bu durumda A olayının olasılığı denklem 2.30'da verilmiştir.

$$P(A) = P(B_1 \cap A) + P(B_2 \cap A) \tag{2.30}$$

Denklem 2.28'den ve 2.30'dan Denklem 2.31'e ulaşırız.

$$P(B_1|A) = \frac{P(A|B_1).P(B_1)}{P(A)} = \frac{P(A|B_1).P(B_1)}{P(B_1 \cap A) + P(B_2 \cap A)} \tag{2.31}$$

Formülü genelleyecek olursak $B_1 \cup B_2 \cup B_3 \cup \ldots \cup B_n = B$ ve $\forall i, j \leq n$ için $B_1 \cap B_2 = \emptyset$ olmak üzere A, B kümesi içinde bir olay ise A olayının B_j sınıfında olma olasılığını veren eşitlik denklem 2.32'deki gibidir.

$$P(B_j|A) = \frac{P(A|B_j).P(B_j)}{P(A)} \tag{2.32}$$

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i) \tag{2.33}$$

Denklem 2.33 A'nın B_i kümeleri kullanılarak olasılığını vermektedir. Denklem 2.33'ü Denklem 2.32'de yerine koyarsak Denklem 2.34'teki gibi Genelleştirilmiş Bayes kuralını elde ederiz.

$$P(B_j|A) = \frac{P(A|B_j).P(B_j)}{\sum_{i=1}^{n} P(A \cap B_i)} \tag{2.34}$$

Bayes Sınıflandırma: E, E_i $(i \leq n)$ ayrık sınıflarından oluşan (Kim vd., 2002) bir uzay $\boldsymbol{X}$ ise E içerisinde $x_1, x_2, x_3, \ldots, x_n$ niteliklerinden oluşan bir veri örneği

olsun. Bu durumda *X'* in hangi E_i sınıfına ait olduğunu bulmak için Denklem 2.35 kullanılır:

$$P(E_i|X) = \frac{P(\boldsymbol{X}|E_i).P(E_i)}{P(\boldsymbol{X})} \tag{2.35}$$

Ve $\forall i$ için paydadaki $P(X)$ sabit olacağından Denklem 2.36 değerini bulmak yeterlidir.

$$max_{i=1,2,3\ldots n}\{P(\boldsymbol{X}|E_i).P(E_i)\} \tag{2.36}$$

$$P(X|E_i) = \prod_{k=1}^{n} P(x_k|E_i) \tag{2.37}$$

$\boldsymbol{X}$ olayının E_i'ye koşullu olasılığı her bir x_i elemanının E_i'ye koşullu olasılıklarının çarpımı olarak yazılabileceği Denklem 2.37'de verilmiştir. O halde Denklem 2.37 Denklem 2.36'da yerine yazılırsa Denklem 2.38 elde edilir.

$$max_{i=1,2,3\ldots n}\left\{\prod_{k=1}^{n} P(x_k|E_i)\,.P(E_i)\right\} \tag{2.38}$$

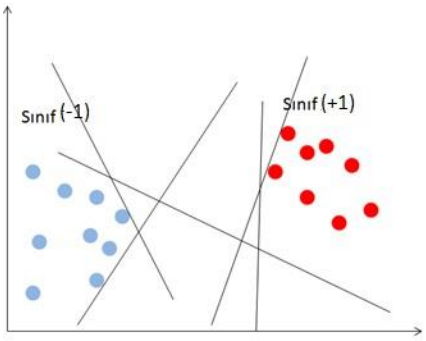
Buradan $\boldsymbol{X}$ en olasılığı en büyük olan E_i sınıfına dahil olur.

2.2.4. Destek Vektör Makineleri (DVM)

Destek Vektör Makineleri (Support Vector Machine) istatistiksel öğrenme teorisi ve yapısal riski en aza indirme ilkesine dayanan, denetimli öğrenme yöntemini kullanarak öğrenmesini gerçekleştiren bir sınıflandırma modelidir (Ateş, 2014). Denetimli öğrenme yönteminde sınıfları belli olan veriler iki sınıf etiketine ayrılır. Nitelikleri belli olan sınıflar göz önünde bulundurularak yeni eklenen verinin hangi sınıfa ait olduğu belirlenir. Destek Vektör Makineleri ile sınıflandırma iki başlık altında toplanabilir. Doğrusal ayrılabilen uzaylar için sınıflandırma ve doğrusal ayrılamayan uzaylar için sınıflandırma.

2.2.4.1. Doğrusal Ayrılabilen Uzaylar İçin Sınıflandırma

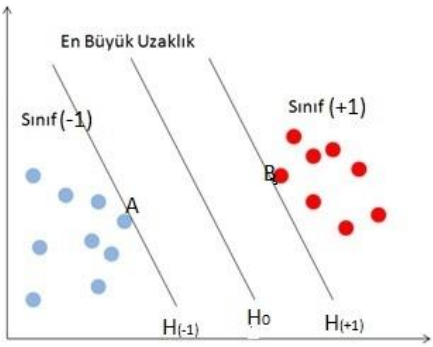
Doğrusal sınıflandırmada asıl amaç *n* boyutlu bir uzayda verilen iki sınıfı ayıran hiper düzlemler arasından optimal ayırma hiper düzleminin seçilmesidir.



Şekil 2.3. İki sınıf için doğrusal ayrılabilen verilerin hiper düzlemleri

Bu doğrultuda *n* hacimli bir eğitim kümesi olan S kümesi verilsin. $\forall x_i \in S$ k adet niteliğe sahip örnek, $\{-1,+1\}$ sınıf etiketleri olmak üzere $y_i \in \{-1,+1\}$ örneklerin hangi sınıfa ait olduklarını temsil eden çıktı, $\boldsymbol{w} = \{w_1, w_2, w_3, \dots, w_n\}$ hiper düzlemin normali ve aynı zamanda da ağırlık vektörü ve b sabit olmak üzere her bir $\boldsymbol{x_i}$ örneğini k boyutlu vektör olarak alacak olursak hiper düzlemlerin denklemlerini veren eşitlik Denklem 2.39'da verilmiştir.

$$H = < \boldsymbol{w}, \boldsymbol{x_i} > + b \tag{2.39}$$



Şekil 2.4. İki sınıfı ayırmak için kullanılan hiper düzlemler

Optimal ayırma düzlemini bulmak için her iki sınıf arasındaki hiper düzlemler arasından birbirine en uzak iki hiper düzlem $H_{(-1)}$ ve $H_{(+1)}$ hiperdüzlemleri seçilir. Bu her iki hiperdüzlemin tam ortasından geçen H_0 hiper düzlemi ise aranan optimal ayırma hiper düzlemidir (Ayhan ve Erdoğmuş, 2014).

En uzak iki hiper düzlemin seçilebilmesi için sınıfların sınırdaki örneklerinden faydalanılır. Üzerinden sınır hiper düzlemlerinin geçtiği bu örneklere destek

vektörleri adı verilir. Şekilde görüldüğü üzere A ve B noktaları sınır hiperdüzlemlerinin belirlenmesinde kullanıldığı için destek vektörleridir.

$$H_{(-1)}: < \boldsymbol{w}, \boldsymbol{x_i} > + b = -1 \tag{2.40}$$

$$H_{(+1)}: < \boldsymbol{w}, \boldsymbol{x_i} > + b = +1 \tag{2.41}$$

Denklem 2.40 ve 2.41 ile birbirine paralel ve en uzak olan sırasıyla $H_{(-1)}$ ve $H_{(+1)}$ hiperdüzlemleri bulunur. Bu düzlemlerin tam ortasından geçen ve optimal ayırma hiper düzlemi olan H_0' ı veren ifade Denklem 2.42'deki gibidir.

$$H_0 : < \boldsymbol{w}, \boldsymbol{x_i} > + b = 0 \tag{2.42}$$

H_0 hiper düzleminin üst ve altında kalan noktalar sırasıyla Denklem 2.43 ve 2.44'te verilmiştir.

$$< \boldsymbol{w}, \boldsymbol{x_i} > + b > 0, \quad y_i = +1 \tag{2.43}$$

$$< \boldsymbol{w}, \boldsymbol{x_i} > + b < 0, \quad y_i = -1 \tag{2.44}$$

Sonuç olarak, Denklem 2.45'teki optimizasyon problemi ifade edilir. Bu problemin çözülmesi ile destek vektörler arasındaki mesafe maksimize edilecek, sonucunda ise en az hata ile sınıflandırma yapılacaktır.

$$y_i(< \boldsymbol{w}, \boldsymbol{x_i} > + b) - 1 \geq 0 \tag{2.45}$$

Destek vektör makineleri sınır hiper düzlemleri arasındaki uzaklığı maksimuma çıkararak hatayı en aza indirmek ister. Bunun için $H_{(-1)}$ ve $H_{(+1)}$ hiper düzlemlerinin H_0 hiperdüzlemi üzerindeki bir noktaya olan uzaklık formülü Denklem 2.46'da verilmiştir.

$$d = \frac{< \boldsymbol{w}, \boldsymbol{x_i} > + b}{||\boldsymbol{w}||} \tag{2.46}$$

Burada $||\boldsymbol{w}||, \boldsymbol{w}$ aağırlık vektörünün normudur.

2.3. Bilgi Kazanımı

Bilgi kazanımı (BK), kararsızlığın tersidir. Bir kelime ya da nesnenin önemini; bir özelliğin konu, nesne veya bir sınıf ile ne kadar alakalı olduğunu ölçen bu sayede de daha çok alakalı özelliklerin kullanılmasını sağlayan kriterdir (Gündüz, 2013). Bilgi kazanımı işlemlerinin çıktısı 0 ve 1 arasında bir değerdir. Bu süreç sonunda bir niteliğin kararsızlık ve bilgi kazanımı değerleri birbirinin tam tersi değerler olarak elde edilir. Örneğin, bir niteliğin kararsızlık değeri 0 ise bilgi kazanım değeri 1 dir.

S_x eğitim örneklerinin (Roobaert, 2006) kümesi, $\boldsymbol{x_i}$ bu kümedeki i. değişkenin vektörü, $p_{\mp}(S)$ S eğitim kümesinin pozitif/negatif sınıfına ait oma olasılığı ve $\left|\frac{S_{\boldsymbol{x_i}=\boldsymbol{v}}}{S_x}\right|$ v değerine sahip i. değişkenin örneklerinin eğitim kümesine bölümünün büyüklüğü olmak üzere S kümesinin kararsızlığı (entropisi), Denklem 2.47'de verilmiştir:

$$H(S) = -p_s \, log_2 \, p_+(S) - p_-(S) \, log_2 \, p_-(S) \tag{2.47}$$

Bu durumda $\boldsymbol{x_i}$ vektörünün, S_x eğitim kümesindeki bilgi kazanımını veren eşitlik Denklem 2.48'de verilmiştir:

$$BK(S_x, \boldsymbol{x_i}) = H(S_x) - \sum_{v=values(x_i)}^{\left|\frac{S_{\boldsymbol{x_i}=\boldsymbol{v}}}{S_x}\right|} H\left(S_{\boldsymbol{x_i}=\boldsymbol{v}}\right) \tag{2.48}$$

Denklem 2.48'de verilen eşitlik ile hesaplanan x_i vektörünün bilgi kazanım değeri önceden belirlenen eşik değerinden aşağıda ise bu öznitelik sınıflandırma sürecinde ihmal edilir, üstünde bir değer ise dikkate alınır. Bu şekilde

sınıflandırmada etkisi olmayan nitelikler göz ardı edilmiş, dikkate alınacak olan vektör uzayının boyutu azaltılmış dolayısıyla da sınıflandırma daha hızlı ve güvenilir hale getirilmiş olur.

2.4. Görüntünün İskeletlenmesi

İskeletleme, ikili görüntüdeki önplanda bulunan gereksiz pikselleri arka plana iten ve bu sayede ikili görüntünün tek boyutlu bir ifadesini sağlayan bir piksel dönüştürme işlemidir (Atay, 2014). İskeletleme işlemi ardından kenar belirleme işlemi yapıldığı için çok önemli bir adımdır. İskeletleme işleminin bir sonucu olarak iki boyutlu bir şeklin iskeleti, konturdan eşit uzaklıktaki noktaların yeridir (Blum, 1967).

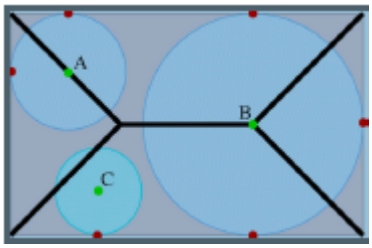
2.4.1. Ortalama İskeletin Tanımı

Ortalama iskelet ya da bir diğer ifadeyle ortalama eksen tanımı için en iyi bilinen analojilerden biri Maksimal-Çizilmiş Toplar'dır.

2.4.1.1. Maksimal-Çizilmiş Toplar (Maximally-Inscribed Balls)

Bu metot ile her bir sınır pikseli üzerinde 1 piksel boyutlu toplar çizilir ve bu toplar sınırın iç tarafına doğru bir diğer sınır pikseli topa teğet olana değin genişlemeye devam eder. Topa teğet olan en az iki sınır pikseli ile topun merkezi ortalama eksen noktaları olarak belirlenir. İşte bir nesnenin ortalama eksen dönüşümü (OED), maksimal çizilmiş topların merkezlerinin ve yarıçaplarının birleşimidir (Tagliasacchi, 2013).

Matematiksel ifadeyle, M maksimal çizilmiş topun merkezi ve R yarıçapı olmak üzere O nesnesinin ortalama eksene dönüşümü (OED); $[M, R] = OED(O)$ olur. Şekil 2.5'te maksimal çizilmiş toplar ile bir dikdörtgenin ortalama ekseni gösterilmiştir.



Şekil 2.5. A ve B noktaları iskelet; C noktası ise bir önplan noktasıdır (Palágyi, 2001).

2.4.2. İskeletin Özellikleri ve İskelet Çıkarım Metotları

İskeletleme işleminin bir çıktısı olarak gözlenen iskelet aşağıdaki özellikleri sağlamalıdır (Siddiqi ve Pizer, 2008):

1. İskelet, nesnenin topolojik özelliklerini korumalıdır.
2. İskelet, nesnenin önemli görsel özelliklerini tasvir edebilmelidir.
3. İskelet, küçük bozulmalardan etkilenmemeli, döndürme ya da dönüştürme işlemleri karşısında değişmemelidir.
4. İskelet, olabildiğince ince olmalıdır.
5. İskelet, nesnenin tekrar inşasında kullanılacak maksimal disk merkezlerini içermelidir.
6. İskelet, nesnenin sınırında merkezlenmelidir.
7. İskelet, orijinal nesnenin bağlantılılığını kaybetmemelidir.

İskeletleştirme algoritmaları tüm özelliklerin tamamını aynı anda sağlayamaz. Her biri farklı bir özelliği sağlayabilir. En uygun metot ise ancak uygulamada belirlenebilir. İskeletleştirme metotları 4 şekilde sınıflandırılabilir:

- Uzaklık dönüşümleri
- Voronoi diyagramı
- İnceltme teknikleri

- Matematiksel morfoloji

2.4.2.1. Uzaklık Dönüşümü

İkili görüntüdeki nesnenin her bir pikseli için arka plandaki piksellere en yakın uzaklıkları hesaplar ve öznitelik ve öznitelik olmayan pikselleri en yakın öznitelik pikseline dönüştürür.

Uzaklık dönüşümü için 5 tane metrik kullanılır (Borgefors, 1986). Bunlar Manhattan, Maksimum, Chamfer 3-4, Chamfer 5-7-11 ve Öklid metrikleridir.

İki boyutlu bir görüntüde (x_1, y_1) ve (x_2, y_2) pikselleri arasındaki farklı metriklere uzaklık ölçümleri Denklem 2.49, Denklem 2.50 ve Denklem 2.51'de verilmiştir. Denklem 2.49 Manhattan metriğini, denklem 2.50 Maksimum metriğini ve denklem 2.51 ise Öklid metriğini ifade etmektedir.

$$d_{Manhattan}\big((x_1, y_1), (x_2, y_2)\big) = |x_1 - x_2| + |y_1 - y_2| \tag{2.49}$$

$$d_{maksimum}\big((x_1, y_1), (x_2, y_2)\big) = max\{|x_1 - x_2|, |y_1 - y_2|\} \tag{2.50}$$

$$d_{öklid}\big((x_1, y_1), (x_2, y_2)\big) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.51}$$

Farklı metriklere ait kerneller ise Şekil 2.6'de verilmiştir.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 1 & \sqrt{2} \\ 1 & 0 & 1 \\ \sqrt{2} & 1 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 7 & 5 & 7 \\ 5 & 0 & 5 \\ 7 & 5 & 7 \end{bmatrix} \begin{bmatrix} 4 & 3 & 4 \\ 3 & 0 & 3 \\ 4 & 3 & 4 \end{bmatrix}$$

(a) (b) (c) (d) (e) (f)

Şekil 2.6. Farklı metriklere ait kerneller (a) Girdi (b) Manhattan metriği (c) Maksimum metriği (d) Öklid metriği (e) Chamfer 5-7-11 (f) Chamfer 3-4

Manhattan metriği, pikseller üzerindeki 4 bağlantılı komşuluğa dayanan yolu, Maksimum metrik 8 bağlantılı komşuluğa dayanan yolu, Öklid metriği ise 2 piksel arasındaki en kısa mesafeyi ölçer. Klasik Chamfer olarak da bilinen Chamfer 3-4 ve Chamfer 5-7-11 metrikleri pikselin 3×3 ve 5×5 komşuluklarında tamsayı değerleri ile Öklid metriğine yakınsar. Örneğin, Öklid

metriğindeki $\frac{\sqrt{2}}{1} = 1{,}41$ oranına Chamfer 3-4'te $\frac{4}{3} = 1{,}33$ ile yaklaşılırken, Chamfer 5-7-11'de ise $\frac{7}{5} = 1{,}4$ ile yaklaşılır.

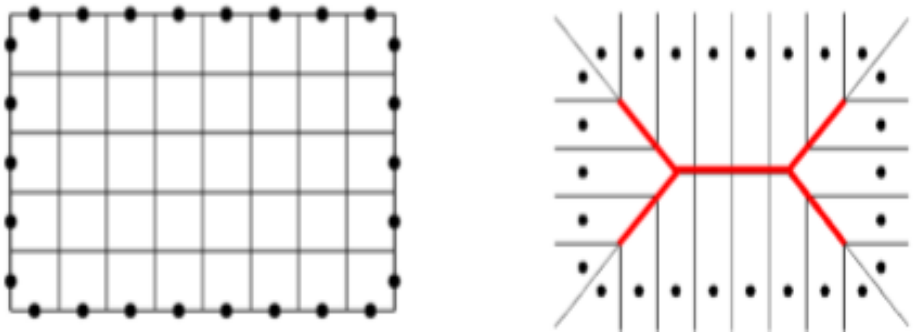
Şeklin iskeleti ise bu uzaklık dönüşümlerinden biri kullanılarak oluşturulabilir. Öncelikle şeklin yukarıdaki metrikler ile bir uzaklık haritası çıkarılır ve bir tepe noktasına yani bir lokal maksimum ya da eyer noktasına varana kadar uzaklık haritası ortalama değerlere sahiptir. Her bir tepe noktası şeklin ortalama eksenini inşa eder. Şekil 2.7'de uzaklık dönüşümünün tepe noktaları gösterilmiştir.

4	3	2	1	2	3	4
3	2	1	0	1	2	3
2	1	0	1	0	1	2
2	1	0	1	1	0	1
1	0	1	2	2	1	0
1	0	1	2	3	2	1
0	1	2	3	4	3	2

Şekil 2.7 Sıfır noktaları tepe noktalarını belirtir

2.4.2.2. Voronoi Diyagramı

Uzayı parçalamanın bir yolu olan Voronoi diyagramında, hücre olarak adlandırılan her bir parçanın şeklin sınırından üretilen örnek üretim noktaları vardır. Buradaki üretim noktaları sonsuza gittiğinde Voronoi diyagramı iskelete yakınsar. Şekil 2.8' da Voronoi diyagramı kullanılarak dikdörtgensel şeklin iskeletinin çıkarılması gösterilmiştir. Üretim noktaları olarak seçilen bazı sınır noktaları özelleştirilerek Voronoi diyagramının alt grafiğince yaklaşılır (Palágyi, 2001).



(a) (b)

Şekil 2.8. Voronoi Diyagramı ile dikdörtgensel şeklin iskeletinin çıkarılması (a) Dikdörtgensel şeklin üretim noktaları olarak seçilen bazı sınır noktaları (b) Üretim noktalarının sonsuza gitmesi ile oluşan şeklin iskeleti

2.4.2.3. İnceltme Algoritmaları

Geriye nesnenin iskeleti kalana kadar sınırından iterasyonel bir şekilde piksellerin uzaklaştırılması işlemidir. İnceltme işleminin sonunda görüntünün tam ortasında şeklin 1 piksel kalınlığında iskeleti kalır. İnceltme işlemi Şekil 2.9'da gösterilmiştir.

(a) (b) (c)

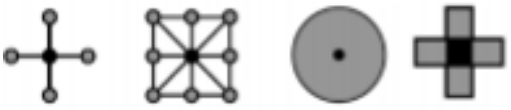
Şekil 2.9. T harfinin inceltme teknikleri ile iskeletinin çıkarılması (Atay, 2014)

Ardışık ve paralel yolla silme olmak üzere iki çeşit inceltme tekniği vardır (Lohmann, 1998). Ardışık silme tekniğinde her bir iterasyonda tek bir piksel silinirken paralel yolla silmede bir önceki iterasyonun bir sonucu olarak tüm pikseller tarandıktan sonra tek bir iterasyonda birçok piksel silinir. Ardışık silme tekniği şeklin yapısını korurken, paralel teknik yapıyı bozabilmektedir. İnceltme teknikleri gürültüden etkilendikleri için çoğunlukla iskeletin yerini tam olarak

belirlemede başarısız olabilir. Ayrıca piksel silme kurallarının şeklin türüne bağlı olarak değişebilmesi de inceltme teknikleri için bir diğer problemdir.

2.4.2.4. Matematiksel Morfoloji

Görüntü işlemede küme teorisine dayanan ve lineer olmayan bir metot olan matematiksel morfolojiye göre morfolojik süreç için kullanılmak üzere bir yapı elemanı tanımlanarak işleme başlanır. Bu yapı elemanı, kökenine ve ona bağlı elemanların pozisyonuna bağlı olarak seçilir ve şeklin geometrik özelliklerine uyum sağlamalıdır. Matematiksel morfoloji çıktısındaki her bir piksel değeri, girdi görüntüsündeki karşılık gelen piksel ile onun yapı elemanındaki komşularının kıyaslaması esasına dayanır. Bazı yapı elemanları Şekil 2.10'de verilmiştir.



Şekil 2.10. Yapı elemanları (Curic, 2013)

4 temel yapı operatörü mevcuttur: $\ominus$: aşındırma, $\oplus$: genişletme, o: açma, $\bullet$: kapatma (Curic, 2014)

Genişleme, görüntüdeki nesnenin sınırına piksel ekleme iken aşındırma, piksel uzaklaştırma; açma, aynı yapı elemanı kullanarak genişlemenin takip ettiği aşındırma iken kapatma, aşındırmanın takip ettiği genişleme işlemi olarak tanımlanır. A girdi görüntüsü, B yapı elemanı olmak üzere Denklem 2.52 ve Denklem 2.53' de sırasıyla açma ve kapama işlemlerinin matematiksel ifadeleri verilmiştir.

$$A \mathrm{o} B = (A \ominus B) \oplus B \tag{2.52}$$

$$A \mathrm{o} B = (A \oplus B) \ominus B \tag{2.53}$$

Açma işlemi çıkıntı ve bağlantıları yok ederken, kapatma işlemi çukur ve boşlukları doldurarak sınırda düzleşmeyi sağlar. Şekil 2.11 3×3 bir yapı elemanı kullanılarak ikili görüntüdeki bir açma işlemini göstermektedir.

(a) (b) (c)

Şekil 2.11. Bir yapı elemanı kullanılarak ikili görüntüdeki bir açma işlemi (a) Girdi görüntüsü (b) Aşındırma işlemi (c) Aşındırmayı takip eden genişleme işlemi

rB topolojik olarak açık disk, drB kapalı disk olmak üzere nesnenin iskeletini ifade denklem 2.54'teki gibidir:

$$S(O) = O \ominus rB - [(O \ominus rB) \oplus drB] \tag{2.54}$$

Tüm bu iskeletleme metotları iskeleti, sınırdaki gürültüden dolayı istenmeyen kolları ile beraber üretir. Tüm bu olumsuz etkilerin üstesinden gelebilmek için şeklin sınırı iskeletleme işlemi yapmadan önce düzleştirilebilir. Bu işlemi iskeletin budaması denir (Bai, 2007).

2.5. Kenar Tespiti

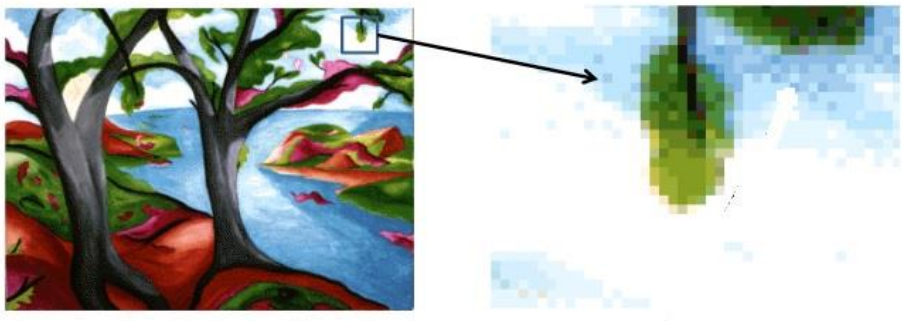
Resim sınıflandırmanın en önemli adımlarından birisi kenar tespitidir. Resimlerde parlaklıklardaki ani ve belirgin değişimler resim içindeki nesnelerin kenarlarını gösterdiğinden kenar tespit algoritmaları bu geçişleri bulmayı amaçlar (Bilgi, 2012).

Kenar tespitinin yapılabilmesi için renkli resimler önce gri tonlamalı ardından siyah beyaz resimlere dönüştürülür. Çünkü, renkli resimlerde kenar tespiti çok daha zordur. Fakat düşük kontrastlı resimlerde renk geçişleri yumuşak olduğundan bunların siyah beyaz görüntülerinde, içerisindeki nesnelerin sınırları net değildir. Bu tarz resimleri siyah beyaza dönüştürmeden kenar tespiti işlemini uygulamak daha sağlıklı olacaktır (Aybar, 2008).

2.5.1. Görüntülerin Sayısallaştırılması

Bir resmi olduğu haliyle bilgisayara vermek bilgisayar için hiç bir şey ifade etmez. Dolayısıyla öncelikle resmin sayısal görüntüsünü oluşturmak gerekir.

Bir görüntünün temel elemanına piksel denir (Gonzalez ve Woods, 2008). Resimdeki her bir noktanın bilgisayardaki karşılığı bir piksel olarak ifade edilir. Dolayısıyla bir resim deyince akla mxn boyutunda, piksellerden oluşan bir matris gelmelidir. Bu matrisin boyutu ne kadar fazla ise piksel sayısı o kadar fazla olacağından görüntünün kalitesi de o denli yüksek olacaktır.

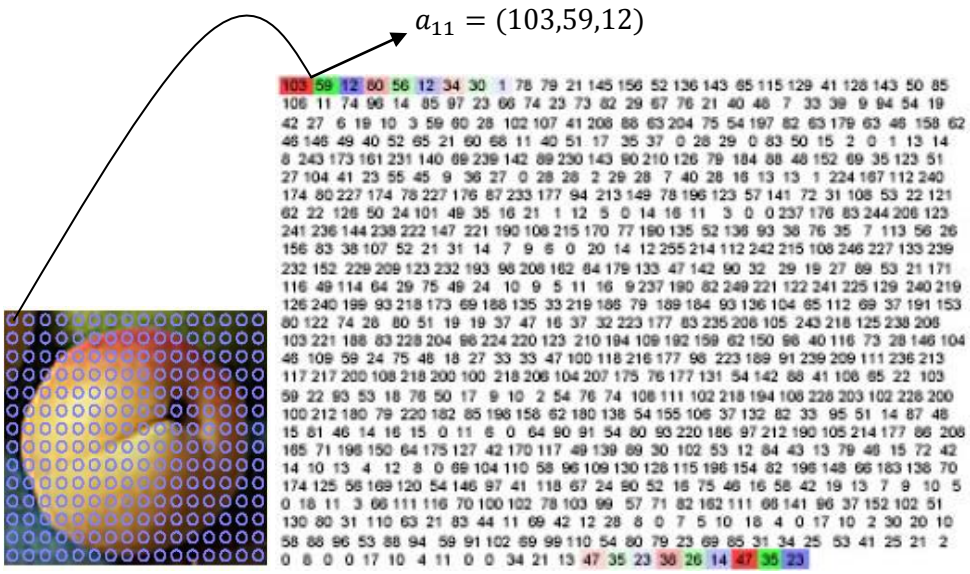


Şekil 2.12. Bir resim ve onun büyütülmüş bir parçası (Uğur, 2013)

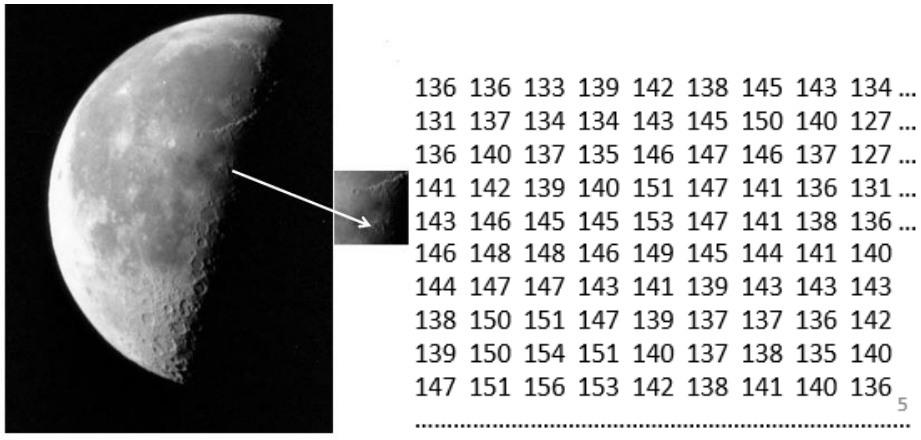
Şekil 2.12'de bir resim ve onun büyütülmüş parçası gösterilmektedir. Görüldüğü üzere büyütülmüş parçanın görüntü kalitesi resmin tamamına göre daha düşüktür. Bunun sebebi büyütülmüş kısmın pikseller matrisinin boyutunun resmin tamamına göre daha az olmasıdır.

Matrisin her bir elemanı resimde bir piksele karşılık gelir ve alacağı değer resmin türüne göre belirlenir.

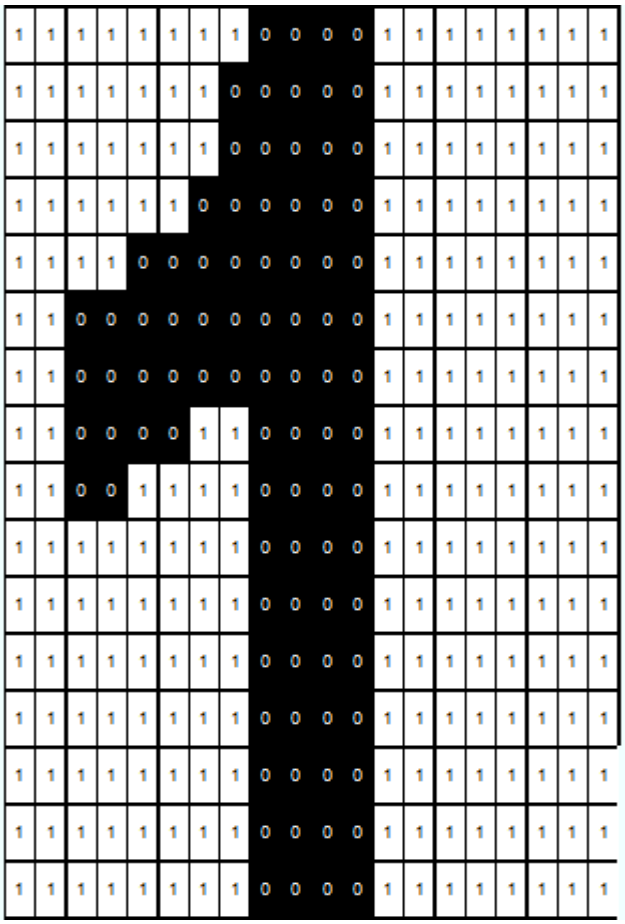
Renkli resimlerde her bir noktanın matrislerdeki karşılığı RGB (Red, Green, Blue) uzayında bir sıralı üçlüdür. Bu sıralı üçlünün ilk bileşeni noktadaki kırmızı oranını, ikinci bileşen yeşil, üçüncü bileşen ise noktadaki mavi oranını belirtir. Her bir bileşen 0 ile 255 arasında değer alır. Örneğin $(0,0,0)$ sıralı üçlüsü siyah rengi, $(255,255,255)$ sıralı üçlüsü de beyaz rengi belirtir. Bu durum gri tonlamalı ve siyah beyaz resimlerde ise çok daha basittir. Her bir noktaya matriste tek bir sayı karşılık gelir. Bu sayı gri tonlamalı resimlerde 0 ile 255 arasında bir değer iken, siyah beyaz resimlerde sadece 0 ve 1 değerleridir.



Şekil 2.13. Renkli resim ve matris karşılığı



Şekil 2.14. Gri tonlamalı resim ve bir kısmının matris karşılığı (Uğur, 2013)



Şekil 2.15. Siyah beyaz bir resim ve matris karşılığı (Bayram, 2015)

Resimlerin bilgisayarlara tanıtılmasının ardından kenar tespit aşamasına geçilebilir. Kenar tespit için farklı yöntemler vardır. Bu yöntemlerin hemen hepsi sadece filtrelemede kullandıkları kernel matrislerine ile ayrılır. Bu çalışmada ise kenar tespit yöntemleri olarak Prewitt, Sobel, Zero-Cross ve LoG filtreleri seçilmiştir.

2.5.2. Sobel Kenar Filtresi

En çok bilinen kenar belirleme algoritmalarından biridir. Resmin sayısal görüntüsündeki her bir elemanın 3×3 komşuluğundaki yatay ve dikey yönlerde elde edilen gradyan değerlerinin vektörel toplanması ile gradyan ölçümlerinin yönü üzerinde ortalama değerin bulunması esasına dayanır ve eğimin maksimum olduğu noktalarda kenar oluşturur (Sobel ve Feldman 1968).

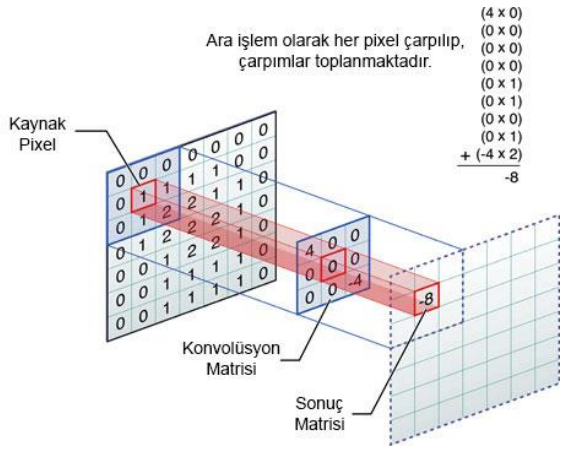
Türev yaklaşımlarını hesaplamak için 3×3 lük iki kernel matrisi kullanılır (Anonim, 2004). Bunlardan biri yataydaki diğeri ise düşeydeki değişimleri hesaplar. Yataydaki değişimler için $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$, düşeydeki değişimler için ise $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ matrisleri A kaynak görüntüsünün matrisi ile konvolüsyon işlemine girer. G_x yataydaki, G_y ise düşeydeki değişimleri içeren görüntüler olmak üzere Denklem 2.55 ve Denklem 2.56 sırasıyla G_x ve G_y değişimleri içeren matrisleri verir.

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A \tag{2.55}$$

$$G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A \tag{2.56}$$

2.5.2.1. Konvolüsyon İşlemi

Bir resmin üzerinde kare bir matrisin maske şeklinde sol üst köşeden başlayarak, maskenin merkezi her bir pikselin üzerinden geçecek şekilde bütün resmin taranması işlemidir. Bu tarama işlemi sırasında maske içerisinde kalan her bir piksel maskenin katsayıları ile çarpılıp bu çarpımların toplamı, yeni resme maskenin merkezinin geldiği konuma yazılır. Bu işlem Şekil 2.16'de görüldüğü gibidir (Kim, 2013).



Şekil 2.16. Konvolüsyon işlemi

Resmin her noktasında, gradyan büyüklüğünü bulabilmek için Denklem 2.57 kullanılır.

$$\|\mathrm{G}\| = \sqrt{{G_x}^2 + {G_y}^2} \tag{2.57}$$

Fakat Denklem 2.58 hızlı ve kolay bir şekilde yaklaşık sonucu verdiği için çoğu zaman tercih edilir.

$$|G| = |G_x| + |G_y| \tag{2.58}$$

Ayrıca G_x ve G_y kernelleri kullanılarak uzaya ait gradyanı veren yönünün açısı veren eşitlik Denklem 2.59'da verilmiştir.

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \tag{2.59}$$

Bu bağlamda θ açısının 0 olması maksimum kontrastı yani soldan sağa, siyahtan beyaza keskin geçişi ifade eder. Diğer açılar ise buradan saatin tersi yönde hesaplanabilir.

Yöntemin Uygulanışı şu şöyledir: Sobel operatörüne ait düşey ve yatay filtreler, girdi görüntüsünde düşey ve yatay yönde hareket ettirilerek sonuç görüntüsü elde edilir. Görüntüde sol üst köşeden başlayan operatör filtresinin hareketi boyunca her piksel değerine karşılık gelen filtre katsayısıyla piksel değerleri çarpıldıktan sonra tüm sonuçlar toplanır ve operatörün tepkisi hesaplanır. Girdi görüntüsünde düşey ve yatay yönlü filtrelerin aynı yerlerdeki tepkilerinin kareleri toplamlarının karekökü alınarak gerçek gradyan değerleri bulunur. Bulunan bu gradyan değerleri en son olarak girdi görüntüsü ile aynı boyuta sahip bir matrise karşılık gelen yerlere yerleştirildiğinde orijinal görüntünün kenarlarını gösteren bir görüntü elde edilmiş olur (Aybar, 2008).

3×3 bir matris yapılan filtreleme 5×5 filtreler ile de yapılabilir. Hatta 5×5 filtre ile yapılan işlem daha sonucunda fazla bilgi edilebildiği için daha iyi bir sonuç verir. Kenarlar daha belirgin, gürültü olarak adlandırılan resimdeki belirsiz küçük çizgiler ise daha az olur. Aşağıda sırasıyla düşey ve yatay değişimleri hesaplamada kullanılan 5×5 kernel matrisleri verilmiştir.

$$\begin{bmatrix} -5 & -4 & 0 & 4 & 5 \\ -8 & -10 & 0 & 10 & 8 \\ -10 & -20 & 0 & 20 & 10 \\ -8 & -10 & 0 & 10 & 8 \\ -5 & -4 & 0 & 4 & 5 \end{bmatrix} \quad \text{ve} \quad \begin{bmatrix} 5 & 8 & 10 & 8 & 5 \\ 4 & 10 & 20 & 10 & 4 \\ 0 & 0 & 0 & 0 & 0 \\ -4 & -10 & -20 & -10 & -4 \\ -5 & -8 & -10 & -8 & -5 \end{bmatrix}$$

2.5.3. Prewitt Kenar Filtresi

Sobel filtresi benzer olarak görüntü yoğunluk fonksiyonunun gradyanının bir yaklaşımını hesaplayan fakat Sobel filtresinden daha kolay ve hızlı sonuç veren kenar belirleme algoritmasıdır. Prewitt operatörü de Sobel operatörü gibi ayrık differasyon operatörüdür (Anonim, 2015). Görüntünün her noktasında Prewitt işleminin sonucu ya uygun gradyan vektörü yada bu vektörün normudur. Prewitt operatörü görünümün sayısal karşılığı olan matrisin dikey ve yatay yönlerde tam değerli olarak konvolüsyon işlemine dayanır.

Prewitt beyazdan siyaha mümkün olan en geniş artışın yönünü ve bu yöndeki değişim oranını veren yoğunluğun gradyanını hesaplayarak, görüntü değişimlerinin ne kadar keskin ne kadar yumuşak olduğunu ve böylece oranın bir sınır olup olamayacağını belirler.

Prewitt, Sobel operatörü ile tamamen aynı mantıkta çalışır. Sadece kullandığı kernel matrisi yönünden Sobel operatöründen ayrılır. Sobel operatörüne benzer olarak, kaynak matrisi ile konvolüsyon işlemine girecek iki tane 3×3matris kullanır. Bunlardan biri $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$ yataydaki değişimleri, diğeri $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ ise düşeydeki değişimleri hesaplamada kullanılır(Prewitt, 1970). A, kaynak matrisi, G_x ve G_y sırasıyla yataydaki ve düşeydeki değişimleri içeren görüntüler olmak üzere Denklem 2.60 ve Denklem 2.61 sırasıyla G_x ve G_y değişimleri içeren matrisleri verir.

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A \tag{2.60}$$

$$G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A \tag{2.61}$$

eşitlikleri ile G_x ve G_y bulunabilir. Burada " $*$ " işlemi ile konvolüsyon işlemi belirtilmektedir. Bu ise Sobel operatörü ile aynı konvolüsyon işlemi olduğu için ayrıca burada bir açıklama yapılmasına gerek yoktur.

Görüntüdeki her noktada elde edilen gradyan yaklaşımları gradyan büyüklüğünü vermek için Denklem 2.62 ile birleştirilebilir.

$$\|G\| = \sqrt{{G_x}^2 + {G_y}^2} \tag{2.62}$$

Ayrıca, G_x ve G_y kernelleri kullanılarak gradyanın yönü de Denklem 2.63 ile bulunabilir.

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \tag{2.63}$$

Örneğin, burada $\theta = 0$ olması durumu sağ tarafı siyah olan dikey sınırı ifade eder.

2.5.4. LoG Kenar Filtresi

Görüntünün ikinci yerel türevlerinin iki boyutlu bir ölçümü olan LoG operatörü kenar belirlemede sıklıkla kullanılan bir yöntemdir (Anonim, 2006). Gürültüye karşı hassaslığını indirgemek için başta Gauss tipli filtre ile yumuşatma sıklıkla kullanılır ve buradaki değişken tanımlanabilir. Girdi olarak tek bir gri tonlamalı resim alan operatör çıktı olarak birden fazla gri tonlamalı resim verir. LoG operatörü iki işi aynı anda yapar. Gauss tipli filtre ile gürültü uzaklaştırarak görüntüyü düzleştirirken, Laplace fonksiyonu ile de parlaklık değişimlerini hesaplayarak kenarları tespit eder (Özkul, 1995).

$I(x, y)$de piksel değerleri olmak üzere Denklem 2.64 ile görüntünün Laplas fonksiyonu olan $L(x, y)$ bulunabilir:

$$L(x, y) = \frac{d^2 I}{dx^2} + \frac{d^2 I}{dy^2} \tag{2.64.}$$

Girdi görüntüsü piksellerin bir birleşimi olarak ifade edildiği için ikinci türevlere ayrık iki konvolüsyon kerneli tanımlanmalıdır. Bunlar $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ ve $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ kernelleridir. Ve bunlarla standart konvolüsyon işlemi yapılır. Bu yüzden Laplace tipli filtrelemeden önce Gauss tipli filtreleme ile yumuşatma sıklıkla uygulanır ve böylece yüksek frekanslı gürültü bileşenleri indirgenir.

Yumuşatmada kullanılan Gauss tipli filtreleme ile Laplace tipli filtre konvol edilebilir ve ortaya çıkan karışmış filtre ile de görüntü konvolüsyon işlemine tabi tutulabilir. Bu durumun sağlayacağı avantaj ise sadece bir konvolüsyon kerneli LoG kerneli hesaplanabilir ve filtrelerin boyutları genelde görüntüden küçük olacağı için daha az aritmetik işlem ile sonuca gidilebilir.

Gauss tipli standart türevli ve sıfır merkezli iki boyutlu LoG fonksiyonunun formu Denklem 2.65'te verilmiştir:

$$LoG(x,y) = \frac{-1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right].exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{2.65}$$

Şekil 2.17'de ise $\sigma = 1.4$ olan LoG fonksiyonuna ayrık Gauss tipli yaklaşımları verilmiştir:

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

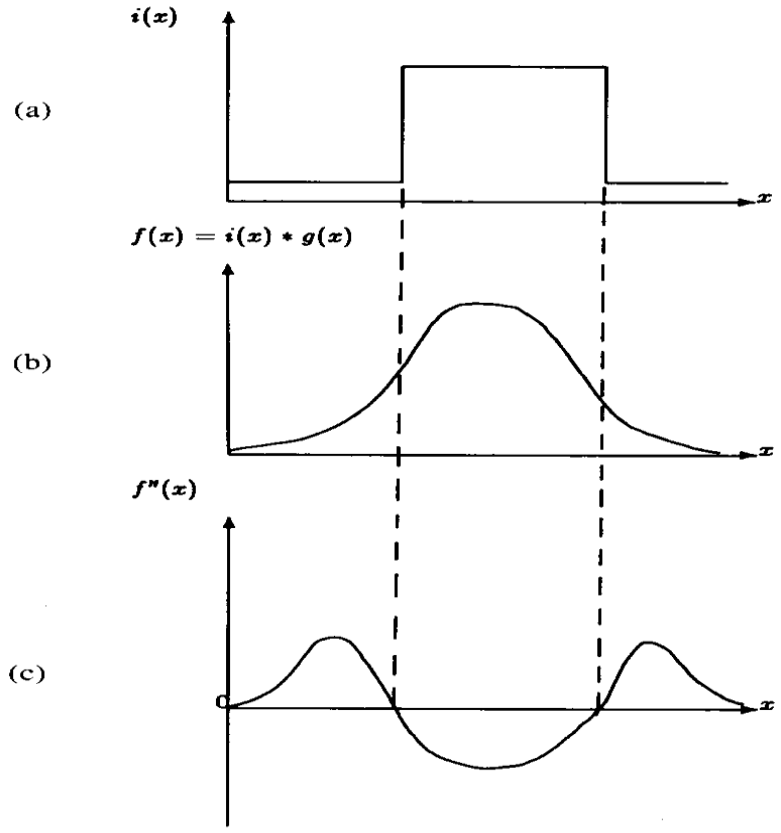
Şekil 2.17. $\sigma = 1.4$ olan LoG fonksiyonuna ayrık Gausyan yaklaşımları

Gausyan artan bir şekilde sınırlı yapıldığında LoG kernelleri, Laplace tipli çekirdek matrisine dönüşür. Bunun nedeni ise Gauss tipi filtreleme çok sınırlı olduğunda ($\sigma < 0.5$ piksel) yumuşatmanın hiçbir etkisinin olmamasıdır.

Sonuç olarak LoG operatörü görüntünün ikinci yerel türevlerini hesapladığından görüntünün sabit bir yoğunluğunun olduğu bölgelerde (Ör: yoğunluk gradyanının sıfır olduğu yerlerde) LoG tepkisi de sıfır olacaktır. Fakat farklı yoğunluklara sahip iki tarafın olduğu keskin sınırda karanlık tarafta pozitif aydınlık tarafta negatif olacaktır.

2.5.5. Zero-Cross Kenar Filtresi

Görüntünün ikinci türevindeki sıfır noktalarının bulunmasına dayanan Zero-Cross operatörü gürültüden etkilendiği için sıklıkla Gauss yumuşatma filtresini kullanarak işleme başlar. Filtrelenmiş sinyallerin ikinci türevlerinin sıfır-geçiş noktalarıyla sınırları bağdaştırarak kenar tespitini yapar. Şekil 2.18'da bu süreç resmedilmiştir:



Şekil 2.18. Filtrelenmiş bir sinyalin Zero-Cross filtrelemesi (a) Bir boyutlu basamak sinyali (b) Gauss yumuşatmasından sonra (c)Yumuşatılmış sinyalin ikinci türevleri ve sıfır geçiş noktaları (Clark, 1989)

Zero-Cross işleminin arkasındaki düşünce yumuşatılmış sinyalin fonksiyonunun birinci türevinin ekstrem noktalarının yerlerini ikinci türevinin sıfır noktalarının belirlemesidir. Fonksiyonun birinci türevinin büyüklüğünün bir lokal maksimum olduğu yerlerdeki noktalar sınır olarak düşünüldüğü için ikinci türevin sıfır geçiş noktaları buradaki sınırı belirleyebilir. Çünkü bu noktalar bize görüntüdeki yoğunluk fonksiyonundaki değişimlerini işaret eder. Fakat ikinci türevin sıfırları yumuşatılmış fonksiyonun birinci türevindeki lokal minimum noktalarını da gösterebilir. Lokal minimum noktaları bize sınır noktalarını veremez. Bu durumda ikinci türevin sıfır noktalarının yumuşatılmış fonksiyonun hangi ekstrem noktalarını verdiğini bilmemiz gerekir. Bu için ise Denklem 2.66 ve Denklem 2.67' deki durumlar kullanılabilir:

$$\frac{df}{dx}\frac{df^3}{dx^3} < 0 \tag{2.66}$$

$$\frac{df}{dx}\frac{df^3}{dx^3} > 0 \tag{2.67}$$

İkinci türevin sıfır noktalarında Denklem 2.66'daki durum geçerli ise burası yoğunluk fonksiyonunun lokal maksimumu, eğer Denklem 2.67'deki durum geçerli ise burası yoğunluk fonksiyonunun lokal minimumudur. Her iki durumda da nokta yoğunluk fonksiyonunun yerel ekstremum noktası olduğu için bu noktalar yoğunluk fonksiyonlarının sınırıdır; fakat yerel maksimum noktası doğal sınır, yerel minimum noktası hayali sınır olarak adlandırılır. Matematiksel ifadeyle:

Tanım. $f(x)$ yumuşatılmış yoğunluk fonksiyonunun bir sınırı olmak üzere $\left|\frac{df}{dx}\right|$ maksimum ise $f(x)$ doğal sınır, $\left|\frac{df}{dx}\right|$ minimum ise $f(x)$ hayali sınır olarak tanımlanır.

$\frac{df}{dx}\frac{df^3}{dx^3} = 0$ olduğu durumlarda ise sınır yoktur.

3. BULGULAR VE TARTIŞMA

3.1. Tekstil Veri Seti

Sistemin uygulanmasında ilk olarak gerçekleştirilen adım, veri setinin oluşturulmasıdır. Tekstil veri seti, "Çiçekli", "Puantiyeli", "Ekoseli", "Yatay çizgili", "Dikey çizgili", "45 derece çizgili" ve "135 derece çizgili" sınıf olmak üzere yedi sınıftan oluşmaktadır. Her sınıf, 60 tekstil resimlerinden oluşmaktadır. Toplamda veri setimiz, 20 MB'lık 420 resimden oluşmaktadır. Her sınıfa ait örnek tekstil resimleri Ek1'de verilmiştir. Bu resimler, renkli veya gri tonlamalı, jpeg formatında dosyalardır.

3.2. Sistemin Değerlendirilmesi

Hassasiyet (Recall), Kesinlik (Precision), Doğruluk (Accuracy) ve F-Ölçüsü (F-Measure) ve K-Katlamalı Çapraz Doğrulama Kriteri sistemin değerlendirilmesinde iyi bilinen ölçütlerdir ve bu tezde kullanılmıştır. Bu terimlerin kısaca ne anlama geldiklerini açıklayabiliriz (Aslıyan, 2002):

- Kesinlik (Precision): Doğru sınıflandırılmış resimlerin sayısının aynı sınıftaki tüm resimlerin sayısına oranı,
- Hassasiyet (Recall): Doğru sınıflandırılmış resimlerin sayısının sistemin o sınıfa atadığı tüm resimlerin sayısına oranı,
- F-Ölçüsü (F-Measure): Kesinlik ile hassasiyeti tek bir işlemde birleştirerek performansın değerlendirilmesini sağlayan ölçüdür.
- Doğruluk (Accuracy): Bütün sınıflardaki resimlerin doğru sınıflandırılan resimlerin sayısının, tüm resimlere oranı olarak ifade edilebilir.

Sistemin sonuçlarını Çizelge 1'deki gibi bir Hata Matrisi ile gösterebiliriz.

Çizelge 3.1. Hata matrisi

	"Pozitif" Sınıfı	**"Negatif" Sınıfı**
Test Sonucu "Pozitif"	DP	YP
Test Sonucu "Negatif"	YN	DN

Çizelge 3.1'deki hata matrisinde iki sınıf, "Pozitif" ve "Negatif" sınıf olarak belirlenmiştir. DP olarak ifade edilen, sistemin pozitif olarak sınıflandırdığı pozitif sınıfındaki resimlerin sayısı; YP, sistemin pozitif olarak sınıflandırdığı negatif resimlerin sayısı; YN, sistemin negatif olarak sınıflandırdığı pozitif ve DN de sistemin negatif olarak sınıflandırdığı negatif resimlerin sayısıdır. Anlaşılacağı üzere sistemin yaptığı doğru sınıflandırmalar D harfi ile, yanlış sınıflandırmalar Y harfi ile temsil edilmiştir. Bu durumda Denklem 3.1, Kesinliği, Denklem 3.2 Hassasiyeti vermek üzere sistemin değerlendirilmesini sağlayan Doğruluk ve F-Ölçüsü değerlerine sırasıyla Denklem 3.3 ve Denklem 3.4'teki eşitliklerden ulaşılabilir:

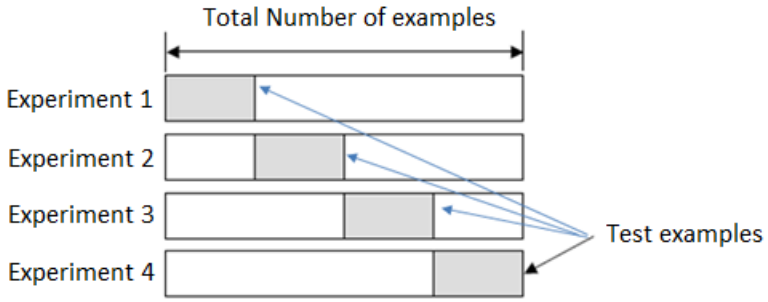
$$Kesinlik = \frac{DP}{DP + YP} \tag{3.1}$$

$$Hassasiyet = \frac{DP}{DP + YN} \tag{3.2}$$

$$Doğruluk = \frac{DP + DN}{DP + DN + YP + YN} \tag{3.3}$$

$$F - \text{Ölçüsü} = \frac{2 \times kesinlik \times hassasiyet}{kesinlik + hassasiyet} \tag{3.4}$$

Sistemin değerlendirilmesinde Şekil 3.1'de görüldüğü gibi K-Katlamalı Çapraz Doğrulama Kriteri kullanılmıştır. Bu yöntem, toplam veri setini *k* adet katmana böler ve her bir iterasyonda eğitim ve test örneklerini değiştirerek sistemin başarısı hakkında bilgi edinmemize yardımcı olur. Her iterasyonun kendi sonucu vardır. Tüm deneylerin ardından elde edilen sonuçların ortalaması yöntemin başarısını verir. K-Katlamalı Çapraz Doğrulama Kriterinde her resim hem eğitim hem test aşamasında kullanılır (Gutierez-Osuna, 2002). Böylece bilinmeyen veriden kaynaklanan hatalar minimize edilebilir (Uludağ, 2005).



Şekil 3.1. K-Katlamalı Çapraz Doğrulama modeli (Gutierez-Osuna, 2002)

3.3. Sistemin Eğitilmesi ve Test Edilmesi

Bu tezde yapılan uygulamalar Intel Core i7-2600 3.4 GHz işlemci, 16 GB RAM ve Windows 7 64 bit işletim sistemi üzerinde gerçekleştirilmiştir. MATLAB R2012a programı ile kodlar yazılarak resimler üzerinde önişleme ve frekans hesaplama yapılarak arff formatında veri seti dosyaları oluşturulmuştur. WEKA 3.7 (Hall vd., 2009) programıyla arff formatındaki veri setleri alınarak sınıflandırma metotlarıyla eğitim ve test işlemleri yapılmıştır. Bunun için 10-Katlamalı Çapraz Doğrulama metodu uygulanmıştır. WEKA programında K-NN yöntemi için K=1, 3 ve 5; Naive Bayes metodu için OldFormat=True ve Discretization=True; ÇKA için Ara katmanlardaki Nöron Sayısı=17, Öğrenme Katsayısı=0,1 ve Momentum Katsayısı=0,8; DVM için ise BuildLogisticModel=True, FilterType=Standardize Training Data olarak seçilmiştir.

Çizelge 3.2.'de kenar tespit filtrelerine göre 2x2, 3x3 ve 4x4 çekirdek matrislerinin toplam öznitelik sayıları görülmektedir. Çekirdek matrislerinin boyutu arttığında toplam öznitelik sayıları çok büyük sayılara ulaşmaktadır. Bu yüzden öznitelik boyut indirgeme metotlarından bilgi kazancı (Information Gain) kullanılmıştır. Aynı zamanda, farklı öznitelik sayılarındaki ortalama Doğruluk ve F-Ölçüsü sonuçları karşılaştırılmıştır.

Çizelge 3.2. Filtreler ve iskeletlemeye göre toplam öznitelik sayıları

Filtreler	İskeletleme	Toplam Öznitelik Sayısı		
		2x2 Çekirdek Matrisi	3x3 Çekirdek Matrisi	4x4 Çekirdek Matrisi
Sobel	**Evet**	15	398	30079
	Hayır	15	418	31268
Prewitt	**Evet**	15	398	30979
	Hayır	15	418	39853
LoG	**Evet**	15	430	20764
	Hayır	15	482	31268
Zero-Cross	**Evet**	15	430	20764
	Hayır	15	482	31268

Çizelge 3.3. Sobel operatörü ve K-NN metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**1**	**Evet**	0,874	0,873
		Hayır	0,886	0,885
	3	**Evet**	**0,898**	**0,896**
		Hayır	0,883	0,881
	5	**Evet**	0,888	0,884
		Hayır	0,879	0,876
10	**1**	**Evet**	0,893	0,892
		Hayır	0,888	0,886
	3	**Evet**	0,888	0,887
		Hayır	0,890	0,889
	5	**Evet**	0,879	0,876
		Hayır	0,879	0,876
5	**1**	**Evet**	0,850	0,848
		Hayır	0,855	0,855
	3	**Evet**	0,838	0,833
		Hayır	0,857	0,854
	5	**Evet**	0,845	0,840
		Hayır	0,848	0,844

Çizelge 3.4. Prewitt operatörü ve K-NN metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Ortalama Doğruluk Oranı	F-Ölçüsü
15	1	Evet	0,888	0,888
		Hayır	0,879	0,878
	3	Evet	0,890	0,890
		Hayır	0,876	0,872
	5	Evet	**0,893**	**0,893**
		Hayır	0,883	0,880
10	1	Evet	0,879	0,879
		Hayır	0,886	0,883
	3	Evet	0,883	0,883
		Hayır	0,881	0,878
	5	Evet	0,874	0,874
		Hayır	0,871	0,867
5	1	Evet	0,850	0,850
		Hayır	0,836	0,831
	3	Evet	0,850	0,850
		Hayır	0,845	0,841
	5	Evet	0,840	0,840
		Hayır	0,840	0,837

Çizelge 3.5. LoG operatörü ve K-NN metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	1	Evet	**0,926**	**0,926**
		Hayır	0,914	0,915
	3	Evet	**0,926**	**0,926**
		Hayır	0,910	0,910
	5	Evet	0,912	0,912
		Hayır	0,886	0,886
10	1	Evet	0,898	0,898
		Hayır	0,895	0,895
	3	Evet	0,898	0,898
		Hayır	0,900	0,900
	5	Evet	0,898	0,898
		Hayır	0,905	0,905
5	1	Evet	0,876	0,876
		Hayır	0,876	0,876
	3	Evet	0,886	0,886
		Hayır	0,871	0,871
	5	Evet	0,869	0,869
		Hayır	0,864	0,864

Çizelge 3.6. Zero-Cross operatörü ve K-NN metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**1**	**Evet**	**0,926**	**0,926**
		Hayır	0,914	0,914
	3	**Evet**	**0,926**	**0,926**
		Hayır	0,910	0,910
	5	**Evet**	0,912	0,912
		Hayır	0,886	0,886
10	**1**	**Evet**	0,898	0,898
		Hayır	0,895	0,895
	3	**Evet**	0,898	0,898
		Hayır	0,900	0,900
	5	**Evet**	0,898	0,898
		Hayır	0,905	0,905
5	**1**	**Evet**	0,876	0,876
		Hayır	0,876	0,876
	3	**Evet**	0,886	0,886
		Hayır	0,871	0,871
	5	**Evet**	0,869	0,869
		Hayır	0,864	0,864

Çizelge 3.7. Sobel operatörü ve K-NN metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**1**	**Evet**	0,891	0,879
		Hayır	0,881	0,879
	3	**Evet**	0,867	0,860
		Hayır	0,881	0,877
	5	**Evet**	0,848	0,840
		Hayır	0,867	0,861
100	**1**	**Evet**	0,893	0,890
		Hayır	**0,907**	**0,906**
	3	**Evet**	0,871	0,865
		Hayır	0,890	0,887
	5	**Evet**	0,876	0,870
		Hayır	0,874	0,867
50	**1**	**Evet**	0,898	0,894
		Hayır	0,902	0,900
	3	**Evet**	0,883	0,878
		Hayır	0,902	0,899
	5	**Evet**	0,864	0,859
		Hayır	0,881	0,877
15	**1**	**Evet**	0,874	0,871
		Hayır	0,864	0,862
	3	**Evet**	0,864	0,860
		Hayır	0,869	0,866
	5	**Evet**	0,871	0,869
		Hayır	0,874	0,871
10	**1**	**Evet**	0,883	0,882
		Hayır	0,869	0,868
	3	**Evet**	0,883	0,879
		Hayır	0,869	0,865
	5	**Evet**	0,888	0,884
		Hayır	0,871	0,866
5	**1**	**Evet**	0,717	0,708
		Hayır	0,690	0,682
	3	**Evet**	0,710	0,698
		Hayır	0,700	0,687
	5	**Evet**	0,700	0,688
		Hayır	0,686	0,669

Çizelge 3.8. Prewitt operatörü ve K-NN metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**1**	**Evet**	0,883	0,880
		Hayır	0,883	0,881
	3	**Evet**	0,876	0,870
		Hayır	0,879	0,876
	5	**Evet**	0,864	0,860
		Hayır	0,864	0,860
100	**1**	**Evet**	0,890	0,890
		Hayır	0,898	0,897
	3	**Evet**	0,883	0,878
		Hayır	0,895	0,893
	5	**Evet**	0,871	0,868
		Hayır	0,883	0,882
50	**1**	**Evet**	**0,917**	**0,916**
		Hayır	**0,917**	0,914
	3	**Evet**	0,890	0,886
		Hayır	0,902	0,899
	5	**Evet**	0,881	0,876
		Hayır	0,895	0,894
15	**1**	**Evet**	0,873	0,872
		Hayır	0,876	0,875
	3	**Evet**	0,871	0,868
		Hayır	0,867	0,863
	5	**Evet**	0,871	0,867
		Hayır	0,867	0,861
10	**1**	**Evet**	0,855	0,853
		Hayır	0,857	0,856
	3	**Evet**	0,871	0,868
		Hayır	0,860	0,856
	5	**Evet**	0,871	0,866
		Hayır	0,857	0,853
5	**1**	**Evet**	0,681	0,671
		Hayır	0,654	0,634
	3	**Evet**	0,698	0,685
		Hayır	0,676	0,654
	5	**Evet**	0,690	0,676
		Hayır	0,685	0,660

Çizelge 3.9. LoG operatörü ve K-NN metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**1**	**Evet**	**0,921**	**0,919**
		Hayır	0,914	0,914
	3	**Evet**	0,883	0,880
		Hayır	0,888	0,886
	5	**Evet**	0,870	0,865
		Hayır	0,871	0,867
100	**1**	**Evet**	0,902	0,901
		Hayır	0,902	0,901
	3	**Evet**	0,883	0,878
		Hayır	0,888	0,887
	5	**Evet**	0,888	0,885
		Hayır	0,876	0,874
50	**1**	**Evet**	0,888	0,887
		Hayır	0,881	0,879
	3	**Evet**	0,869	0,868
		Hayır	0,860	0,856
	5	**Evet**	0,864	0,861
		Hayır	0,860	0,858
15	**1**	**Evet**	0,886	0,888
		Hayır	0,876	0,878
	3	**Evet**	0,883	0,884
		Hayır	0,874	0,875
	5	**Evet**	0,879	0,880
		Hayır	0,874	0,875
10	**1**	**Evet**	0,898	0,898
		Hayır	0,898	0,897
	3	**Evet**	0,914	0,915
		Hayır	0,888	0,887
	5	**Evet**	0,895	0,896
		Hayır	0,879	0,879
5	**1**	**Evet**	0,700	0,670
		Hayır	0,686	0,655
	3	**Evet**	0,714	0,681
		Hayır	0,710	0,673
	5	**Evet**	0,685	0,647
		Hayır	0,686	0,643

Çizelge 3.10. Zero-Cross operatörü ve K-NN metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**1**	**Evet**	**0,921**	**0,919**
		Hayır	0,914	0,914
	3	**Evet**	0,883	0,880
		Hayır	0,888	0,886
	5	**Evet**	0,869	0,865
		Hayır	0,871	0,867
100	**1**	**Evet**	0,902	0,901
		Hayır	0,902	0,901
	3	**Evet**	0,883	0,878
		Hayır	0,888	0,887
	5	**Evet**	0,888	0,885
		Hayır	0,876	0,874
50	**1**	**Evet**	0,888	0,887
		Hayır	0,881	0,879
	3	**Evet**	0,869	0,868
		Hayır	0,860	0,856
	5	**Evet**	0,864	0,861
		Hayır	0,860	0,858
15	**1**	**Evet**	0,886	0,888
		Hayır	0,876	0,878
	3	**Evet**	0,883	0,884
		Hayır	0,874	0,875
	5	**Evet**	0,879	0,880
		Hayır	0,874	0,875
10	**1**	**Evet**	0,898	0,898
		Hayır	0,898	0,897
	3	**Evet**	0,914	0,915
		Hayır	0,888	0,887
	5	**Evet**	0,895	0,896
		Hayır	0,879	0,879
5	**1**	**Evet**	0,700	0,670
		Hayır	0,686	0,655
	3	**Evet**	0,714	0,681
		Hayır	0,710	0,673
	5	**Evet**	0,686	0,647
		Hayır	0,686	0,643

Çizelge 3.11. Sobel operatörü ve K-NN metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**1**	**Evet**	0,910	0,906
		Hayır	0,910	0,907
	3	**Evet**	0,890	0,885
		Hayır	0,900	0,896
	5	**Evet**	0,881	0,875
		Hayır	0,895	0,891
500	**1**	**Evet**	**0,924**	**0,921**
		Hayır	**0,924**	**0,921**
	3	**Evet**	0,905	0,900
		Hayır	0,912	0,908
	5	**Evet**	0,898	0,895
		Hayır	0,902	0,899
250	**1**	**Evet**	0,917	0,915
		Hayır	0,919	0,916
	3	**Evet**	0,921	0,920
		Hayır	**0,924**	**0,921**
	5	**Evet**	0,902	0,901
		Hayır	0,912	0,909
100	**1**	**Evet**	0,890	0,887
		Hayır	0,895	0,893
	3	**Evet**	0,890	0,888
		Hayır	0,890	0,888
	5	**Evet**	0,881	0,878
		Hayır	0,876	0,872
50	**1**	**Evet**	0,860	0,858
		Hayır	0,848	0,842
	3	**Evet**	0,843	0,838
		Hayır	0,850	0,842
	5	**Evet**	0,848	0,842
		Hayır	0,840	0,832
15	**1**	**Evet**	0,860	0,858
		Hayır	0,850	0,847
	3	**Evet**	0,843	0,838
		Hayır	0,831	0,828
	5	**Evet**	0,848	0,842
		Hayır	0,838	0,833
10	**1**	**Evet**	0,869	0,868
		Hayır	0,848	0,844
	3	**Evet**	0,847	0,845
		Hayır	0,840	0,837
	5	**Evet**	0,847	0,844
		Hayır	0,826	0,822
5	**1**	**Evet**	0,850	0,846
		Hayır	0,795	0,791
	3	**Evet**	0,840	0,832
		Hayır	0,769	0,761
	5	**Evet**	0,833	0,825
		Hayır	0,795	0,781

Çizelge 3.12. Prewitt operatörü ve K-NN metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**1**	**Evet**	0,893	0,886
		Hayır	0,897	0,892
	3	**Evet**	0,883	0,875
		Hayır	0,888	0,881
	5	**Evet**	0,879	0,871
		Hayır	0,888	0,882
500	**1**	**Evet**	0,912	0,910
		Hayır	0,914	0,910
	3	**Evet**	0,902	0,897
		Hayır	0,907	0,902
	5	**Evet**	0,905	0,902
		Hayır	0,898	0,893
250	**1**	**Evet**	0,902	0,901
		Hayır	**0,923**	**0,922**
	3	**Evet**	0,902	0,900
		Hayır	0,912	0,909
	5	**Evet**	0,895	0,894
		Hayır	0,907	0,906
100	**1**	**Evet**	0,893	0,890
		Hayır	0,900	0,898
	3	**Evet**	0,871	0,866
		Hayır	0,895	0,892
	5	**Evet**	0,869	0,863
		Hayır	0,879	0,874
50	**1**	**Evet**	0,886	0,884
		Hayır	0,886	0,882
	3	**Evet**	0,864	0,859
		Hayır	0,867	0,861
	5	**Evet**	0,850	0,841
		Hayır	0,852	0,846
15	**1**	**Evet**	0,857	0,838
		Hayır	0,852	0,850
	3	**Evet**	0,855	0,836
		Hayır	0,833	0,829
	5	**Evet**	0,857	0,853
		Hayır	0,826	0,821
10	**1**	**Evet**	0,862	0,860
		Hayır	0,829	0,829
	3	**Evet**	0,843	0,841
		Hayır	0,836	0,834
	5	**Evet**	0,838	0,833
		Hayır	0,836	0,832
5	**1**	**Evet**	0,829	0,828
		Hayır	0,817	0,816
	3	**Evet**	0,810	0,802
		Hayır	0,831	0,829
	5	**Evet**	0,800	0,793
		Hayır	0,831	0,828

Çizelge 3.13. LoG operatörü ve K-NN metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**1**	**Evet**	0,902	0,902
		Hayır	0,905	0,903
	3	**Evet**	0,900	0,899
		Hayır	0,895	0,893
	5	**Evet**	0,893	0,891
		Hayır	0,895	0,894
500	**1**	**Evet**	0,910	0,910
		Hayır	**0,914**	**0,915**
	3	**Evet**	0,902	0,902
		Hayır	0,910	0,909
	5	**Evet**	0,895	0,896
		Hayır	0,900	0,901
250	**1**	**Evet**	0,910	0,910
		Hayır	0,912	0,917
	3	**Evet**	0,895	0,895
		Hayır	0,893	0,893
	5	**Evet**	0,893	0,893
		Hayır	0,881	0,882
100	**1**	**Evet**	0,905	0,906
		Hayır	0,905	0,905
	3	**Evet**	0,893	0,894
		Hayır	0,886	0,886
	5	**Evet**	0,879	0,880
		Hayır	0,881	0,882
50	**1**	**Evet**	0,898	0,898
		Hayır	0,907	0,907
	3	**Evet**	0,883	0,883
		Hayır	0,893	0,891
	5	**Evet**	0,886	0,885
		Hayır	0,876	0,875
15	**1**	**Evet**	0,871	0,870
		Hayır	0,860	0,857
	3	**Evet**	0,871	0,870
		Hayır	0,864	0,861
	5	**Evet**	0,864	0,861
		Hayır	0,874	0,871
10	**1**	**Evet**	0,879	0,879
		Hayır	0,860	0,859
	3	**Evet**	0,867	0,866
		Hayır	0,838	0,836
	5	**Evet**	0,855	0,852
		Hayır	0,864	0,859
5	**1**	**Evet**	0,719	0,710
		Hayır	0,619	0,607
	3	**Evet**	0,717	0,705
		Hayır	0,669	0,653
	5	**Evet**	0,721	0,710
		Hayır	0,681	0,662

Çizelge 3.14. Zero-Cross operatörü ve K-NN metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	K Değeri	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**1**	**Evet**	0,902	0,902
		Hayır	0,905	0,903
	3	**Evet**	0,900	0,899
		Hayır	0,895	0,893
	5	**Evet**	0,893	0,891
		Hayır	0,895	0,894
500	**1**	**Evet**	0,910	0,910
		Hayır	0,914	0,915
	3	**Evet**	0,902	0,902
		Hayır	0,910	0,909
	5	**Evet**	0,895	0,896
		Hayır	0,900	0,901
250	**1**	**Evet**	0,910	0,910
		Hayır	**0,917**	**0,917**
	3	**Evet**	0,895	0,895
		Hayır	0,893	0,893
	5	**Evet**	0,893	0,893
		Hayır	0,881	0,882
100	**1**	**Evet**	0,905	0,905
		Hayır	0,905	0,905
	3	**Evet**	0,893	0,894
		Hayır	0,886	0,886
	5	**Evet**	0,879	0,880
		Hayır	0,881	0,882
50	**1**	**Evet**	0,898	0,898
		Hayır	0,907	0,907
	3	**Evet**	0,883	0,883
		Hayır	0,893	0,891
	5	**Evet**	0,886	0,885
		Hayır	0,876	0,875
15	**1**	**Evet**	0,871	0,870
		Hayır	0,860	0,857
	3	**Evet**	0,871	0,870
		Hayır	0,864	0,861
	5	**Evet**	0,864	0,864
		Hayır	0,874	0,871
10	**1**	**Evet**	0,879	0,879
		Hayır	0,860	0,859
	3	**Evet**	0,867	0,866
		Hayır	0,838	0,836
	5	**Evet**	0,855	0,852
		Hayır	0,864	0,859
5	**1**	**Evet**	0,719	0,710
		Hayır	0,619	0,607
	3	**Evet**	0,717	0,705
		Hayır	0,669	0,653
	5	**Evet**	0,721	0,
		Hayır	0,681	0,681

Çizelge 3.15. Sobel operatörü ve Naive Bayes metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,843	0,846
	Hayır	**0,848**	**0,851**
10	**Evet**	0,810	0,815
	Hayır	0,814	0,819
5	**Evet**	0,767	0,769
	Hayır	0,783	0,788

Çizelge 3.16. Prewitt operatörü ve Naive Bayes metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	**0,848**	**0,852**
	Hayır	0,824	0,826
10	**Evet**	0,821	0,826
	Hayır	0,800	0,805
5	**Evet**	0,814	0,819
	Hayır	0,757	0,763

Çizelge 3.17. LoG operatörü ve Naive Bayes metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,826	0,825
	Hayır	**0,845**	**0,846**
10	**Evet**	0,800	0,791
	Hayır	0,793	0,786
5	**Evet**	0,786	0,779
	Hayır	0,776	0,767

Çizelge 3.18. Zero-Cross operatörü ve Naive Bayes metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,826	0,825
	Hayır	**0,845**	**0,846**
10	**Evet**	0,800	0,791
	Hayır	0,793	0,786
5	**Evet**	0,786	0,779
	Hayır	0,776	0,767

Çizelge 3.19. Sobel operatörü ve Naive Bayes metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,881	0,883
	Hayır	0,878	0,880
100	**Evet**	**0,895**	**0,899**
	Hayır	0,879	0,880
50	**Evet**	0,881	0,882
	Hayır	0,883	0,885
15	**Evet**	0,879	0,880
	Hayır	0,871	0,872
10	**Evet**	**0,895**	0,895
	Hayır	0,892	0,892
5	**Evet**	0,699	0,639
	Hayır	0,667	0,638

Çizelge 3.20. Prewitt operatörü ve Naive Bayes metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,886	0,887
	Hayır	0,879	0,880
100	**Evet**	0,893	0,895
	Hayır	**0,905**	**0,907**
50	**Evet**	0,876	0,878
	Hayır	0,883	0,885
15	**Evet**	0,888	0,889
	Hayır	0,881	0,883
10	**Evet**	0,895	0,895
	Hayır	0,883	0,884
5	**Evet**	0,667	0,628
	Hayır	0,671	0,634

Çizelge 3.21. LoG operatörü ve Naive Bayes metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,881	0,882
	Hayır	0,876	0,877
100	**Evet**	**0,890**	**0,893**
	Hayır	0,881	0,884
50	**Evet**	0,869	0,871
	Hayır	0,861	0,864
15	**Evet**	0,879	0,880
	Hayır	0,874	0,875
10	**Evet**	0,871	0,873
	Hayır	0,869	0,870
5	**Evet**	0,660	0,649
	Hayır	0,650	0,631

Çizelge 3.22. Zero-Cross operatörü ve Naive Bayes metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,881	0,882
	Hayır	0,877	0,877
100	**Evet**	**0,890**	**0,893**
	Hayır	0,881	0,884
50	**Evet**	0,869	0,871
	Hayır	0,862	0,864
15	**Evet**	0,879	0,880
	Hayır	0,874	0,875
10	**Evet**	0,871	0,873
	Hayır	0,869	0,870
5	**Evet**	0,660	0,649
	Hayır	0,650	0,631

Çizelge 3.23. Sobel operatörü ve Naive Bayes metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,890	0,890
	Hayır	0,890	0,890
500	**Evet**	0,921	0,922
	Hayır	0,910	0,909
250	**Evet**	**0,943**	**0,944**
	Hayır	0,933	0,933
100	**Evet**	0,857	0,856
	Hayır	0,862	0,861
50	**Evet**	0,864	0,864
	Hayır	0,862	0,862
15	**Evet**	0,867	0,862
	Hayır	0,838	0,832
10	**Evet**	0,857	0,851
	Hayır	0,831	0,826
5	**Evet**	0,845	0,839
	Hayır	0,821	0,815

Çizelge 3.24. Prewitt operatörü ve Naive Bayes metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,898	0,897
	Hayır	0,886	0,885
500	**Evet**	0,929	0,929
	Hayır	0,919	0,919
250	**Evet**	**0,936**	**0,936**
	Hayır	0,933	0,934
100	**Evet**	0,864	0,862
	Hayır	0,874	0,873
50	**Evet**	0,867	0,866
	Hayır	0,860	0,859
15	**Evet**	0,855	0,848
	Hayır	0,855	0,858
10	**Evet**	0,848	0,839
	Hayır	0,850	0,846
5	**Evet**	0,819	0,806
	Hayır	0,850	0,848

Çizelge 3.25. LoG operatörü ve Naive Bayes metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,914	0,915
	Hayır	0,917	0,916
500	**Evet**	**0,921**	**0,922**
	Hayır	0,919	0,920
250	**Evet**	0,910	0,910
	Hayır	0,912	0,913
100	**Evet**	0,883	0,884
	Hayır	0,881	0,882
50	**Evet**	0,890	0,892
	Hayır	0,898	0,899
15	**Evet**	0,880	0,882
	Hayır	0,883	0,884
10	**Evet**	0,852	0,847
	Hayır	0,857	0,852
5	**Evet**	0,640	0,606
	Hayır	0,640	0,595

Çizelge 3.26. Zero-Cross operatörü ve Naive Bayes metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,914	0,915
	Hayır	0,917	0,916
500	**Evet**	**0,921**	**0,922**
	Hayır	0,919	0,920
250	**Evet**	0,910	0,910
	Hayır	0,912	0,913
100	**Evet**	0,883	0,884
	Hayır	0,881	0,882
50	**Evet**	0,890	0,892
	Hayır	0,898	0,899
15	**Evet**	0,881	0,882
	Hayır	0,883	0,884
10	**Evet**	0,852	0,847
	Hayır	0,857	0,852
5	**Evet**	0,640	0,606
	Hayır	0,640	0,595

Çizelge 3.27. Sobel operatörü ve ÇKA metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,876	0,875
	Hayır	**0,879**	**0,878**
10	**Evet**	0,871	0,870
	Hayır	0,860	0,858
5	**Evet**	0,814	0,810
	Hayır	0,833	0,831

Çizelge 3.28. Prewitt operatörü ve ÇKA metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,871	0,870
	Hayır	0,857	0,853
10	**Evet**	**0,876**	**0,876**
	Hayır	0,848	0,847
5	**Evet**	0,833	0,830
	Hayır	0,829	0,826

Çizelge 3.29. LoG operatörü ve ÇKA metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,893	0,893
	Hayır	0,886	0,885
10	**Evet**	0,881	0,881
	Hayır	**0,905**	**0,904**
5	**Evet**	0,869	0,868
	Hayır	0,852	0,851

Çizelge 3.30. Zero-Cross operatörü ve ÇKA metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,893	0,893
	Hayır	0,886	0,885
10	**Evet**	0,881	0,881
	Hayır	**0,905**	**0,904**
5	**Evet**	0,869	0,868
	Hayır	0,852	0,851

Çizelge 3.31. Sobel operatörü ve ÇKA metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,685	0,672
	Hayır	0,719	0,714
100	**Evet**	0,895	0,893
	Hayır	0,907	0,905
50	**Evet**	0,905	0,903
	Hayır	**0,912**	**0,910**
15	**Evet**	0,876	0,876
	Hayır	0,888	0,887
10	**Evet**	0,847	0,845
	Hayır	0,829	0,826
5	**Evet**	0,664	0,653
	Hayır	0,650	0,624

Çizelge 3.32. Prewitt operatörü ve ÇKA metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,690	0,685
	Hayır	0,728	0,729
100	**Evet**	0,881	0,880
	Hayır	0,869	0,869
50	**Evet**	0,886	0,884
	Hayır	**0,898**	**0,897**
15	**Evet**	0,879	0,877
	Hayır	0,873	0,873
10	**Evet**	0,836	0,832
	Hayır	0,838	0,833
5	**Evet**	0,650	0,629
	Hayır	0,614	0,582

Çizelge 3.33. LoG operatörü ve ÇKA metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,687	0,673
	Hayır	0,679	0,673
100	**Evet**	0,902	0,904
	Hayır	0,895	0,895
50	**Evet**	0,904	0,905
	Hayır	**0,936**	**0,936**
15	**Evet**	0,888	0,888
	Hayır	0,905	0,905
10	**Evet**	0,898	0,897
	Hayır	0,869	0,868
5	**Evet**	0,631	0,611
	Hayır	0,631	0,613

Çizelge 3.34. Zero-Cross operatörü ve ÇKA metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,686	0,673
	Hayır	0,679	0,673
100	**Evet**	0,902	0,904
	Hayır	0,895	0,895
50	**Evet**	0,905	0,905
	Hayır	**0,936**	**0,936**
15	**Evet**	0,888	0,888
	Hayır	0,905	0,905
10	**Evet**	0,888	0,897
	Hayır	0,869	0,868
5	**Evet**	0,631	0,611
	Hayır	0,631	0,613

Çizelge 3.35. Sobel operatörü ve ÇKA metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,733	0,728
	Hayır	0,202	0,195
500	**Evet**	0,912	0,911
	Hayır	0,602	0,591
250	**Evet**	**0,938**	**0,937**
	Hayır	0,819	0,823
100	**Evet**	0,924	0,922
	Hayır	0,914	0,914
50	**Evet**	0,864	0,863
	Hayır	0,874	0,872
15	**Evet**	0,800	0,790
	Hayır	0,788	0,780
10	**Evet**	0,795	0,791
	Hayır	0,783	0,778
5	**Evet**	0,795	0,780
	Hayır	0,750	0,737

Çizelge 3.36. Prewitt operatörü ve ÇKA metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,795	0,796
	Hayır	0,836	0,835
500	**Evet**	0,917	0,916
	Hayır	0,919	0,918
250	**Evet**	**0,933**	**0,933**
	Hayır	0,924	0,923
100	**Evet**	0,892	0,893
	Hayır	0,914	0,913
50	**Evet**	0,893	0,893
	Hayır	0,898	0,895
15	**Evet**	0,800	0,895
	Hayır	0,781	0,767
10	**Evet**	0,802	0,801
	Hayır	0,783	0,769
5	**Evet**	0,747	0,739
	Hayır	0,771	0,763

Çizelge 3.37. LoG operatörü ve ÇKA metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,221	0,224
	Hayır	0,207	0,208
500	**Evet**	0,602	0,594
	Hayır	0,592	0,594
250	**Evet**	0,807	0,810
	Hayır	0,874	0,875
100	**Evet**	0,926	0,926
	Hayır	0,929	0,929
50	**Evet**	0,926	0,926
	Hayır	**0,933**	**0,933**
15	**Evet**	0,883	0,883
	Hayır	0,855	0,853
10	**Evet**	0,883	0,883
	Hayır	0,845	0,842
5	**Evet**	0,640	0,618
	Hayır	0,621	0,690

Çizelge 3.38. Zero-Cross operatörü ve ÇKA metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,793	0,792
	Hayır	0,845	0,845
500	**Evet**	0,917	0,915
	Hayır	0,938	0,938
250	**Evet**	0,933	0,933
	Hayır	**0,940**	**0,940**
100	**Evet**	0,926	0,926
	Hayır	0,926	0,925
50	**Evet**	0,924	0,923
	Hayır	0,926	0,926
15	**Evet**	0,864	0,863
	Hayır	0,852	0,849
10	**Evet**	0,812	0,812
	Hayır	0,795	0,783
5	**Evet**	0,812	0,805
	Hayır	0,607	0,561

Çizelge 3.39. Sobel operatörü ve DVM metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	0,876	0,873
	Hayır	**0,880**	**0,879**
10	**Evet**	0,876	0,874
	Hayır	0,876	0,874
5	**Evet**	0,812	0,809
	Hayır	0,807	0,806

Çizelge 3.40. Prewitt operatörü ve DVM metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	**0,883**	0,879
	Hayır	**0,883**	**0,880**
10	**Evet**	0,876	0,873
	Hayır	0,871	0,869
5	**Evet**	0,831	0,829
	Hayır	0,812	0,807

Çizelge 3.41. LoG operatörü ve DVM metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	**0,895**	**0,892**
	Hayır	0,889	0,885
10	**Evet**	0,883	0,883
	Hayır	0,883	0,882
5	**Evet**	0,831	0,828
	Hayır	0,824	0,821

Çizelge 3.42. Zero-Cross operatörü ve DVM metodu kullanarak 2x2 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
15	**Evet**	**0,895**	**0,892**
	Hayır	0,888	0,885
10	**Evet**	0,883	0,883
	Hayır	0,883	0,882
5	**Evet**	0,831	0,828
	Hayır	0,824	0,821

Çizelge 3.43. Sobel operatörü ve DVM metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,902	0,903
	Hayır	0,914	0,915
100	**Evet**	0,914	0,914
	Hayır	0,919	0,919
50	**Evet**	0,910	0,909
	Hayır	**0,936**	**0,935**
15	**Evet**	0,874	0,872
	Hayır	0,869	0,867
10	**Evet**	0,881	0,879
	Hayır	0,862	0,860
5	**Evet**	0,657	0,621
	Hayır	0,659	0,620

Çizelge 3.44. Prewitt operatörü ve DVM metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,905	0,906
	Hayır	0,893	0,894
100	**Evet**	0,926	0,927
	Hayır	0,926	0,926
50	**Evet**	0,926	0,927
	Hayır	**0,929**	**0,929**
15	**Evet**	0,881	0,880
	Hayır	0,876	0,874
10	**Evet**	0,859	0,859
	Hayır	0,864	0,862
5	**Evet**	0,645	0,613
	Hayır	0,626	0,580

Çizelge 3.45. LoG operatörü ve DVM metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	**0,940**	**0,940**
	Hayır	0,933	0,933
100	**Evet**	0,938	0,938
	Hayır	0,926	0,926
50	**Evet**	0,926	0,926
	Hayır	0,926	0,924
15	**Evet**	0,912	0,912
	Hayır	0,923	0,924
10	**Evet**	0,914	0,914
	Hayır	0,879	0,875
5	**Evet**	0,664	0618,
	Hayır	0,660	0,600

Çizelge 3.46. Zero-Cross operatörü ve DVM metodu kullanarak 3x3 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
250	**Evet**	0,940	0,940
	Hayır	0,933	0,933
100	**Evet**	0,938	0,938
	Hayır	0,926	0,926
50	**Evet**	0,926	0,926
	Hayır	**0,944**	**0,944**
15	**Evet**	0,912	0,912
	Hayır	0,902	0,903
10	**Evet**	0,914	0,914
	Hayır	0,879	0,875
5	**Evet**	0,664	0,618
	Hayır	0,660	0,600

Çizelge 3.47. Sobel operatörü ve DVM metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,921	0,922
	Hayır	0,923	0,925
500	**Evet**	0,938	0,939
	Hayır	**0,950**	**0,950**
250	**Evet**	0,948	0,948
	Hayır	0,948	0,948
100	**Evet**	0,929	0,929
	Hayır	0,933	0,933
50	**Evet**	0,855	0,856
	Hayır	0,895	0,894
15	**Evet**	0,840	0,836
	Hayır	0,831	0,827
10	**Evet**	0,831	0,825
	Hayır	0,826	0,821
5	**Evet**	0,809	0,799
	Hayır	0,783	0,761

Çizelge 3.48. Prewitt operatörü ve DVM metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,914	0,915
	Hayır	0,910	0,910
500	**Evet**	0,936	0,936
	Hayır	0,933	0,934
250	**Evet**	0,943	0,943
	Hayır	**0,948**	**0,948**
100	**Evet**	0,905	0,904
	Hayır	0,919	0,919
50	**Evet**	0,890	0,890
	Hayır	0,900	0,899
15	**Evet**	0,833	0,828
	Hayır	0,814	0,808
10	**Evet**	0,831	0,827
	Hayır	0,817	0,810
5	**Evet**	0,781	0,761
	Hayır	0,800	0,791

Çizelge 3.49. LoG operatörü ve DVM metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,921	0,922
	Hayır	0,917	0,916
500	**Evet**	0,919	0,919
	Hayır	0,924	0,923
250	**Evet**	**0,936**	**0,936**
	Hayır	0,931	0,931
100	**Evet**	0,924	0,924
	Hayır	0,921	0,922
50	**Evet**	0,924	0,924
	Hayır	0,929	0,929
15	**Evet**	0,888	0,888
	Hayır	0,874	0,874
10	**Evet**	0,855	0,851
	Hayır	0,850	0,846
5	**Evet**	0,638	0,597
	Hayır	0,636	0,577

Çizelge 3.50. Zero-Cross operatörü ve DVM metodu kullanarak 4x4 çekirdek matrisi frekanslarıyla Doğruluk ve F-Ölçüsü sonuçları

Öznitelik Sayısı	İskeletleme	Doğruluk Oranı	F-Ölçüsü
1000	**Evet**	0,921	0,922
	Hayır	0,917	0,916
500	**Evet**	0,919	0,919
	Hayır	0,924	0,923
250	**Evet**	**0,936**	**0,936**
	Hayır	0,931	0,931
100	**Evet**	0,924	0,924
	Hayır	0,921	0,922
50	**Evet**	0,924	0,924
	Hayır	0,929	0,929
15	**Evet**	0,924	0,924
	Hayır	0,874	0,874
10	**Evet**	0,854	0,851
	Hayır	0,850	0,846
5	**Evet**	0,638	0,597
	Hayır	0,636	0,577

Çizelge 3.51. K-NN metodunun Sobel, Prewitt, LoG ve Zero-Cross filtrelerine göre en başarılı sonuçları

Çekirdek Matrisi: 2x2					
Filtre	**Öznitelik Sayısı**	**K**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	15	3	Evet	0,898	0,896
Prewitt	15	5	Evet	0,893	0,893
LoG	**15**	**1 ve 3**	**Evet**	**0,926**	**0,926**
Zero-Cross	**15**	**1 ve3**	**Evet**	**0,926**	**0,926**
Çekirdek Matrisi: 3x3					
Filtre	**Öznitelik Sayısı**	**K**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	100	1	Hayır	0,907	0,906
Prewitt	50	1	Evet/Hayır	0,917	0,916
LoG	250	1	Evet	0,921	0,919
Zero-Cross	250	1	Evet	0,921	0,919
Çekirdek Matrisi: 4x4					
Filtre	**Öznitelik Sayısı**	**K**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	250	1	Evet/Hayır	0,924	0,921
Prewitt	250	1	Hayır	0,923	0,922
LoG	500	1	Hayır	0,914	0,915
Zero-Cross	250	1	Hayır	0,917	0,917

Çizelge 3.52. Naive Bayes metodunun Sobel, Prewitt, LoG ve Zero-Cross filtrelerine göre en başarılı sonuçları

Çekirdek Matrisi: 2x2				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	15	Hayır	0,848	0,851
Prewitt	15	Evet	0,848	0,852
LoG	15	Hayır	0,845	0,846
Zero-Cross	15	Hayır	0,845	0,846
Çekirdek Matrisi: 3x3				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	100	Evet	0,895	0,899
Prewitt	100	Hayır	0,905	0,907
LoG	100	Evet	0,890	0,893
Zero-Cross	100	Evet	0,890	0,893
Çekirdek Matrisi: 4x4				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	**250**	**Evet**	**0,944**	**0,944**
Prewitt	250	Evet	0,936	0,936
LoG	500	Evet	0,921	0,922
Zero-Cross	500	Evet	0,921	0,922

Çizelge 3.53. ÇKA metodunun Sobel, Prewitt, LoG ve Zero-Cross filtrelerine göre en başarılı sonuçları

Çekirdek Matrisi: 2x2				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	15	Hayır	0,879	0,878
Prewitt	10	Evet	0,876	0,870
LoG	10	Hayır	0,905	0,904
Zero-Cross	10	Hayır	0,905	0,904
Çekirdek Matrisi: 3x3				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	50	Hayır	0,912	0,910
Prewitt	50	Hayır	0,898	0,897
LoG	50	Hayır	0,936	0,936
Zero-Cross	50	Hayır	0,936	0,936
Çekirdek Matrisi: 4x4				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	250	Evet	0,938	0,937
Prewitt	250	Evet	0,933	0,933
LoG	50	Hayır	0,933	0,933
Zero-Cross	**250**	**Hayır**	**0,940**	**0,940**

Çizelge 3.54. DVM metodunun Sobel, Prewitt, LoG ve Zero-Cross filtrelerine göre en başarılı sonuçları

Çekirdek Matrisi: 2x2				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	15	Hayır	0,880	0,879
Prewitt	15	Evet/Hayır	0,883	0,880
LoG	15	Evet	0,895	0,892
Zero-Cross	15	Evet	0,895	0,892
Çekirdek Matrisi: 3x3				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	50	Hayır	0,936	0,935
Prewitt	50	Hayır	0,929	0,929
LoG	250	Evet	0,940	0,940
Zero-Cross	50	Hayır	0,944	0,944
Çekirdek Matrisi: 4x4				
Filtre	**Öznitelik Sayısı**	**İskeletleme**	**Doğruluk**	**F-Ölçüsü**
Sobel	**500**	**Hayır**	**0,950**	**0,950**
Prewitt	250	Hayır	0,948	0,948
LoG	250	Evet	0,936	0,936
Zero-Cross	250	Evet	0,936	0,936

Çizelge 3.50'deki Doğruluk ve F-Ölçüsü değerlerinden Çizelge 51-54'deki en başarılı değerler tespit edilmiştir.

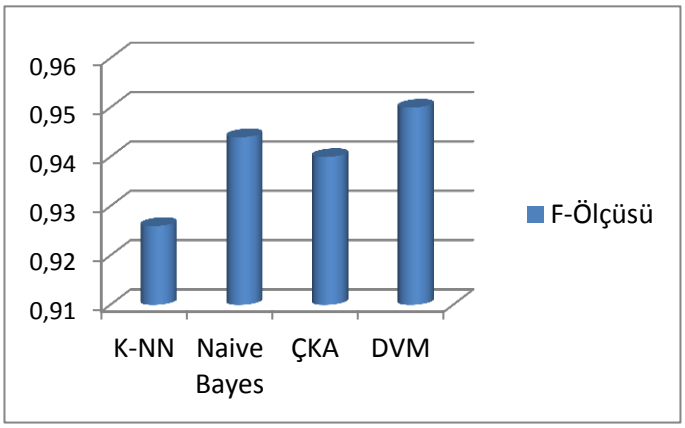
Çizelge 3.51'deki verilere göre K-NN metoduyla, Doğruluk ve F-Ölçüsü değerlerinden en iyi sonuç 0,926 olmuştur. Bu sonuç, LoG ve Zero-Cross filtresiyle, 15 öznitelik sayısıyla, K=1 alınarak, iskeletleme yapılarak ve 2x2 çekirdek matrisi kullanılarak elde edilmiştir.

Çizelge 3.52'de Naive Bayes metoduna göre, Doğruluk ve F-Ölçüsü değerlerinin sırasıyla 0,943 ve 0,944 olduğu görülebilir. Bu değerlere, Sobel filtresiyle, 250 öznitelik sayısıyla, iskeletleme yapılarak ve 4x4 çekirdek matrisi kullanarak ulaşılmıştır.

Çizelge 3.53'teki sonuçlara göre ÇKA metodu kullandığımızda Doğruluk ve F-Ölçüsü değerlerinden en başarılı olan değer 0,940 olmuştur. Bu sonuç, Zero-Cross filtresiyle, 250 öznitelik sayısıyla, 4x4 çekirdek matrisiyle ve iskeletleme yapılmadan elde ettik.

Çizelge 3.54'te görüldüğü üzere, DVM metodu kullanıldığında Doğruluk ve F-Ölçüsü sonuçlarının en başarılısı 0,950 olmuştur. Bu değere, Sobel filtresiyle, 500 öznitelik sayısıyla, 4x4 çekirdek matrisiyle ve iskeletleme yapılmayarak ulaşılmıştır.

2x2 çekirdek matrisi kullanıldığında en başarılı metot K-NN metodu (Doğruluk ve F-Ölçüsü: 0,926, Öznitelik Sayısı: 15, Filtre: LoG ve Zero-Cross, K:1 ve 3, İskeletleme: Evet) olmuştur. 3x3 çekirdek matrisiyle en iyi sonuç DVM metodu (Doğruluk ve F-Ölçüsü: 0,944, Öznitelik Sayısı: 50, Filtre: Zero-Cross, İskeletleme: Hayır) kullanılarak elde edilmiştir. DVM metodu (Doğruluk ve F-Ölçüsü: 0,950, Öznitelik Sayısı: 500, Filtre: Sobel, İskeletleme: Hayır) 3x3 çekirdek matrisinde olduğu gibi 4x4 matrisinde de en başarılı metot olduğu görülmektedir.



Şekil 3.2. Metotların başarı grafiği

Yapılan deneysel çalışma sonucunda, en başarılı çekirdek matrisleri genel olarak sırasıyla 4x4, 3x3 ve 2x2 olduğu söylenebilir. Şekil 3.2'de görüldüğü gibi en iyi sonuçların elde edildiği metotlar, DVM, Naive Bayes, ÇKA ve K-NN şeklinde sıralanmaktadır.

Çizelge 3.55. Çekirdek matrisleri için Sobel, Prewitt, LoG ve Zero-Cross filtrelerine göre önişleme ve frekans hesaplama zamanları

2x2 Çekirdek Matrisi		
Filtre	**İskeletleme**	**Gerçekleştirme Zamanı (Saniye)**
Sobel	Evet	13,9
	Hayır	13,8
Prewitt	Evet	13,7
	Hayır	13,6
LoG	Evet	13,7
	Hayır	13,7
Zero-Cross	Evet	13,8
	Hayır	13,7
3x3 Çekirdek Matrisi		
Filtre	**İskeletleme**	**Gerçekleştirme Zamanı (Saniye)**
Sobel	Evet	21,9
	Hayır	21,8
Prewitt	Evet	21,8
	Hayır	21,8
LoG	Evet	21,9
	Hayır	21,9
Zero-Cross	Evet	21,8
	Hayır	21,7
4x4 Çekirdek Matrisi		
Filtre	**İskeletleme**	**Gerçekleştirme Zamanı (Saniye)**
Sobel	Evet	125
	Hayır	126
Prewitt	Evet	122
	Hayır	123
LoG	Evet	123
	Hayır	123
Zero-Cross	Evet	129
	Hayır	126

Çizelge 3.56. Çekirdek matrisleri için Sobel filtresini kullanılarak sınıflandırma metotlarının eğitim ve test işlemlerinin süreleri

2x2 Çekirdek Matrisi		
Metot	**Öznitelik Sayısı**	**Gerçekleştirme Zamanı (Saniye)**
K-NN, K=5	5	< 0,01
	10	< 0,01
	15	< 0,01
Naive Bayes	5	< 0,01
	10	0,02
	15	0,02
ÇKA	5	9
	10	12
	15	15
DVM	5	0,06
	10	0,06
	15	0,08
3x3 Çekirdek Matrisi		
Metot	**Öznitelik Sayısı**	**Gerçekleştirme Zamanı (Saniye)**
K-NN, K=5	5	< 0,01
	10	< 0,01
	15	< 0,01
	50	< 0,01
	100	< 0,01
	250	< 0,01
Naive Bayes	5	< 0,01
	10	0,01
	15	0,01
	50	0,01
	100	0,01
	250	0,03
ÇKA	5	9
	10	12
	15	16
	50	34
	100	63
	250	144
DVM	5	0,08
	10	0,08
	15	0,08
	50	1
	100	2
	250	3

Çizelge 3.56. Çekirdek matrisleri için Sobel filtresini kullanılarak sınıflandırma metotlarının eğitim ve test işlemlerinin süreleri (devamı)

4x4 Çekirdek Matrisi		
Metot	**Öznitelik Sayısı**	**Gerçekleştirme Zamanı (Saniye)**
K-NN, K=5	5	< 0,01
	10	< 0,01
	15	< 0,01
	50	< 0,01
	100	< 0,01
	250	< 0,01
	500	< 0,01
	1000	< 0,01
Naive Bayes	5	< 0,01
	10	< 0,01
	15	0,01
	50	0,01
	100	0,01
	250	0,02
	500	0,04
	1000	0,09
ÇKA	5	8
	10	13
	15	17
	50	36
	100	67
	250	148
	500	313
	1000	745
DVM	5	0,08
	10	0,08
	15	0,08
	50	1
	100	2
	250	3
	500	3
	1000	4

Çizelge 3.55'te bir tekstil resmi için, Sobel, Prewitt, LoG ve zero-cross filtreleri kullanarak önişleme işlemlerinin yapılmasında ve 2x2, 3x3 ve 4x4 çekirdek matrislerinin frekanslarının hesaplanmasında geçen süreler görülmektedir. Burada yaklaşık olarak bütün filtreler benzer zamanlarda işlemleri gerçekleştirmektedir. Prewitt filtresi en kısa sürede bitirmektedir. Çekirdek matrisi büyükçe gerçekleştirme zamanı artmaktadır. 2x2 çekirdek matrisiyle yaklaşık 14 saniyede bir resim önişleme ve frekans hesaplama işleminde gerçekleştirme işlemi yapılırken 3x3 çekirdek matrisiyle 22 saniye sürmektedir. 4x4 matrisiyle ise 122 ile 129 saniye arasında gerçekleştirme zamanı bulunmuştur.

Çizelge 3.56'da sınıflandırma metotlarıyla 2x2, 3x3 ve 4x4 çekirdek matrislerinin frekans değerlerini kullanarak eğitim ve test işlemlerinin yapılmasında geçen süreler bulunmaktadır. Buradaki verilere göre en hızlı çalışan metotlar sırasıyla K-NN, Naive Bayes, DVM ve ÇKA olmaktadır. K-NN metoduyla bütün öznitelik sayılarında 0,01 saniyenin altında eğitim ve test işlemi sonuçlanmıştır. Hız bakımından ikinci sırayı alan Naive Bayes metodunun gerçekleştirme zamanı 0,09 saniyenin altında olmuştur. DVM metodu ile maksimum 4 saniyelik gerçekleştirme zamanı ölçülmüştür. 745 saniyelik gerçekleştirme süresiyle ÇKA metodu en fazla zaman alan metot olmuştur.

Çizelge 3.57. En başarılı sistemin yedi sınıfa göre hata matrisi

a	b	c	d	e	f	g	Aşağıdaki gibi sınıflandırıldı
60	0	0	0	0	0	0	**a=45 derece çizgili**
0	60	0	0	0	0	0	**b=135 derece çizgili**
0	0	58	0	1	1	0	**c=Çiçekli**
0	0	0	58	2	0	0	**d=Dikey çizgili**
0	1	5	0	51	3	0	**e=Ekose**
0	0	4	0	1	55	0	**f=Puantiyeli**
0	0	0	0	3	0	57	**g=Yatay**

Çizelge 3.57'de, DVM metodu ile en başarılı sistem olarak hata matrisi yer almaktadır. Burada her sınıfın kaçta kaçının doğru sınıflandırdığı ve yanlış sınıflandırmış ise hangi sınıfta yer aldığı açıkça görülmektedir. 45 ve 135 derece çizgili sınıfların her biri 60 resmin 60'ı da doğru sınıflandırılmıştır. Fakat, ekose sınıfındaki 60 resmin 55'i doğru yani ekoseli olarak, 4'ü çiçekli, 1'i dikey çizgili, 3'ü de puantiyeli olarak yanlış sınıflandırmıştır. En çok hatalı sınıflandırma ekose, sonrasında ise puantiyeli sınıflarda olmuştur.

4. TARTIŞMA VE SONUÇLAR

Bu tezde, tekstil resimlerinin K-NN, Naive Bayes, ÇKA ve DVM metotlarıyla "Çiçekli", "Puantiyeli", "Ekoseli", "Yatay çizgili", "Dikey çizgili", "45 derece çizgili" ve "135 derece çizgili" resimlerden hangi sınıfa ait olduğunu belirleyen çalışmalar yapıldı. Bunun için ilk olarak önişleme safhasında Sobel, Prewitt, LoG ve Zero-Cross filtresiyle resimlerin kenarları tespit edildi ve sonrasında 2x2, 3x3 ve 4x4 çekirdek matrislerinin resim içindeki frekansları hesaplatılıp normalize edilerek öznitelik vektörleri oluşturuldu. Sınıflandırma metotlarıyla sistemlerin başarıları 10-Katlı Çapraz Doğrulama kullanılarak Doğruluk ve F-Ölçüsü değerlerine göre karşılaştırılmıştır. En başarılı metotlar sırasıyla DVM, Naive Bayes, ÇKA ve K-NN olmuştur. DVM, 4x4 çekirdek matrisi kullanarak Sobel filtresiyle 0,95 Doğruluk ve F-Ölçüsü değeriyle en başarılı metot olmuştur. Çekirdek matrislerinde ise en yüksek başarı sırasıyla 4x4, 3x3 ve 2x2 boyutları olmuştur.

Bilgi kazancı (Information Gain) kullanılarak öznitelik vektörlerinin boyutu küçültülerek hem sistemlerin başarısı artırılmış hem de sistemlerin gerçekleştirme zamanı azaltılmıştır.

Daha sonraki çalışmalarda farklı sınıflandırma metotlarıyla, farklı kenar tespit filtreleri kullanılarak uygulamalar yapılıp var olan sistemlerle karşılaştırılacaktır. Aynı zamanda, farklı çekirdek matrisleriyle sistemler geliştirilecektir.

KAYNAKLAR

Akkuş, A & Güvenir, H. A. K-Nearest Neighbor Classification on Feature Projections. In Procedings of ICMI'96 Lorenzo Saitta (Ed.), Morgan Kaufmann, Bari, Italy, p. 12-19, 1996.

Anonim, 01.03.2004. Feature Detectors. (http://homepages.inf.ed.ac.uk/rbf/HIPR2/featops.htm), Erişim Tarihi: 09.06.2015.

Anonim, 15.02.2006. Laplacian/Laplacian of Gaussian. (http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm), Erişim Tarihi: 09.06.2015.

Anonim, (10.06.2015). Prewitt Operator. (https://en.wikipedia.org/wiki/Prewitt_operator) Erişim Tarihi: 10.06.2015.

Aşlıyan, R. 2002. Classification of Textile Images. Graduate School of Natural and Applied Sciences, Computer Engineering, Dokuz Eylül University, M.Sc. Thesis, İzmir.

Aşlıyan, R. ve Alpkoçak, 2002. A. Tekstil Desenlerinin Otomatik Olarak Sınıflandırılması Üzerine Bir Çalışma. SİU2002. 10. Sinyal İşleme ve İletişim Uygulamaları Kurultayı. Cilt I s. 123-128, Pamukkale, Denizli.

Aşlıyan, R. 2010. Textile Image Classification: Categorizing huge amout of textile images efficiently. Lambert Academic Publishing AG& CO. KG. Saarbrücken, Germany. ISBN: 978-3-8383-5732-4.

Atay, M. 2014. Real Time Skeletonization on Fpga With a Hand Tracking Application. Istanbul Technical University Institute of Science and Technology Department of Electronics and Communications Engineering Electronics Engineering Programme, M.Sc. Thesis, Istanbul.

Ateş, N. 2014. Destek Vektör Makineleri ve Gauss Karışım Modeli ile İstenmeyen E-postaların Tespiti. Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi, Isparta.

Aybar, E. 2008. Sobel İşleci Kullanılarak Renkli Görüntülerde Kenar Bulma. **Afyon Kocatepe Üniversitesi Fen Bilimleri Dergisi**, Cilt VIII, Sayı 1, 205-217, Afyon.

Ayhan, S. ve Erdoğmuş, Ş. 2014. Destek Vektör Makineleriyle Sınıflandırma Probleminin Çözümü İçin Çekirdek Fonksiyonu Seçimi. **Eskişehir Osmangazi Üniversitesi İİBF Dergisi,** 9(1): 175-198.

Bai, X. Latecki, L. and Yu Liu, W. 2007. Skeleton Pruning by Contour Partitioning with Discrete Curve Evolution, **Pattern Analysis and Machine Intelligence**, **IEEE Transactions on**, 29(3): 449–462.

Bayram, B., (13.04.2015). Sayısal Görüntü İşleme. (http://www.yildiz.edu.tr /~bayram/sgi/saygi.htm) Erişim Tarihi: 09.06.2015.

Bilgi, S. 2012. Çok Ölçekli Kartografik Gösterimlerde Mekansal Bilginin Nicelik Analizi. İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Geomatik Mühendisliği Programı Gemomatik Mühendisliği Anabilimdalı, Doktora Tezi, sf. 13-15, İstanbul.

Blum, H., 1967. A Transformation for Extracting New Descriptors of Shape, **Models for the Perception of Speech and Visual Form**, MIT Press, Cambridge, pp.362–380.

Borgefors, G. 1986. Distance Transformations in Digital Images. **Computer Vision, Graphics, and Image Processing**, 34(3): 344 – 371.

Boyle, R. and Thomas, R. 1988. Computer Vision: A First Course. **Blackwell Scientific Publications**, pp 48 - 50.

Burges, C. J. C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition, **Knowledge Discovery and Data Mining**: 2(2).

Chtioui, Y. Bertrand, D. Devaux, M.F. and Barba, D. 1997. Comparison of Multilayer Perceptron and Probabilistic Neural Networks in Artificial Vision. Application to the Discrimination of Seeds, **Journal of Chemometrics**, 11(2):111-129.

Clark, J. J. 1989. Authenticating Edges Produced by Zero-Crossing Algorithms, **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2(1), 43-57.

Curic, V. (06.02.2013). Mathematical Morphology and Distance Transforms. (http://www.it.uu.se/edu/course/homepage/bild1/vt13/mathematical_morphology_and_distance_transforms.pdf). Uppsala University, Erişim Tarihi: 09.06.2015.

Curic, V. 2014. Mathematical Morphology. Distance Functions and Their Use in Adaptive Mathematical Morpholgy, Uppsala University Department of Information Technology, Ph. D. Thesis, pp. 33-40, Sweden.

Dimitrov, P. Phillips, C. and Siddiqi, K. 2000. Robust and Efficient Skeletal Graphs, Computer Vision and Pattern Recognition. **Proceedings. IEEE Conference on**, volume 1: pp.417–423.

Dong, Q. Y. Song, C. C. Ben, C. S. Quan, L. J. 2005. A Fast Subpixel Edge Detection Method Using Sobel–Zernike Moments Operator, **Image and Vision Computing Journal-Elsevier**, 23: 11-17.

Efe, M.Ö. and Kaynak, O. 2000. Artificial Neural Networks, **Boğaziçi University Publishing**, İstanbul.

Elmas, Ç. 2003. Artificial Neural Networks, Seçkin Yayıncılık, Ankara.

Geyers, T. Aldershoff, F. and Smeulders, A.W.M. 2000. Classification of Images on Internet by Visual and Textual Information, **In SPIE Vol: 3964:** doi: 10.1117/12.373453, San Jose.

Gonzalez, R.C. Woods, R. 2008. Introduction. Digital Image Processing. 3rd Edition, Prentice-Hall, New Jersey.

Gutierez-Osuna, R. (03.08.2002). Lecture 13: Validation. (http://research.cs.tamu.edu/prism/lectures/iss/iss_l13.pdf) Erişim Tarihi: 01.06.2015.

Gündüz, H. 2013. Borsa İstanbul (BİST) 100 Endeksi Yönünün Ekonomi Haberleri ile Tahmin Edilmesi. İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, sf. 17-18, İstanbul.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H. 2009. The WEKA Data Mining Software. An Update, SIGKDD Explorations, 11(1).

Haykin, S. 1994. Neural Networks: A Comprehensive Foundation, **Macmillan College Publishing Company**, New York.

Heijden, F. (1995). Edge and Line Feature Extraction Based on Covariance Models, IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(1): 16-33.

Kim, S.B., Rim, H.C., Yook, D., Lim, H.S. 2002. Effective Methods for İmproving Naive Bayes Text Classifiers, **In The 7th Pacific Rim International Conference on Artificial Intelligence**, pp. 414–423.

Kim, S. 2013. Applications of Convolution in Image Processing with MATLAB. University of Washington, pp. 5,6, USA.

Klette, R. 2014. Concise Computer Vision. Springer, London, ISBN 978-1-4471-6320-6.

Kohonen, T. 1995. Learning Vector Quantization. The Handbook of Brain Theory and Neural Networks, (Editor) Arbib, M.A., MIT Press, p. 537-540, Cambridge.

Konishi, S., Yuille, A. L., Coughlan, J. M., Zhu, S. C. 2003. Statistical Edge Detection: Learning and Evaluating Edge Cues. **IEEE Transactions on,** Pattern Analysis and Machine Intelligence, 25(1): 57-74.

Lohman,G., 1998. Volumetric Image Analysis, Wiley.

Özkan, Y. 2013. Veri Madenciliği Yöntemleri. Dr. Rifat Çölkesen Dr. Cengiz Uğurkaya Papatya Yayıncılık, İstanbul.

Özkul, S. M. 1995. Tek Kamera ile Görüntüde Derinliğin Hesaplanması. Osmangazi Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği, Yüksek Lisans Tezi, Eskişehir.

Öztemel, E., Yapay Sinir Ağları, Papatya Yayınevi, İstanbul, 2006.

Öztemel, E. 2012. Yapay Sinir Ağları, Papatya Yayıncılık Eğitim, Türkiye.

Palágyi, K., (23.02.2001).Skeletonization (http://www.inf.u-szeged.hu /~palagyi/skel/skel#Voronoi) date retreived: 09.06.2015

Prewitt, J. M. S. 1970. Object Enhancement and Extraction. Picture Processing and Psychopictorics. **Editors Lipkin, B., Rosenfeld, A., Academic Press,** New York.

Roobaert, D., Karakulas, G., Chavla N. V., 2006. Information Gain. Correlation and Support Vector Machines. **Editors Guyon, I., Nikravesh, M., Gunn, S., Zadeh, L. A., Springer Berlin Heidelberg,** Volume 207, pp. 463–470.

Rummelhart, D. E., Hinton, G. E., Wiliams, R. J. 1986. Learning Represantations by Propogation Errors, Nature, 323: 533-536.

Sezer, O.G., Erçil, A., Keskinöz, M. 2005. Destek Vektör Makinesi Kullanarak Bağımsız Bileşen Tabanlı 3B Nesne Tanıma, SUI 2005. Sabancı Üniversitesi Mühendislik ve Doğa Bilimleri Fakültesi.

Siddiqi, K. and Pizer, S., 2008. Introduction. Medial Representations: Mathematics, Algorithms and Applications, editors Kalem Siddiqi and Stephen M. Pizer, pp. 4-7, Springer.

Sobel, I. ve Feldman, G. 1968. A 3x3 Isotropic Gradient Operator For Image Processing, John Wiley and Sons, New York, US.

Tagliasacchi, A., 2013. Skeleton Extraction and Skeleton-Driven Processing of Incomplete Data,Simon Fraser University School of Computing Science Faculty of Applied Sciences, Ph.D. Thesis, pp.12-16, Canada.

Timur, E. ve Sarı, C. 2010. Agora (Magnesia/Aydın) Manyetik Verilerinin Kenar Belirleme İşleçleri ve 3-Boyutlu Ters Çözümle Modellenmesi, **Hacettepe Üniversitesi Yerbilimleri Uygulama ve Araştırma Merkezi Dergisi**, 31(2): 67-82.

Uğur A., (20.03.2013). Görüntü İşlemeye Giriş. (http://yzgrafik.ege.edu.tr /~ugur/12_13_Spring/CI/ImageProcessing.pdf) Erişim tarihi: 09.06.2015.

Uludağ, A. K., 2005. Doğrusal Regresyon Modellerinde Çapraz Geçerlilik Yöntemleri. Hacettepe Üniversitesi Sağlık Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara

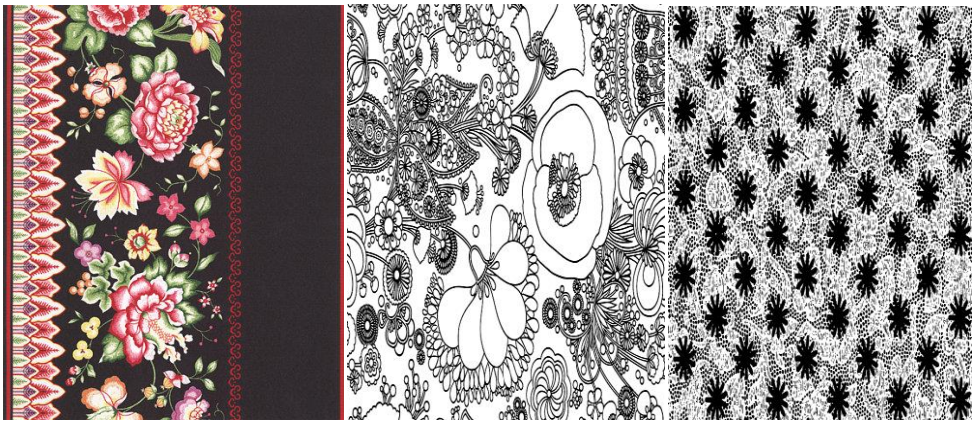
Ulvi İ., Aşlıyan R., Günel K. 2013. Textile Image Classification Using Artificial Neural Networks, **3rd World Conference on Innovation and Computer Science (INSODE - 2013),** Antalya, Turkey.

Yılmaz, R. 2013. Türkçe Dökümanların Sınıflandırılması. Adnan Menderes Üniversitesi Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Aydın.

EKLER

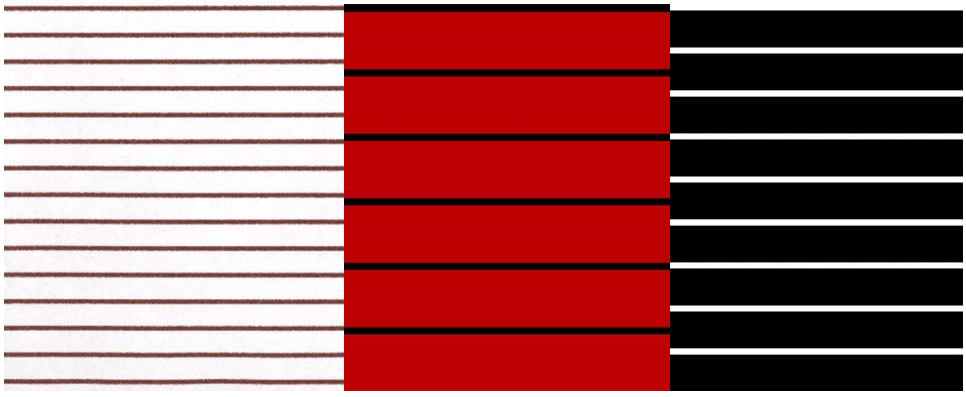
EK 1. Tekstil Resim Örnekleri

Çiçekli Resimler

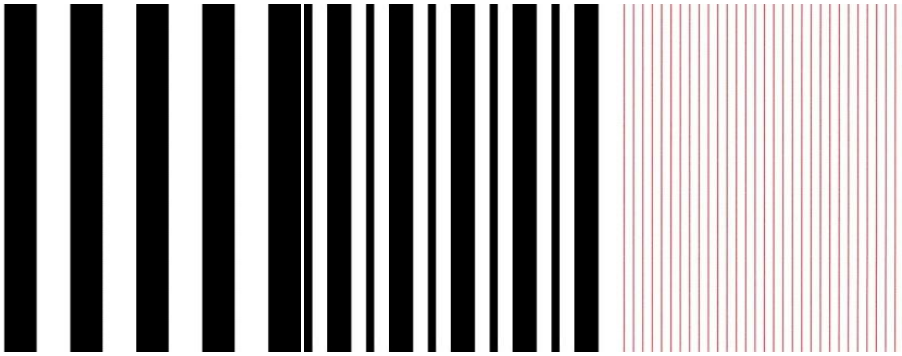


Puantiyeli Resimler

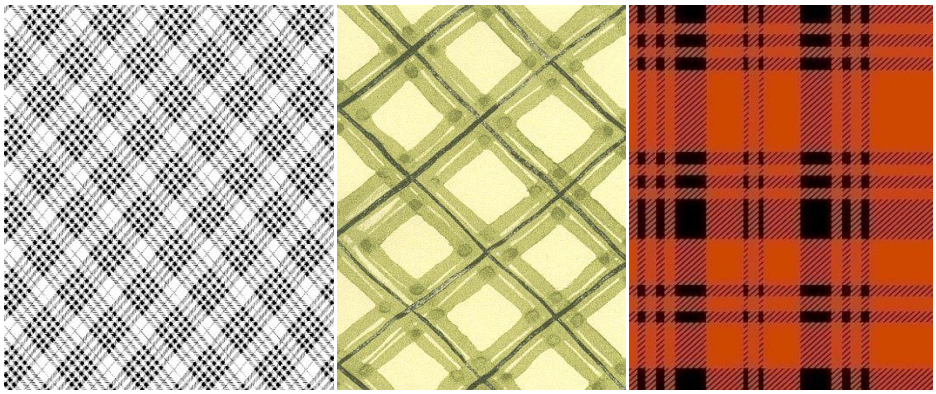
Yatay Çizgili Resimler



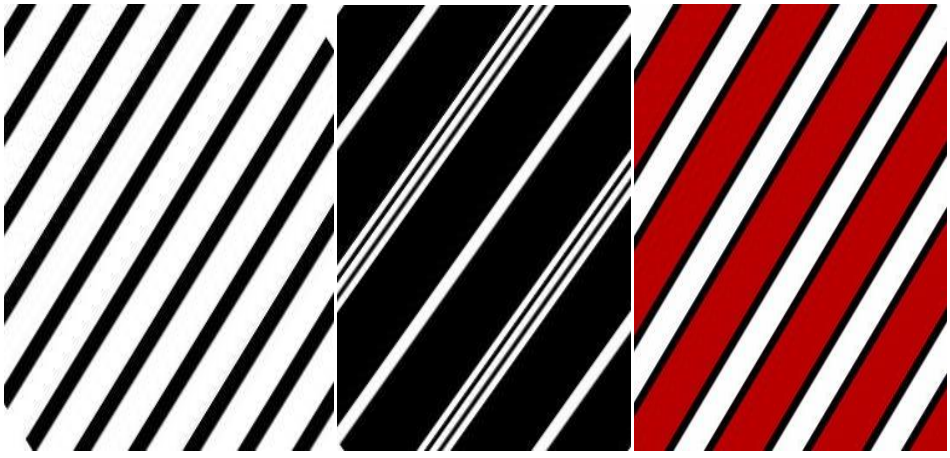
Dikey Çizgili



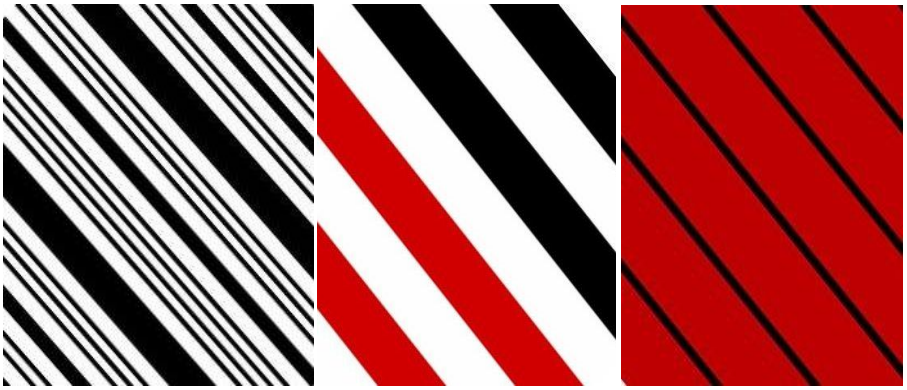
Ekoseli Resimler



45 Derece Çizgili Resimler



135 Derece Çizgili Resimler



EK 2. Önişleme ve Öznitelik Çıkarımı Programları

Farklı Çekirdek Matrislerini Bulan Program

```
clear all
clc
load DatasetVeri.mat; DatasetVeri;
DatasetVeri.toplamornek=420;
DatasetVeri.hersinifornek=60;
DatasetVeri.sinifisimleri={'45DegreeStriped' '135DegreeStriped' 'Flowery'
'VerticalStriped' 'Plaid' 'Spotted' 'HorizontalStriped'};
satir=DatasetVeri.KernelSatir;
sutun=DatasetVeri.KernelSutun;
FarkliKerneller={};
k1=1;
for k=0:(2^(satir*sutun) -1)
   ikilisayi=dec2bin(k,(satir*sutun));
   say=1;
   for i=1:satir
      for j=1:sutun
      A(i,j)= bin2dec(ikilisayi(say));
      say=say+1;
      end
   end
   FarkliKerneller{k1}=A;
   k1=k1+1;
end
DatasetVeri.FarkliKerneller=FarkliKerneller;
save DatasetVeri.mat DatasetVeri;
```

Çekirdek Matrislerini Sıralayan Program

```
clear all
clc
global dizi;
 load DatasetVeri.mat; %DatasetVeri.KernelSatir,
DatasetVeri.KernelSutun,DatasetVeri.sinifisimleri,
DatasetVeri.hersinifornek
FarkliKerneller=DatasetVeri.FarkliKerneller;
for i=1:length(FarkliKerneller)
   FarkliKernellerSirali{i,1}=FarkliKerneller{i};
   FarkliKernellerSirali{i,2}= KernelToCharRowVector(FarkliKerneller{i}) ;
end
 dizi=FarkliKernellerSirali;
 hizlisirala(1,size(dizi,1),2) ;
 clear FarkliKernellerSirali;
 DatasetVeri.FarkliKernellerSirali=dizi;
 save DatasetVeri.mat DatasetVeri;
```

Çekirdek Matrislerinin Frekanslarını Bulan Program

```
clear all
clc
 load DatasetVeri.mat; %DatasetVeri.KernelSatir,
DatasetVeri.KernelSutun,DatasetVeri.sinifisimleri,
DatasetVeri.FarkliKernellerSirali,DatasetVeri.toplamornek,
DatasetVeri.hersinifornek
FarkliKernellerSirali=DatasetVeri.FarkliKernellerSirali;
kernel.satir=DatasetVeri.KernelSatir;
kernel.sutun=DatasetVeri.KernelSutun;
global AramaDizisi;
uzunluk=size(FarkliKernellerSirali,1);
for i=1:uzunluk
 AramaDizisi{i}=FarkliKernellerSirali{i,2};
end
 islemler={'textileimage'};
TextileDataset1=zeros(uzunluk,DatasetVeri.toplamornek;
```

```
tic
sutunRes=1;
for islemsay=1:length(islemler)
clear klasorler sinifisimleri

klasorler={['Resimler\' islemler{islemsay} '\45 Derece Çizgili'] ['Resimler\'
islemler{islemsay} '\135 Derece Çizgili'] ...
   ['Resimler\' islemler{islemsay} '\Çiçekli'] ['Resimler\' islemler{islemsay}
'\Dikey'] ['Resimler\' islemler{islemsay} '\Ekose'] ...
  ['Resimler\' islemler{islemsay} '\Puantiyeli'] ['Resimler\' islemler{islemsay}
'\Yatay']};
klasorsay=size(klasorler,2);
for klasor_i=1:klasorsay
clear resimdosyalar hangiklasor
hangiklasor=klasorler{klasor_i};
resimdosyalar=dir ([hangiklasor '\*.jpg']);
dosyasay=length(resimdosyalar);
fprintf('Toplam dosya: %d \n',dosyasay);
for i=1:dosyasay
 clear dosyayolu I BW1 IGray info;
dosyayolu=[hangiklasor '\' resimdosyalar(i).name];
 fprintf('%d  %s\n',i,dosyayolu)
 edgekernel=DatasetVeri.edgekernel;
 skeletonization=DatasetVeri.skeletonization;
 [BW1,info]=Preprocessing(dosyayolu,edgekernel,skeletonization);
ksat=DatasetVeri.KernelSatir;
ksut=DatasetVeri.KernelSutun;
rsat=size(BW1,1);
rsut=size(BW1,2);
for k=1:(rsat-ksat+1)
    for j=1:(rsut-ksut+1)
      Y=BW1(k:k+ksat-1,j:j+ksut-1);
      AranankKernel=KernelToCharRowVector(Y);
      bulunanKernelIndex=ikiliarama(AranankKernel,1,uzunluk);
      if (bulunanKernelIndex ~= -1)
        TextileDataset1(bulunanKernelIndex,sutunRes)
```

```
      =TextileDataset1(bulunanKernelIndex,sutunRes)+1;
     end
   end
end
TextileDataset1(uzunluk+1,sutunRes) =size(BW1,1);
TextileDataset1(uzunluk+2,sutunRes) =size(BW1,2);
sutunRes=sutunRes+1;
end
end
end
save TextileDataset1.mat TextileDataset1;
sifirvektor=zeros(1,DatasetVeri.toplamornek);
YeniTextileDataset1=[];
say=1;
for g=1:size(TextileDataset1,1)-2
   disp(g)
   DatasetVektoru=TextileDataset1(g,:);
   if (DatasetVeri.toplamornek~=sum(sifirvektor==DatasetVektoru))
YeniTextileDataset1=[YeniTextileDataset1; DatasetVektoru];
      YeniFarkliKerneller{say}=DatasetVeri.FarkliKerneller{g};
      YeniFarkliKernellerSirali{say,1}=DatasetVeri.FarkliKernellerSirali{g,1};
      YeniFarkliKernellerSirali{say,2}=DatasetVeri.FarkliKernellerSirali{g,2};
      say=say+1;
   end
end
YeniTextileDataset1=[YeniTextileDataset1; TextileDataset1(g+1,:)];
YeniTextileDataset1=[YeniTextileDataset1; TextileDataset1(g+2,:)];
 clear TextileDataset1;
TextileDataset1=YeniTextileDataset1
DatasetVeri.FarkliKerneller=[];
DatasetVeri.FarkliKernellerSirali=[];
DatasetVeri.FarkliKerneller=YeniFarkliKerneller;
DatasetVeri.FarkliKernellerSirali=YeniFarkliKernellerSirali;
sinifisimleri=DatasetVeri.sinifisimleri;
siniforneksay=DatasetVeri.hersinifornek;
for j=1:size(TextileDataset1,2)
```

```
    toplampiksel=TextileDataset1(end,j)*TextileDataset1(end-1,j);
    if j<=siniforneksay
      sinif=sinifisimleri{1};
    elseif j<=(2*siniforneksay)
      sinif=sinifisimleri{2};
    elseif j<=(3*siniforneksay)
     sinif=sinifisimleri{3};
    elseif j<=(4*siniforneksay)
      sinif=sinifisimleri{4};
    elseif j<=(5*siniforneksay)
      sinif=sinifisimleri{5};
    elseif j<=(6*siniforneksay)
      sinif=sinifisimleri{6};
    elseif j<=(7*siniforneksay)
      sinif=sinifisimleri{7};
    end
  for i=1:size(TextileDataset1,1)-2
      TextileDataset{i,j}= TextileDataset1(i,j)/toplampiksel;
  end
  TextileDataset{i+1,j}=sinif;
end
DatasetVeri.TextileDataset=TextileDataset;
 save DatasetVeri.mat DatasetVeri;
toc
```

ARFF Formatına Dönüştüren Program

```
clear all
clc
load DatasetVeri.mat;
FarkliKernellerSirali=DatasetVeri.FarkliKernellerSirali;
kernel.satir=DatasetVeri.KernelSatir;
kernel.sutun=DatasetVeri.KernelSutun;
sinifisimleri= DatasetVeri.sinifisimleri;
TextileDataset=DatasetVeri.TextileDataset;
if DatasetVeri.skeletonization==1
  skleton='SkelYes';
```

```
  reply='Yes';
else
  skleton='SkelNo';
  reply='No';
end
kernelismi=[num2str(DatasetVeri.KernelSatir) 'x'
num2str(DatasetVeri.KernelSutun) 'kernel'];
dosyaismi=['TextileDataset_' kernelismi '_' DatasetVeri.edgekernel '_' skleton
'.arff'];
fid = fopen(dosyaismi,'w');
fprintf(fid,'%% 1. Title: Textile Image Database\n');
fprintf(fid,'%%\n');
fprintf(fid,'%% 2. Features:\n');
fprintf(fid,'%%      (a) Kernel Applied: %s\n',kernelismi);
fprintf(fid,'%%      (b) Edge Detection Operator: %s\n',DatasetVeri.edgekernel);
fprintf(fid,'%%      (c) Skeletonization: %s\n',reply);
fprintf(fid,'%%\n');
fprintf(fid,'%% 3. Sources:\n');
fprintf(fid,'%%      (a) Creator: R. Aşlıyan\n');
fprintf(fid,'%%      (b) Date: january, 2015\n');
fprintf(fid,'%%\n');
fprintf(fid,'@RELATION textile\n');
for i=1:size(FarkliKernellerSirali,1)
  fprintf(fid,'@ATTRIBUTE %s NUMERIC\n',FarkliKernellerSirali{i,2});
end
 fprintf(fid,'@ATTRIBUTE class ');
 fprintf(fid,'{');
 for i=1:length(sinifisimleri)
  fprintf(fid,'%s,',sinifisimleri{i});
 end
 fprintf(fid,'}\n');
fprintf(fid,'@DATA\n');
for j=1:size(TextileDataset,2)
 for i=1:size(TextileDataset,1)-1
   fprintf(fid,'%1.10f,',TextileDataset{i,j});
 end
```

```
 fprintf(fid,'%s\n',TextileDataset{i+1,j});
end
fclose(fid);
```

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Ömer KALFA
Doğum Yeri ve Tarihi : 12.12.1988

EĞİTİM ve MESLEKİ DURUMU

Lisans Öğrenimi : Anadolu Üniversitesi Fen Fakültesi Matematik Bölümü
Yüksek Lisans Öğrenimi : Adnan Menderes Üniversitesi-Matematik A.B.D.
Bildiği Yabancı Diller : İngilizce
Mesleği : MEB'de Matematik Öğretmeni

BİLİMSEL FAALİYETLERİ

Bildiriler : -
Katıldığı Projeler : Güncel metotlarla resim sınıflandırma, Bilimsel Araştırma Projeleri (BAP), Adnan Menderes Üniversitesi, 2015

İLETİŞİM

E-posta Adresi : omerkalfa1@gmail.com
Tarih : 15.07.2015