T.C.
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GÖRÜNTÜ İŞLEME YÖNTEMLERİYLE TEMEL DEVRE ELEMANLARININ SINIFLANDIRILMASI

YÜKSEK LİSANS TEZİ

Mihriban GÜNAY

Elektrik Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Murat KÖSEOĞLU

TEMMUZ 2021

T.C
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GÖRÜNTÜ İŞLEME YÖNTEMLERİYLE TEMEL DEVRE ELEMANLARININ SINIFLANDIRILMASI

YÜKSEK LİSANS TEZİ

Mihriban GÜNAY
(36183615045)

Elektrik Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Murat KÖSEOĞLU

TEMMUZ 2021

TEŞEKKÜR VE ÖNSÖZ

Bu tez çalışmasında öneri, bilgi, tecrübe ve desteklerini esirgemeden beni her konuda yönlendiren danışman hocam Sayın Dr. Öğr. Üyesi Murat KÖSEOĞLU'na,

Güleryüzü ile içimi ısıtan, hayatım boyunca maddi ve manevi her konuda desteğini bir an olsun esirgemeyen canım babama,

Şefkati, azmi, sabrı, başarısı ve duaları ile hayatım boyunca her zaman yanımda olan, tüm başarılarımın gerçek sahibi canım anneme,

Sahip olduğum tek kardeşim, üniversite arkadaşım, meslektaşım olan canım ablam Araş. Gör. Miray GÜNAY BULUT'a

Yüksek lisans tez sürecinde çalışmalarımda bana yardımcı olan ve yol gösteren değerli hocam Sayın Doç. Dr. Özal Yıldırım'a

Yüksek lisans eğitim sürecinde 2211 Yurt İçi Lisansüstü Burs Programı kapsamında bursiyer olarak maddi destek veren TÜBİTAK'a

teşekkür ederim.

ONUR SÖZÜ

Yüksek lisans tezi olarak sunduğum "Görüntü İşleme Yöntemleriyle Temel Devre Elemanlarının Sınıflandırılması" başlıklı bu çalışmanın bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurmaksızın tarafımdan yazıldığına ve yararlandığım bütün kaynakların hem metin içinde hem de kaynakçada yöntemine uygun biçimde gösterilenlerden oluştuğunu belirtir, bunu onurumla doğrularım.

Mihriban GÜNAY

İÇİNDEKİLER

ÇİZELGELER DİZİNİ

ŞEKİLLER DİZİNİ

SEMBOLLER VE KISALTMALAR

BG : Bilgisayarlı Görü

1B : Bir Boyutlu

2B : İki Boyutlu

3B : Üç Boyutlu

CNN : Convolutional Neural Network

ESA : Evrişimli Sinir Ağları

B-ESA : Bölge Tabanlı Evrişimli Sinir Ağları

Hızlı B-ESA : Hızlı Bölge Tabanlı Evrişimli Sinir Ağları

Daha Hızlı B-ESA : Daha Hızlı Bölge Tabanlı Evrişimli Sinir Ağları

CPMC : Constrained Parametric Min-Cuts

DPM : Deformable Part Model

DVM : Destek Vektör Makineleri

DSA : Derin Sinir Ağı

GPU : Graphics Processing Unit

HMM : Gizli Markov Modeli

KA : Karar Ağacı

KNN : K-En Yakın Komşu

HOG : Yönlendirilmiş Gradyan Özelliklerinin Histogramı

MRG : Manyetik Rezonans Görüntüleme

MaxPooling : Maksimum Havuzlama

R-CNN : Region Based Convolutional Neural Network

Fast R-CNN : Fast Region Based Convolutional Neural Network

Faster R-CNN : Faster Region Based Convolutional Neural Network

ROI : Reigions Of Interests

RPN : Region Proposal Network

SIFT : Scale-Invariant Feature Transform

YSA : Yapay Sinir Ağı

XML : Extensible Markup Language

α_i : Lagrange katsayısı

B : Bias değeri

λ_i : Lagrange katsayısı

φ : Kernel fonksiyonu

$\{x_i, y_i\}$: Eğitim verileri

M : Örnek sayısı

w	: Filtre parametresi
S	: Sınıf sayısı
s	: Adım sayısı
W, H	: Matris boyutları
K, L	: Filtre boyutları
p, q, r	: Filtre endeksleri
v	: Özellik haritasının çıktı değeri
$sig(x)$	: Sigmoid aktivasyon fonksiyonu
$tanh(x)$	: Tanh aktivasyon fonksiyonu
$ReLU(x)$	: ReLU aktivasyon fonksiyonu
$softmax(x)$	: Softmax aktivasyon fonksiyonu

ÖZET

Yüksek Lisans Tezi

GÖRÜNTÜ İŞLEME YÖNTEMLERİYLE
TEMEL DEVRE ELEMANLARININ SINIFLANDIRILMASI

İnönü Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik Elektronik Mühendisliği Anabilim Dalı

68+xi sayfa

2021

Danışman: Dr. Öğr. Üyesi Murat KÖSEOĞLU

Elektrik Elektronik Mühendisleri, kağıda elle çizilmiş devre gördüklerinde devre elemanlarını kolaylıkla tanıyıp ayırt edebilirler. Fakat bilgisayarların ve makinelerin elle çizilmiş devre elemanlarını sınıflandırması ve elle çizilmiş devreler üzerinden devre elemanlarını tepit etmesi zordur. Bu amaca yönelik olarak, literatürde farklı yöntemler kullanılarak elle çizilmiş devre elemanlarının sınıflandırılması ve tanınması üzerine yapılmış bazı çalışmalar bulunmaktadır.

Bu tez çalışmasında, elle çizilmiş devre elemanlarını sınıflandırmak ve bu devre elemanlarını devre üzerinden tespit edebilmek için iki farklı deneysel çalışma yapılmıştır. Yapılan ilk deneysel çalışmada, Evrişimli Sinir Ağları (ESA) kullanılarak dört farklı model oluşturulmuştur ve elle çizilmiş devre elemanları sınıflandırılmıştır. Bu oluşturulan dört farklı ESA modelinin performansları karşılaştırılmıştır. Yapılan deneysel çalışmanın sonucunda, kullanılan yöntem elle çizilmiş devre elemanlarının sınıflandırılmasında yüksek başarım oranına ulaşmıştır. Yapılan ikinci deneysel çalışmada ise Daha Hızlı Bölge Tabanlı Evrişimli Sinir Ağları (Daha hızlı B-ESA) yöntemi kullanılarak, elle çizilmiş birçok devre üzerinde bulunan devre elemanlarının tespiti işlemi gerçekleştirilmiştir. Kullanılan yöntem ile farklı stillerde çizilen elle çizilmiş devrelerde bulunan devre elemanlarının tespit edilmesi işlemi düşük kayıp ve hızlı bir performans ile gerçekleşmiştir.

Anahtar Kelimeler: Elle çizilmiş devre, Derin Öğrenme, Evrişimli Sinir Ağları , Daha Hızlı Bölge Tabanlı Evrişimli Sinir Ağları

ABSTRACT

Master Thesis

CLASSIFICATION OF BASIC CIRCUIT COMPONENTS
BY IMAGE PROCESSING METHODS

MİHRİBAN GÜNAY

Inonu University
Graduate School of Nature and Applied Sciences
Department of Electrical and Electronic Engineering

68+xi pages

2021

Supervisor: Assist. Prof. Dr. Murat KÖSEOĞLU

Electrical and Electronics Engineers can easily recognize and distinguish circuit components when they see circuits drawn by hand on paper. However, it is difficult for computers and machines to classify hand drawn circuit components and to detect circuit components on hand drawn circuits. For this purpose, there are some studies in the literature on the classification and recognition of hand drawn circuit components using different methods.

In this thesis, two different experimental studies have been conducted to classify hand drawn circuit components and to detect them on the circuit. In the first experimental study, four different Convolutional Neural Networks (CNN) models were created and hand drawn circuit components were classified. And the performances of four different CNN models have been compared. As a result of the experimental study, the method used has reached a high success rate in the classification of hand drawn circuit components. In the second experimental study, by using the CNN-based Faster Region Based Convolutional Neural Networks (Faster R-CNN) method, hand drawn circuit components were detected on many hand drawn circuits. The Faster R-CNN method, on the other hand, performed the detection of circuit components in hand drawn circuits drawn in different styles, with low loss and fast performance.

Keywords: Hand drawn circuit, Deep Learning, Convolutional Neural Networks, Faster Region Based Convolutional Neural Networks

1. GİRİŞ

1.1 Genel Bilgiler

Her bir bilim dalının kendine özgü terimleri ve bu terimlerin görsel olarak ifade edildiği özel sembolleri olabilir. Özellikle uygulamalı bilim dallarında terimlerin gerçek anlamda karşılık geldiği görsellere ihtiyaç duyulmaktadır. Terimlerin şekilleri çizimler ile tarif edilir. Birçok mühendislik alanında çizimlerin yaygın olması görselliğin ön planda olduğunu göstermektedir. Elektrik Elektronik Mühendisliğinde de çizimler ve görsellik önemlidir. Çünkü Elektrik Elektronik Mühendisliğinde, bir devre üzerinde çeşitli analizler yapılmak istendiğinde öncelikle çizilmiş bir devre şeması gereklidir. Bu devre şeması, el çizimi veya herhangi bir çizim programı vasıtasıyla oluşturulmuş olabilir. El çizimi her ne kadar ilkel bir yöntem olarak görülse de genellikle ilk tercih edilen yöntemdir. Elektrik Elektronik Mühendisliğinde bir devre tasarlanmak istendiğinde öncelikle devrede kullanılacak devre elemanlarına karar verilir. Daha sonra belirlenen devre elemanları kullanılarak kağıt üzerine elle taslak bir çizim yapılır ve her bir elemana sayısal değerler verilir.

Elektrik Elektronik Mühendisliği uygulamalı bir bilim olduğu için, tamamlanan taslak çizim üzerinde matematiksel hesaplamalar yapılır. Sonuçlar hedeflendiği gibiyse devre simülasyon programında veya gerçek anlamda laboratuvar ortamında kurulur. Bir devrenin oluşturulmasında ilk başvurulan yol olan kağıt üzerine yapılan elle çizim devre tasarımının temelini oluşturmaktadır.

Bir devre kağıt üzerine elle çizilip uygun sonuçlar alındıktan sonra simülasyon programında test edilmek istendiğinde, bilgisayardaki simülasyon programında tekrar çizilmesi gerekir. Mühendislerin kağıt üzerinde çizilen bir devreyi bilgisayar üzerinde tekrar çizmeleri zaman kaybıdır. Kağıt üzerinde çizilen bir devrenin görüntüsünün alınıp yazılım aracılığıyla otomatik bir şekilde simülasyon programında çizilmesi, mühendisler için çok büyük avantajdır. Bu sayede mühendislerin hem iş yükü azalır hem de devreyi tekrar bilgisayar üzerinde çizmelerine gerek kalmaz. Böylece mühendisler zamandan da tasarruf etmiş olurlar. Böyle bir yazılımın oluşturulması, elle çizilmiş devre görüntülerinde bulunan devre elemanlarının bilgisayar tarafından tespitine ve tanınmasına bağlıdır.

Son zamanlarda, özellikle teknolojideki hızlı gelişmeler sayesinde, araştırmacılar insanların yapabildiği şeyleri bilgisayarlara yaptırmak için belirli girişimlerde

bulunmuşlardır. İnsanların yapmış olduğu yürümek, konuşmak ve düşünmek gibi eylemleri bilgisayarlar aracılığıyla robotlar üzerinde gerçekleştirmek, mühendislikte önemli bir çığır açmıştır. İnsanlar, daha önce gördüğü bir nesneyi birkaç kez daha gördüğünde tanıyıp ayırt edebilirler. İnsanların nesneleri tanıyabilmesi, bir nesneyi diğerinden ayırt edebilmesi, insanların sahip olduğu zeka sayesinde düşünebilmesinin sonucudur. Bilgisayarların da insanlar gibi düşünebilmesi, eğitilmesi, tanıyabilmesi, ayırt edebilmesi, sınıflandırabilmesi konularına artan yoğun ilgi sebebiyle yapay zeka ve makine öğrenmesi üzerine çalışan bilim insanı sayısı artmıştır. Dünya çapında, yapay zeka ve makine öğrenmesi ile çalışmalar yapan bilim insanı sayısında meydana gelen artış ve teknolojideki gelişmeler sebebiyle, her araştırmacı yeni bir fikir ortaya atmıştır. Ortaya atılan yeni fikirler ve yeni yaklaşımlar ile bilgisayarlar, insanların yapabildiği bir çok şeyi yapabilir hale gelmiştir.

Bu tez kapsamında, iki farklı deneysel çalışma yapılmıştır. Bu çalışmaların birincisinde, farklı kişilerin elle çizdiği direnç, indüktör, kapasitör ve voltaj kaynağı görüntülerinden bir veri seti oluşturulmuştur. 4 farklı sınıftan oluşan bu veri setini eğitmek ve sınıflandırmak için derin öğrenme mimarilerinden olan evrişimli sinir ağları mimarisi kullanılmıştır. Evrişimli Sinir Ağları (ESA) kavramının orjinal ve yaygın olarak kullanılan kısaltması CNN'dir. Tez çalışması kapsamında, ESA ve CNN eş anlamlı olarak kullanılmıştır. Tez kapsamında yapılan çalışmaların ilkinde, CNN mimarisi temelinde katman sayıları artırıp azaltılarak dört yeni CNN modeli oluşturulmuştur. Oluşturulan dört farklı CNN modeli, çalışma için özel oluşturulan veri seti ile eğitilmiştir. Tüm CNN modelleri ile 4 farklı devre elemanının sınıflandırılması işlemi gerçekleştirilmiştir. Bu modellerin eğitim ve doğrulama performansları değerlendirilmiştir. Bu çalışmaların ikincisinde; elle çizilmiş bir devre görüntüsünde bulunan devre elemanlarının tespiti için derin öğrenme tabanlı nesne tanıma için geliştirilmiş bir yöntem olan Daha Hızlı Bölge Tabanlı Evrişimli Sinir Ağları (Daha Hızlı B-ESA) yöntemi kullanılmıştır. Daha Hızlı B-ESA kavramının orjinal ve yaygın olarak kullanılan kısaltması Faster R-CNN'dir. Tez çalışması kapsamında, Daha Hızlı B-ESA ve Faster R-CNN eş anlamlı olarak kullanılmıştır. Tez kapsamında yapılan çalışmaların ikincisinde, elle çizilmiş devre görüntüleri toplanarak bir veri seti oluşturulmuştur. Veri setindeki etiketleme işlemleri tamamlandıktan sonra, önceden eğitilmiş Faster R-CNN modeli seçilmiştir. Eğitim işlemi seçilen model kullanılarak gerçekleşmiştir. Son olarak, elle çizilmiş test devreleri üzerinde modelin devre elemanlarını tespiti ve elde edilen doğruluk oranları değerlendirilmiştir.

1.2 Literatür Araştırması

Araştırmacılar bilim ve teknolojinin hızlı ilerleyişi ile son yıllarda popüler olan araştırma alanlarına yönelmişlerdir. Son yıllarda popüleritesi artan konulardan olan nesne tespiti ve sınıflandırma konularıyla ilgili literatürde yapılmış birçok çalışma mevcuttur. Araştırmacıların farklı bakış açıları ve gözlemleri sonucu hem nesne tespitinde hem sınıflandırmada farklı yöntemler kullanmaları literatüre katkı sağlamıştır.

Literatürde yapılmış olan el yazısı rakam tanıma ve el yazısı karakter tanıma çalışmalarında büyük başarılar alınması üzerine, bazı bilim insanları el çizimi devreleri ve devre elemanlarının tanınması ve sınıflandırılması konusuna yoğunlaşmışlardır. El çizimi devrelerin ve devre elemanlarının tespiti ve sınıflandırılması ile ilgili yapılan çalışmalarda, makine öğrenmesi tabanlı ve derin öğrenme tabanlı yöntemler kullanılmıştır.

Literatürde el çizimi devreler ile ilgili yapılan çalışmalar incelendiğinde; elle çizilmiş devre üzerinde bulunan elemanların tanınması aşamasında Yönlendirilmiş Gradyan Özelliklerinin Histogramı (HOG) yöntemi, devre elemanlarının sınıflandırılması aşamasında ise makine öğrenmesi tabanlı yöntemlerden Destek Vektör Makineleri (DVM) ve En Yakın K-Komşu (KNN) yöntemleri kullanılmıştır. Liu ve Xiao [1] çalışmalarında, el çizimi devreyi öncelikle topoloji tabanlı bir segmentasyon yöntemi kullanarak bölümlere ayırmışlardır, daha sonra devrede bulunan elemanları DVM ile sınıflandırmışlardır. Devrede sınıflandırılan her bir devre elemanı için sınıflandırma başarısı %90'ın üzerinde olmuştur. Moetesum ve diğ. [2], yapmış oldukları çalışmada; elle çizilmiş devre üzerindeki devre elemanlarını sınıflandırmak için öncelikle devreye segmentasyon ve çeşitli ön işlemler uygulamışlardır. Segmentasyon işleminden sonra devre elemanlarını tanımak için HOG yöntemini sınıflandırmak için ise DVM yöntemini kullanmayı önermişlerdir. Çalışmada 10 farklı elle çizilmiş devre elemanı sınıflandırılmış ve ortalama sınıflandırma başarım oranı %92 olmuştur. Angadi ve Naika [3], görüntü işleme teknikleri ile el çizimi devre şemaları üzerinde belirli ön işlemler gerçekleştirmişlerdir. El çizimi devre şeması tespitinde karar verme aşaması olan sınıflandırma aşamasında DVM yöntemi kullanmayı önermişlerdir. Dewangan ve Dhole [4], KNN metodu kullanarak 90 adet elle çizilmiş devrelerin taranmış görüntülerini kullanılarak yapmış oldukları sınıflandırmada %90 başarı oranı elde etmişlerdir. Naika ve diğ. [5] çalışmalarında, elle çizilmiş 10 farklı elektronik bileşenin tanınması aşamasında HOG yöntemini, sınıflandırma aşamasında ise DVM yöntemini kullanmışlardır. Çalışmada 2000 adet görüntüden oluşan veri seti kullanılmıştır. Yapılan

çalışma sonucunda, elektronik bileşenleri tanımadaki başarım oranı %96.9 olmuştur. Patare ve Joshi [6], çizilen dijital mantık devresinin görüntüsünü bölümlere ayırmak için bölge tabanlı bölümleme gibi görüntü işleme algoritmalarından yararlanmışlardır ve ardından her bölümdeki lojik kapıyı tanımak için DVM algoritmasını kullanmışlardır. Benzer şekilde, Datta ve diğ. [7], ilk olarak dijital devre görüntülerini segmentasyon işlemi ile bölümlere ayırıp daha sonra devrede bulunan elemanları sınıflandırmak için karar ağacı (KA) kullanmışlardır. Sala [8], elle çizilmiş devre şemalarında bulunan elektronik devre elemanlarının sembollerini sınıflandırmak için Gizli Markov Modelini (HMM) kullanmıştır. HMM yöntemi ile yapılan sınıflandırmanın sonucunda düşük hata oranı elde edildiği görülmüştür. Rabbani ve diğ. [9], elle çizilmiş bir devre görüntüsünden elektrik sembollerini doğrudan okuyabilen bir makine yapmak için Yapay Sinir Ağını (YSA) kullanan yeni bir yöntem önermişlerdir. Tanıma süreci iki adımdan oluşur. İlk adım şekil tabanlı özellikler kullanarak öznitelik çıkarımı, ikincisi ise geri yayılma algoritması aracılığıyla YSA kullanan bir sınıflandırma prosedürüdür. YSA, farklı kişiler tarafından elle çizilmiş elektrik devre görüntüleriyle eğitilmiş ve test edilmiştir. Çalışmanın sonuçları, kullanılan yöntemin uygulanabilir olduğunu ve iyi performans sergilediğini göstermiştir. Edwards ve Chandran [10], elle çizilmiş devre şemalarının taranan görüntülerine ön işlem uyguladıktan sonra, devre şemalarını belirli bölümlere ayırarak düğüm tanıma yöntemi ile 9 farklı elektronik bileşenin tanınması ve sınıflara atanması işlemini gerçekleştirmişlerdir. Çalışmada kulanılan yöntemin başarısının yaklaşık %86 olduğu rapor edilmiştir.

Son yıllarda derin öğrenme alanında yaşanan hızlı gelişmeler ile derin öğrenme tabanlı yöntemler kullanılarak birçok çalışma yapılmıştır. Literatürde derin öğrenme tabanlı yöntemlerden olan CNN genellikle görüntüleri sınıflandırma çalışmalarında kullanılmıştır[11]–[13]. Bu tezde yapılan deneysel çalışmaların ilkinde, elle çizilmiş devre elemanlarının sınıflandırılmasında CNN yöntemi kullanılmıştır. Literatürde, tezdeki ilk deneysel çalışmaya benzer şekilde 2020 yılının başlarında yapılmış bir çalışma bulunmaktadır. Wang ve diğ. [14] tarafından yapılan çalışmada, 3 farklı elle çizilmiş elektronik bileşeni sınıflandırmak için derin öğrenme tabanlı evrişimli sinir ağları yöntemi kullanılmıştır. Çalışmada kullanılan yöntemin elle çizilmiş diyot, direnç ve kapasitör görüntülerini sınıflandırmadaki başarı oranı %90 olarak belirtilmiştir. Son yıllarda, el çizimi imza tanıma [15], el çizimi Arapça kelime tanıma [16], el çizimi rakam tanıma [17], el çizimi Çince karakter tanıma [18] çalışmalarında CNN kullanılmıştır ve yüksek performanslar elde edilmiştir.

Görüntüler üzerinde bulunan nesneleri ve nesnelerin konumunu tespit amacıyla geliştirilen Faster R-CNN yöntemi derin öğrenme tabanlı yeni bir yöntemdir. Literatürde Faster R-CNN kullanarak nesne tespiti yapılmış ve yüksek performans elde etmiş birçok çalışma mevcuttur. Sardogan ve diğ. [19] yaptıkları çalışmada; hastalıklı ve sağlıklı elma yaprağı görüntüleri toplayarak bir veri seti oluşturmuşlardır. Faster R-CNN yöntemini kullanarak görüntüde yer alan hastalıklı ve sağlıklı yaprakları tespit edip sınıflandırmışlardır. Önerilen yöntem bu çalışmada ortalama %84.5 başarı elde etmiştir. Cömert ve diğ. [20] çalışmalarında, ilk olarak 1200 adet elma görüntüsü içeren bir veri seti hazırlamışlardır. Veri setinde bulunan görüntüler ile eğitim aşaması tamamlandıktan sonra, Faster R-CNN yöntemini kullanarak elma üzerinde bulunan çürük olan bölgeyi tespit etmişlerdir. Çalışma sonucunda ortalama %84.5 başarı elde etmişlerdir. Ren ve diğ. [21] çalışmalarında, Faster R-CNN yönteminde bazı değişiklikler ve ayarlamalar yaparak, optik uzaktan algılama görüntülerinde nesne tespiti amacıyla kullanmışlardır. Julca-Aguilar ve Hirata [22], el yazısı grafiklerde ve matematiksel ifadelerde bulunan sembollerin tespit edilmesinde Faster R-CNN yöntemi kullanmayı önermişlerdir. Yang ve diğ. [23], el yazısı metin tanıma çalışmalarında Faster R-CNN yöntemi kullanarak bir deneysel çalışma yapmışlardır. Çalışma sonucunda kullanılan yöntemin, karakter tanımadaki başarı oranının ortalama %97 olduğunu belirtmişlerdir. Albahli ve diğ. [24], 0'dan 9'a kadar elle çizilmiş rakamların otomatik olarak tanınması ve sınıflandırılması için Faster R-CNN yöntemini kullanmışlardır. Önerilen yöntem, aydınlatma koşulları, renklilik, rakamların şekli ve boyutundaki değişiklikler, bulanıklık ve gürültü efektlerinin oluşması vb. açıdan 0'dan 9'a kadar elle çizilmiş rakamların çeşitli görüntülerine sahip olan standart MNIST veri tabanında analiz edilmiştir. Çalışmada kullanılan Faster R-CNN yöntemi, giriş görüntüsündeki rakamları başarılı bir şekilde konumlandırıp, bunları 0'dan 9'a kadar tam sayı değerlerini temsil edecek şekilde 10 sınıfa ayırmıştır. Cao ve diğ. [25], küçük nesne tespiti için Faster R-CNN yöntemini kullanmışlardır. Çalışmanın deneysel sonuçlarında, kullanılan yöntemin doğruluk oranını %87 olarak belirtmişlerdir.

1.3 Tezin Kapsamı

Bu yüksek lisans tez çalışması 5 bölümden oluşmaktadır.

Bölüm 1' de, tez çalışmasıyla ilgili genel bilgiler, tez çalışmasının konusu üzerine yapılan literatür araştırması ve tezin kapsamı verilmektedir.

Bölüm 2' de, bilgisayarlı görme, görüntü işleme, görüntü sınıflandırma ve nesne tanıma ayrı başlıklar halinde incelenip açıklayıcı bilgiler verilmiştir. Ayrıca bu bölümde, görüntü sınıflandırma başlığı altında makine öğrenmesi ve klasik makine öğrenmesi yöntemlerinden DVM, KNN ve Karar Ağaçları (KA) yöntemleri incelenmiştir.

Bölüm 3'de, Derin Öğrenme, CNN, CNN katmanları ve bu katmanların çalışma mantığı örneklerle ayrı başlıklar halinde detaylı olarak incelenmiştir. Ayrıca bu bölümde, nesne tanımada bölge tabanlı CNN yöntemleri alt başklıklar halinde detaylı olarak açıklanmıştır.

Bölüm 4'de, tez çalışmasında yapılan iki farklı deneysel çalışma sunulmuştur. Deneysel çalışmaların birincisinde, kağıt üzerine elle çizilmiş 4 farklı devre elemanının, CNN kullanarak sınıflarını belirleme işlemi gerçekleştirilmiştir. Deneysel çalışmaların ikincisinde, CNN tabanlı nesne tanıma yöntemlerinden Faster R-CNN kullanılarak, kağıt üzerine elle çizilmiş farklı devreler üzerinde elle çizilmiş 4 farklı devre elemanının tespiti yapılmıştır.

Bölüm 5'de ise sonuç ve öneriler verilmiştir.

2. KURAMSAL TEMELLER

2.1 Bilgisayarlı Görü (BG)

Bilgisayarlı görü (BG), bilgisayarları görsel dünyayı yorumlamak ve anlamak için eğiten disiplinler arası bir alandır [26]. BG, farklı bir deyişle görüntülerden otomatik olarak bilgi alınmasıdır. BG, insan görme sisteminin yapabileceği görevleri otomatikleştirmeye çalışır. BG, insan görme sisteminin karmaşıklığının parçalarını kopyalamaya ve bilgisayarların görüntü ve videolardaki nesneleri tıpkı insanların yaptığı gibi tanımlayıp işlemesini sağlamaya odaklanan bilgisayar bilimi alanıdır [27].

Yakın zamana kadar, BG yalnızca sınırlı kapasitede çalışıyordu. 1990 yılında internet kullanımının yaygınlaşmasıyla birlikte internet ortamında paylaşılan veri miktarında artış yaşanmıştır. Dünyada her gün milyonlarca insan çevrimiçi olarak kendi fotoğraflarını paylaşmaktadır. Çevrimiçi erişilen bu görüntüler için yüz tanıma programları geliştirilerek, makinelerin fotoğraflarda ve videolarda bulunan kişilerin yüzünü tanıması sağlanmıştır. BG, görüntüler üzerinde farklı görevler üstlenir. Bu görevler; görüntü sınıflandırma, nesne algılama, anlamsal bölümleme ve örnek bölümlemedir [28]. Bu tez çalışmasının ilk aşamasında elle çizilmiş devre elemanlarının görüntüleri sınıflandırılmıştır. İkinci aşamada ise yeni nesil nesne tanıma algoritması kullanılarak elle çizilmiş devre görüntüleri üzerinde bulunan devre elemanlarının tespiti ve sınıflandırılması gerçekleştirilmiştir.

2.2 Görüntü İşleme

Görüntüde bulunan her bir renk tonu piksel adı verilen bir sayı ile temsil edilir. Bilgisayarlar, görüntüleri birçok farklı piksel değerinden oluşan bir matris olarak görür. Görüntü işleme temel olarak, bir görüntünün bilgisayar ortamına aktarılıp görüntü üzerinde teknolojinin gelişmesine bağlı olarak çeşitli işlemler gerçekleştirmeyi amaçlamaktadır [29]. Görüntü işlemede BG önemli bir konudur [30]. Konuyla ilgili araştırmalar uzun yıllara dayanmaktadır. Makine öğrenmesi, 20. yüzyılın ikinci yarısından beri bir çalışma alanı olarak var olmasına rağmen, çok uzun bir süre boyunca görüntü işlemede kullanılmamıştır.

Görüntüler üzerinde yapılacak tüm işlemler görüntü işleme alanına dahil edilebilir. Görüntü işleme en çok görüntü iyileştirme, görüntü restorasyonu, görüntü segmentasyonu, görüntü sınıflandırma ve nesne tanıma alanlarında kullanılmaktadır. Bu tezde, veri setinde bulunan tüm görüntülerin boyutlarının küçültülmesi ve aynı boyuta getirilmesi aşamasında

görüntü işleme komutları kullanılmıştır. Ayrıca, tezde yapılan ilk deneysel çalışmada kullanılan veri setinde bulunan tüm görüntüler renkli olduğu için griye dönüştürülmüştür. Veri setindeki tüm renkli görüntülerin griye çevrilmesinde görüntü işleme komutu kullanılmıştır.

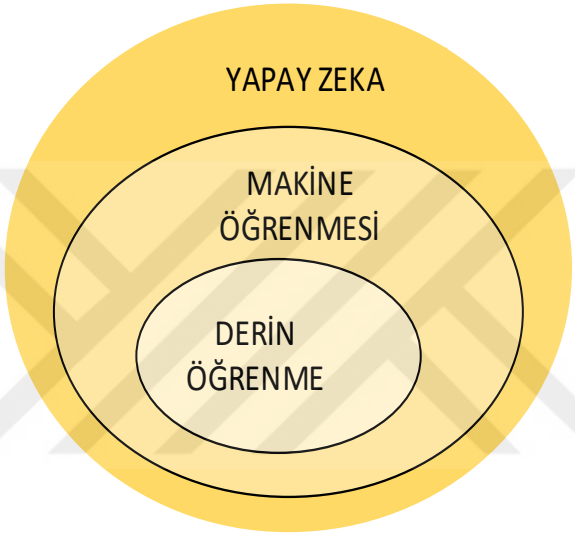
2.3 Görüntü Sınıflandırma

Doğada bulunan canlı ve cansız varlıkların bir ismi veya ait oldukları bir kategori vardır. Canlı ve cansız varlıkların insan gözüyle farkedilir biçimde çeşitli görüntüleri bulunabilir. İnsan gözü, canlı ve cansız varlıkların çeşitli görüntülerini kolaylıkla ayırt edebilir. Sınıflandırma, genellikle basit olarak farklı kategorilerdeki nesnelerin ait oldukları kategorilere yerleştirilmesi olarak ifade edilebilir. Sınıflandırma, bir görüntü içindeki pikselleri etiketlemek yerine tüm görüntüleri etiketlemeyi amaçlar [31]. Buradan yola çıkarak, görüntü sınıflandırma günlük hayatta insanlar tarafından saniyeler içerisinde ve basit bir şekilde yapılan bir aktivitedir. Bir insana, kedi, insan, ve çiçek resimlerinden oluşan 3 farklı kategorideki resimler gösterildiğinde ve bunları sınıflara ayırması istendiğinde kolayca sınıflara ayırabilir. İnsan gözüyle yapılan ayırt etme işlemi olan bu görüntü sınıflandırma işlemi, kolay bir şekilde yapılır. Fakat bilgisayarlar tarafından bu görüntü sınıflandırma işleminin gerçekleşmesi o kadar kolay değildir. Görüntü sınıflandırma, belirli yöntemler ve kodlamalar sayesinde gerçekleşir. Şekil 2.1, sınıflandırma aşamasını blok diyagramlar ile göstermektedir. Sınıflandırma işlemlerinde genellikle makine öğrenmesi yöntemleri ve derin öğrenme yöntemleri kullanılır. Tez çalışması kapsamında yapılan ilk deneysel çalışmada kullanılan görüntü verilerinin hangi sınıfa ait olduğu bilindiğinden, makine öğrenmesi yöntemlerinden olan danışmanlı öğrenme yöntemleri incelenmiştir. Makine öğrenmesi yöntemlerinin avantajları ve dezavantajları incelenmiştir. Makine öğrenmesi yöntemlerinin avantajları ve dezavantajları göz önüne alındığında tez çalışması kapsamında, görüntü sınıflandırma yaparken derin öğrenme yöntemi olan CNN tercih edilmiştir.

Şekil 2.1 : Sınıflandırma aşamasının genel gösterimi.

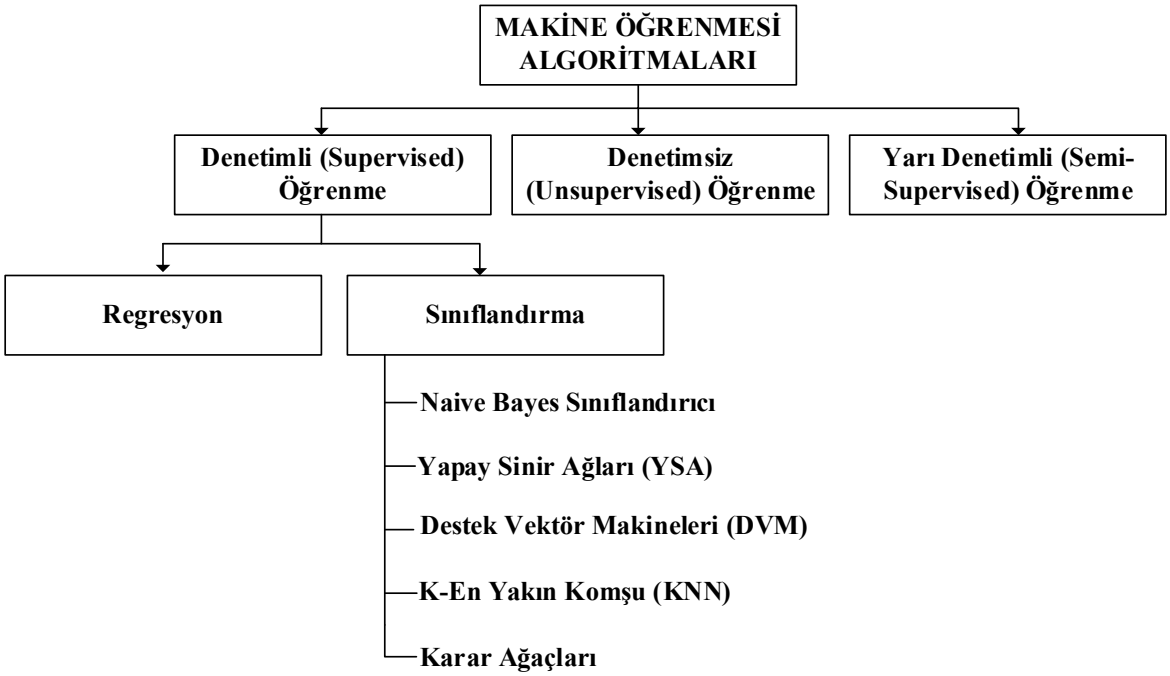
2.4 Makine Öğrenmesi

1959 yılında Arthur Samuel makine öğrenmesini "bilgisayarlara açıkça programlanmadan öğrenme yeteneği veren bilim alanı" şeklinde tanımlamıştır [32]. Daha basit bir ifadeyle makine öğrenmesi, algoritmaları ilgili verilerle besleyerek, bilgisayarlara insanlar gibi öğrenme ve davranma yeteneği kazandıran algoritmaları içeren bir bilim alanı olarak tanımlanabilir. Bir makine öğrenimi algoritmasının temel amacı, sağlanan eğitim verilerinden özellikleri öğrenmek ve eğitim için kullanılan verilerin ötesinde genelleme yapmaktır [33]. Makine Öğrenmesi, yapay zekanın bir alt kümesi olarak görülmektedir (Şekil 2.2) [34]. Makine öğrenmesi algoritmaları, istenmeyen e-posta filtreleme ve BG gibi çok çeşitli uygulamalarda kullanılır [35].



Şekil 2.2 : Derin öğrenme, makine öğrenmesi ve yapay zeka konularının kümesi [36].

Makine öğrenmesi algoritmaları, yaklaşımlarına, verilerin türüne ve çözmeyi amaçladıkları görev veya sorunun türüne göre farklılık gösterir. Makine öğrenmesi yöntemleri genel olarak denetimli, denetimsiz ve yarı denetimli öğrenme olarak üç kategoride incelenir. Denetimli öğrenme ve denetimsiz öğrenme arasındaki temel fark denetimli öğrenmenin, algoritmaları eğitmek için etiketli verilere ihtiyaç duyması, denetimsiz öğrenmenin ise etiketlere sahip verileri gerektirmemesidir [37,38]. Yarı denetimli öğrenmede ise algoritmaları eğitmek için etiketli ve etiketlenmemiş veriler kullanılır. Şekil 2.3'de makine öğrenmesi yöntemleri gösterilmektedir.



Şekil 2.3 : Makine öğrenmesi yöntemleri [39].

Denetimli öğrenme, verileri sınıflandırmak veya sonuçları doğru bir şekilde tahmin etmek için algoritmaları eğitirken etiketli veriler kullanır. Bu öğrenme modelinde, eğitim veri setinin ne olduğu ve istenilen çıktının hangi sınıf olması gerektiği bilinmektedir. Denetimli öğrenme modelinin amacı, her bir yeni giriş verisi için doğru etiketi tahmin etmektir. En temel haliyle, bir denetimli öğrenme algoritması basitçe Denklem 2.1'deki gibi ifade edilir.

$$Y = f(X) \tag{2.1}$$

Denklem 2.1'deki X giriş verisi, Y tahmin edilen çıkış değeri, f ise eşleşme fonksiyonunu ifade etmektedir. Denetimli öğrenme, sınıflandırma ve regresyon olmak üzere iki alt kategoriye ayrılır. Sınıflandırma ve regresyon, etiketli veriler üzerinde çalışır. Sınıflandırma, verileri birden fazla kategorik sınıfa, yani ayrık değerlere ayırmaya yarayan bir süreçtir. Regresyon, sınıfları ve ayrık değerleri kullanmak yerine verileri sürekli değerlere ayırmak için bir fonksiyon bulma sürecidir [40]. Denetimli öğrenme, sınıflandırma, yüz tanıma, satış tahmini ve istenmeyen e-posta sınıflandırma alanlarında kullanılır [37].

Denetimsiz öğrenme, veri seti ve Y çıkış değerinin bilinmediği sadece X giriş verisinin bulunduğu durumlarda kullanılan öğrenme yöntemidir. Bu öğrenme yönteminde, denetimli öğrenmeden farklı olarak çıkış değerinin ne olacağını tahmin etmek için veri

eğitilmemektedir [39]. Denetimsiz öğrenmenin amacı, veri eğitimi olmadan öğrenme modeli giriş olarak verilen verilerden öğrenmektir. Hangi sınıfa ait olduğu bilinmeyen iki farklı kategoriye ait veri verildiğinde, denetimsiz öğrenme bu iki kategorideki verileri benzerlik, farklılık, yakınlık gibi özelliklerine göre iki ayrı sınıf olarak ayırt edebilir.

Yarı denetimli öğrenme, denetimli ve denetimsiz öğrenmenin arasında yer alır. Bilgisayarların, hem etiketli hem de etiketsiz verilerin varlığında nasıl öğrendiğinin incelendiği bir öğrenme yöntemidir. Yarı denetimli öğrenmenin amacı, etiketli ve etiketsiz verilerin birleştirilmesinin öğrenme davranışını nasıl değiştirebileceğini anlamak ve böyle bir kombinasyondan yararlanan algoritmalar tasarlamaktır. Yarı denetimli öğrenme, makine öğrenimi ve veri madenciliğinde büyük ilgi görmektedir. Çünkü etiketli verilerin hem sayısı az hem de maliyeti yüksektir. Bunun için yarı denetimli öğrenme, denetimli öğrenme görevlerini iyileştirmek için hazır etiketsiz veriler kullanabilir. Yarı denetimli öğrenme, yeterli etiketli verinin bulunmadığı veya verileri etiketleyecek kaynağın olmadığı durumlarda kullanılır [41].

2.4.1 Bayes sınıflandırıcı

Bayes sınıflandırıcı yöntemine ismini veren 17. Yüzyılda yaşamış olan Thomas Bayes'dir [42]. Bayes teoreminin temelini koşullu olasılık hesaplamaları oluşturur. Bayes sınıflandırma, spam filtreleme, metin sınıflandırma, çok sınıflı tahmin, gerçek zamanlı tahmin gibi alanlarda kullanılır. Bayes teoremi, koşullu olasılıkları hesaplamada kullanılır [39]. Bayes teoreminde, değişkenler birbirinden bağımsızdır. Bayes teoremi formülü Denklem 2.2'de verilmiştir [43].

$$P(X/Y) = \frac{P(Y/X) * P(X)}{P(Y)} \tag{2.2}$$

P(X): X olayının gerçekleşme olasılığı

P(Y): Y olayının gerçekleşme olasılığı

P(X\Y): Y olayı gerçekleştiğinde X olayının gerçekleşme olasılığı

P(Y\X): X olayı gerçekleştiğinde Y olayının gerçekleşme olasılığı

Bayes ile sınıflandırma yapmak için ilk önce veri setinde bulunan veriler frekans tablosuna dönüştürülür ve değişkenlerin olasılıkları hesaplanır. Daha sonra veri setindeki her

bir sınıf için olasılık hesaplanır. En yüksek olasılık değerine sahip olan sınıf, tahmin sonucudur [44].

Görüntü sınıflandırmada Bayes Sınıflandırıcı kullanmanın avantajları ve dezavantajları [44]**:**

Avantajları:

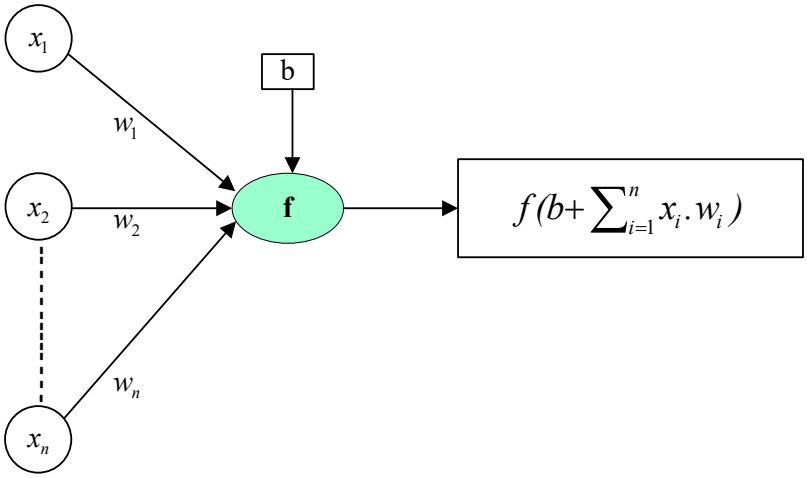
- Basit ve kullanımının kolay olması
- Veri sayısı az olsa bile sınıflandırma performansının yüksek olması
- Gerçek zamanlı sistemlerde hızlı performans sergilemesi

Dezavantajları:

- Gerçek hayattaki verilerde değişkenlerin birbirine bağımlı olması.

2.4.2 Yapay sinir ağları (YSA)

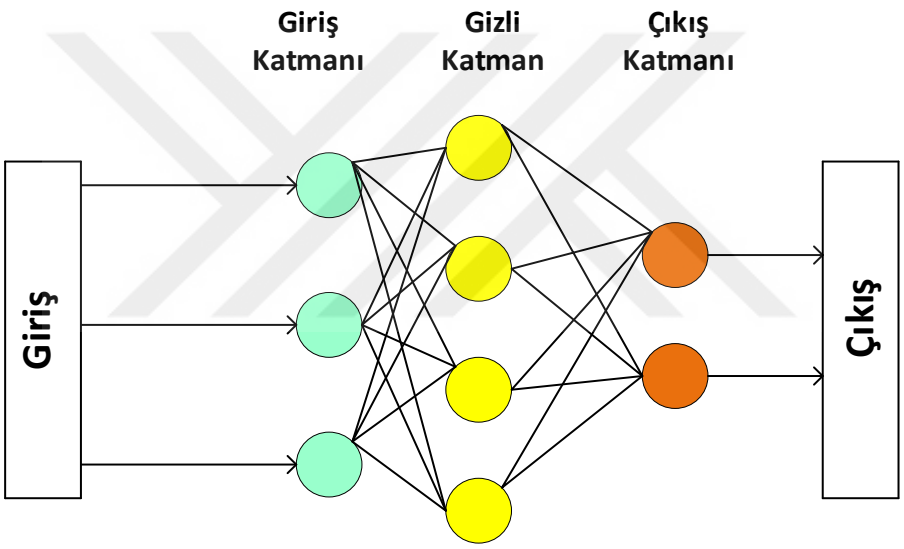
Beynin geleneksel dijital bilgisayarlardan farklı şekilde hesaplama yaptığının farkına varılmasıyla birlikte, Yapay Sinir Ağları (YSA) oluşturulması çalışmaları başlamıştır. Geçmişten günümüze YSA'lar birçok karmaşık problemde kullanılmıştır ve etkili bir makine öğrenmesi olarak yaygın kabul görmüştür. YSA algoritması, nöron aktivitesi yardımıyla insan beyni ile ilişkilendirilir veya karşılaştırılır. Bu algoritma matematiksel bir model veya hesaplama modelidir. YSA'lar doğrusal olmayan istatistiksel veri modelleme araçları olarak kabul edilir ve buna ek olarak biyolojik sinir ağlarının yapısını ve işlevsel yönlerini simüle etmeye çalışırlar. Bir sinir ağı, nöron adı verilen birimlerden oluşan bir mimaridir. Bir nöronun biyolojik yapısında bulunan dendritler, sinyali hücre gövdesine getirir ve ardından alınan tüm sinyaller toplanır. Toplam değer bir eşik değerinden büyükse, nöron sinyalleri iletir. Yani sinyaller hücre gövdesinden bir sonrakine taşınır.



Şekil 2.4 : Bir nöronun çalışmasını gösteren şema [45].

Şekil 2.4'de yapay bir nöronun yapısı ve çalışması gösterilmiştir. Şekil 2.4 incelendiğinde, giriş değerleri $x_1, x_2, ..., x_n$ olarak verilmiştir. Verilen her bir giriş değerine karşılık gelen ağırlıklar $w_1, w_2, ..., w_n$ olarak verilmiştir. Tahmin değeri, b ile ifade edilmiştir. Aktivasyon fonksiyonu ise f ile gösterilmiştir. Son olarak, nöronun çıkış değerinin hesaplanışı şekil üzerinde bir formül ile gösterilmiştir [46].

YSA mimarileri genellikle üç farklı katmandan oluşur: giriş özelliği vektörünü içeren giriş katmanı, sinir ağı yanıtından oluşan çıktı katmanı ve gizli katman. Gizli katman, hem girdi hem de çıktıya bağlanan nöronları içerir. Şekil 2.5'de genel bir sinir ağı mimarisi gösterilmiştir. İleri beslemeli sinir ağı olarak adlandırılan bu yapay sinir ağı mimarisi, yalnızca sinyallerin girişten çıkışa geçmesine izin verir. Ağ mimarisi ve işlevleri ilk aşamada seçilir ve eğitim sırasında aynı kalır. Sinir ağının performansı, ağırlıkların değerine bağlıdır. Ağırlıklar eğitim sırasında ayarlanır, böylece belirli bir çıktı elde edilir.



Şekil 2.5 : İleri beslemeli yapay sinir ağı modeli [47].

Giriş katmanı: Giriş bilgilerinin sonraki katmanlara aktarılmasından sorumludur [48].

Gizli katman: Gerekli tüm hesaplamalar bu katmanda yapılır ve ardından sinyaller veya bilgiler sonraki katmanlara aktarılır [48].

Çıkış katmanı: Çıkış katmanında, alınan sinyalleri veya bilgileri istenen formatta dönüştürecek bir aktivasyon işlevi kullanılır. Bir aktivasyon fonksiyonu genellikle bir dizi giriş üzerindeki belirli bir düğümün bir çıktısını tanımlamak için kullanılır [48].

Görüntü sınıflandırmada YSA kullanmanın avantajları ve dezavantajları [49,50]**:**

Avantajları:

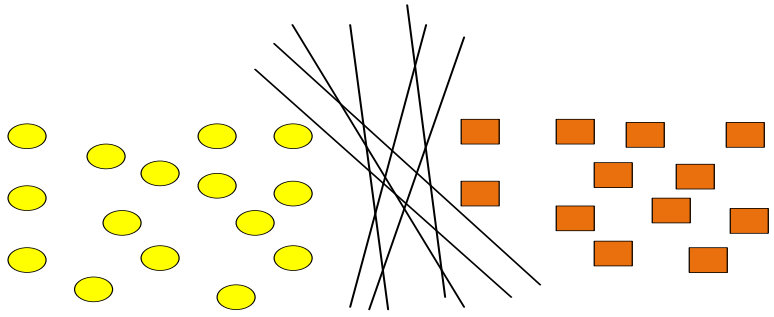
- Görüntüler gibi yüksek boyutlu özelliklere sahip verileri işlemede başarı.

Dezavantajları:

- Doğru gizli katman boyutunu ayarlamanın zor olması
- Nöron sayısı doğru bir şekilde belirlenmediğinde başka örneklere iyi bir şekilde genellenememesi
- Düğüm sayısı çok fazla olduğunda aşırı uyum sorununun ortaya çıkması
- Hesaplama maliyetlerinin fazla olmasından dolayı, işlemci gücü yüksek bilgisayarlara ihtiyaç olması ve bellek kullanımının fazla olması
- YSA algoritmalarının anlaşılmasının zor olması

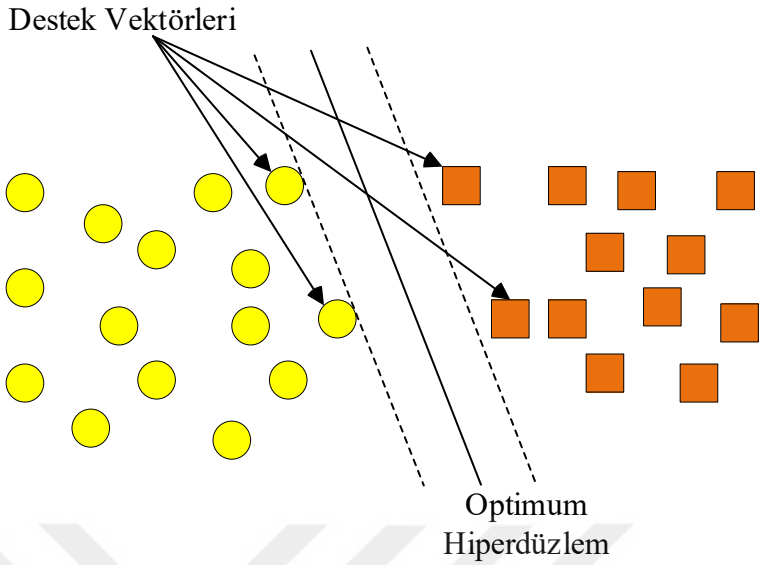
2.4.3 Destek vektör makinesi (DVM)

Destek vektör makinesi (DVM), el yazısı görüntülerinin sınıflandırılmasında yaygın olarak kullanılan bir yöntemdir. Doğrusal bir yöntem olan DVM, iki kategoriden oluşan sınıflandırmalarda yüksek performans elde eden bir sınıflandırma metodudur [51]. DVM, ilk zamanlar sadece iki sınıflı sınıflandırma problemlerinin çözümünde kullanılmak üzere geliştirilmiş makine öğrenmesi algoritmasıdır. İlerleyen yıllarda bu algoritma, zamanla geliştirilerek çok sınıflı problemlerin çözümünde kullanılabilir hale gelmiştir. DVM algoritması ilk gelişmeye başladığı yıllarda, doğrusal veri setleri üzerinde kullanılmıştır. Daha sonra bilim insanlarının katkılarıyla büyük ilerlemeler kaydedilmiştir ve bununla birlikte doğrusal olmayan veri setleri üzerinde de kullanılmaya başlanmıştır. DVM ile sınıflandırmanın temeli, iki sınıfı birbirinden ayırabilen uygun bir hiperdüzlem bulma işlemidir [52]. Şekil 2.6'da görüldüğü üzere sınıf sayısı iki olan veriler, birden fazla hiperdüzlem ile ayırt edilebilir.



Şekil 2.6 : Doğrusal ayrılabilen iki sınıflı sınıflandırma problemi [53].

DVM, çizilen hiperdüzlemler içerisinden en uygun hiperdüzlem olarak bilinen optimum hiperdüzlemi bulmayı amaçlar. İki sınıflı sınıflandırmada optimum hiperdüzlem, Şekil 2.7'de görülen her iki sınıfa ait olan mesafeyi maksimuma çıkaran düzlemdir. İki sınır arasındaki genişliği sınırlandıran noktalara, destek vektörleri denir [53].



Şekil 2.7 : Doğrusal ayrılabilen iki sınıflı sınıflandırma probleminde optimum hiperdüzlem ve destek vektörleri [53].

Doğrusal olarak ayrılabilen iki sınıflı verileri ayırt etmek için kullanılacak DVM'de bir karar fonksiyonu kullanılır. Sınıflar $\{-1,+1\}$ etiketleri ile ifade edilirler. M adet örneğin bulunduğu varsayılırsa, $i=1,\ldots,M$ için eğitim verileri $\{x_i,y_i\}$ şeklinde ifade edilir. Optimum hiperdüzlemi bulmak için gerekli Denklem 2.3 ve Denklem 2.4'de verilmiştir.

$$w.x_i+b\geq+1 \text{ her } y=+1 \text{ için} \qquad (2.3)$$

$$w.x_i+b\geq+1 \text{ her } y=+1 \text{ için} \qquad (2.4)$$

Eğitim verisinde bulunan tüm örneklerin verilen denklemlerdeki w ağırlık vektörü ve b eğilim değerini doğruluyorsa, doğrusal olarak ayırt edilebilir. Optimum hiperdüzlem belirlenirken $\|w\|$'nin minimum düzeyde olması gerekir. Bu sebeple, en uygun hiperdüzlemin belirlenmesi, Denklem 2.5'de verilen sınırlı optimizasyon probleminin çözümünü gerektirir [53].

$$\min\left[\frac{1}{2}\|w\|^2\right] \tag{2.5}$$

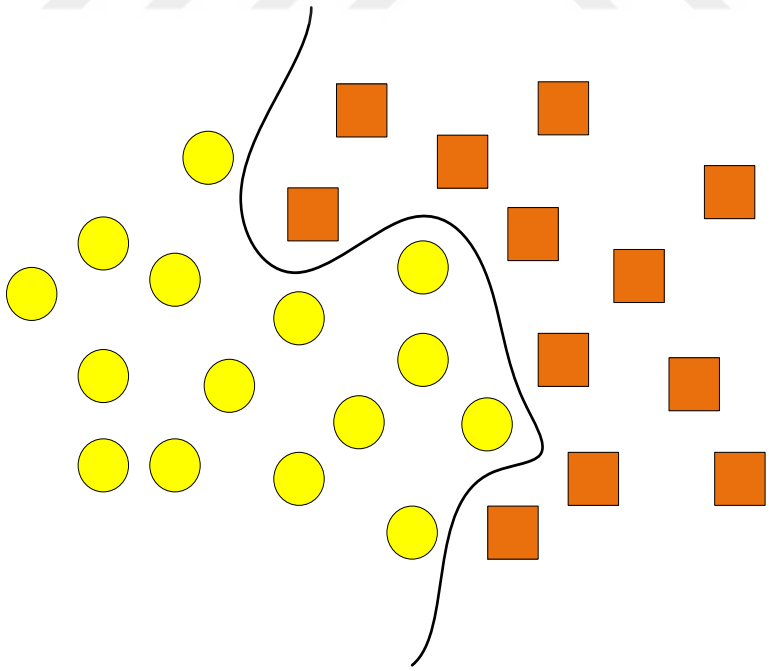
Denklem 2.5'de verilen optimizasyon problemi denklemi [53] ve Lagrange denklemleri kullanılarak, Denklem 2.6'da belirtilen yeni bir denklem elde edilir. Bu denklem;

$$L(\mathrm{w},\mathrm{b},\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{M}\alpha_i y_i\left(w.x_i + b\right) + \sum_{i=1}^{M}\alpha_i \tag{2.6}$$

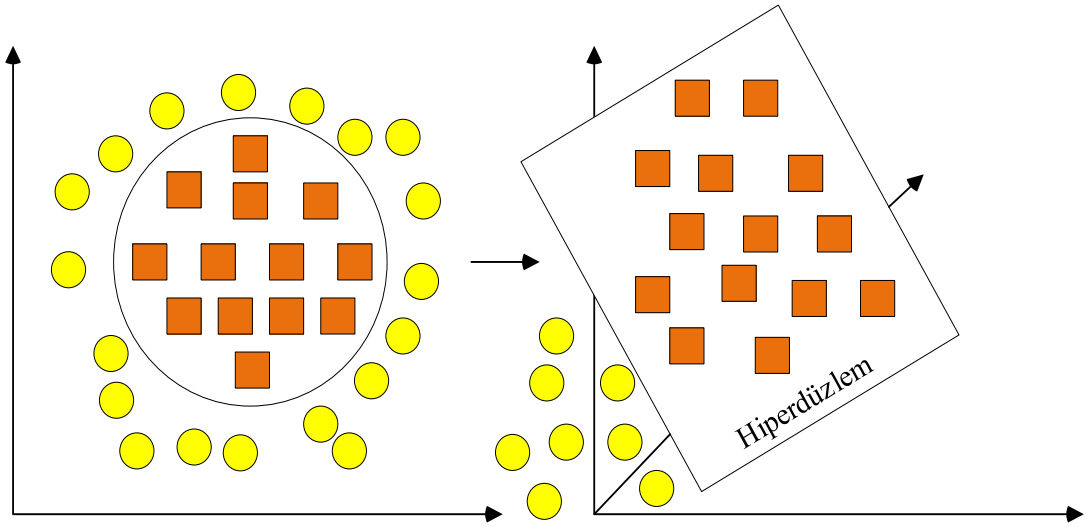
Bu denklemlerden yola çıkarak iki sınıfı birbirinden ayırt etmek için gereken karar fonksiyonu, aşağıdaki Denklem 2.7'deki gibi ifade edilir [54].

$$f\left(x\right) = sign\left(\sum_{i=1}^{M}\lambda_i\, \mathrm{y}_i(\mathrm{x}.\mathrm{x}_i) + \mathrm{b}\right) \tag{2.7}$$

Şekil 2.8'de görüleceği üzere eğitim veri setinde bulunan veriler her zaman doğrusal olarak ayrılamaz. Verilerin doğrusal bir şekilde ayrılamadığı durumlarda, verileri birbirinden ayırt edecek en uygun eğrinin bulunması gerekir. Bunun için veriler, orjinal girdi uzayından, Şekil 2.9'da görüldüğü gibi daha yüksek boyutlu özellik uzayına aktarılır.



Şekil 2.8 : Doğrusal ayrılamayan iki sınıflı veri seti [53].



Şekil 2.9 : Doğrusal ayrılamayan iki sınıflı verilerin özellik uzayına geçişi [53].

Doğrusal olarak ayrılamayan verilerin sınıflandırılması sürecinde, bu verilerin, matematiksel olarak $K\left(x_i, x_j\right) = \varphi\left(x\right).\varphi\left(x_j\right)$ şeklinde ifade edilen kernel fonksiyonu ile doğrusal olarak ayrılmasına olanak sağlanır [54].

Görüntü sınıflandırmada DVM kullanmanın avantajları ve dezavantajları [55]**:**

Avantajları:

- Eğitimin hızlı olması ve kısa zaman alması
- Sınıflandırma aşamasında, yüksek doğruluk elde etmesi
- Genişletilerek diğer sınıflandırma problemlerine uygulanabilir olması

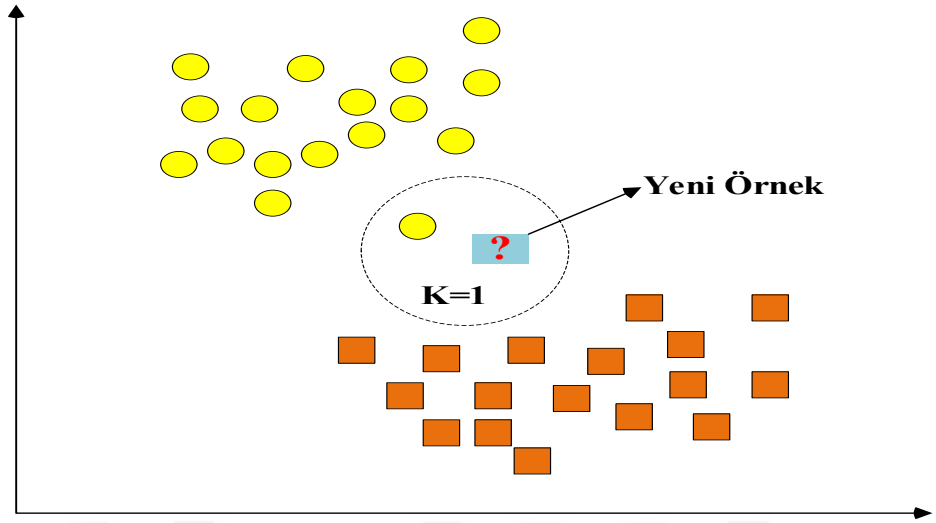
Dezavantajları:

- Büyük veri setlerinde yavaş çalışması ve eğitimin uzun sürmesi
- Kernel fonksiyonu seçiminin kolay olmaması
- Büyük veri setlerinde bilgisayarlarda büyük belleklere ihtiyaç duyması

2.4.4 K-en yakın komşu (KNN)

K-En Yakın Komşu (KNN) metodu, literatürde görüntü sınıflandırma çalışmalarında yaygın olarak kullanılan basit ve anlaşılır bir algoritmadır [56,57]. Eğitim veri kümesinde bulunan S adet sınıfa ait M adet eğitim verisi olduğu düşünüldüğünde, veri setinde bulunan M adet verinin hepsi ait oldukları sınıflara göre etiketlenir. Etiketlenmiş M adet eğitim verisi

içerisinde bulunmayan yeni bir örneğin hangi sınıfa ait olduğu bulunmak istendiğinde, sınıfı bilinmeyen örnek, eğitim veri setinde sınıflar halinde bulunan verilerden hangisine mesafe olarak yakınsa o sınıfa dahil edilir [58]. Şekil 2.10'da görüleceği üzere "?" ile gösterilen yeni bir örnek K=1'e göre hangi sınıfa mesafe olarak yakınsa o sınıfa dahil edilir.



Şekil 2.10 : Etiketlenmemiş yeni bir örneğin K parametresine göre sınıfının belirlenmesi [59].

Örnekler arasındaki mesafeye dayalı bu sınıflandırmada, iki örnek arasındaki mesafenin belirlenmesinde birçok yöntem bulunmasına ragmen çoğunlukla öklid uzaklık yöntemi kullanılır. Kullanıcı tarafından etiketlenmemiş yeni bir veri, mevcut veri setine dahil edilmek istendiğinde, etiketlenmemiş verinin K sayıda en yakın komşuları bulunur. K sayıda komşu veriler ile yeni veri arasındaki uzaklığın belirlenmesinde genellikle öklid uzaklık yöntemi kullanıılır. K parametresi komşu sayısını ifade etmektedir. İki nokta arasındaki mesafeyi bulmak için kullanılan öklid uzaklık yöntemi, Denklem 2.8 ile ifade edilmiştir [53].

$$d(i,j)=\sqrt{\sum_{k=i}^{N}\left(x_{ik}-x_{jk}\right)^{2}} \qquad (2.8)$$

Görüntü sınıflandırmada KNN kullanmanın avantajları ve dezavantajları [60]**:**

Avantajları:

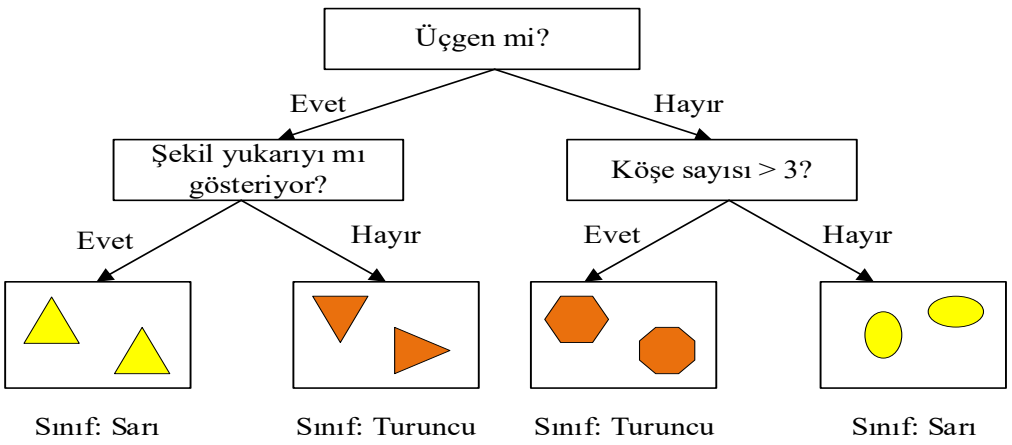
- Anlaşılılır ve kolay bir şekilde uygulanabilir olması
- Eğitim aşamasının kısa zaman alması
- Gürültülü eğitim verilerinde iyi performans sergilemesi

Dezavantajları:

- Her bir veri için uzaklık hesabının yapılması
- Eğitim aşamasında ezberleme yapması
- Uzaklık hesabı her bir veri için yapıldığından dolayı kullanılan bilgisayarların hafızasında fazla yer kaplar.
- Büyük veri setlerinde büyük hafızaya ihtiyaç duyar.

2.4.5 Karar ağaçları (KA)

Karar ağacı (KA), akış şemasına benzeyen bir ağaç yapısıdır [61]. KA algoritmaları, özellik değerlerine bakarak verileri kategorilere ayıran ağaçlardan oluşur. KA'nın en üstünde bulunan hücreye kök denir [62]. Kök hücresinin farklı dallar ile alt hücrelere ayrılmasıyla oluşan hücreler düğüm olarak adlandırılır. Bir karar ağacındaki her düğüm, sınıflandırılması gereken bir özelliği gösterir. KA'nın en altında bulunan düğümler yaprak olarak adlandırılır. Düğümler, kök hücresinden yaprak düğüme bir ağaç yapısı oluşturur, bu da başka bir deyişle yukarıdan aşağıya yaklaşım olarak adlandırılır. Şekil 2.11'de örnek bir KA verilmiştir [63]. Eğitilmiş veri setinde verilen belirli kriterler göz önünde bulundurularak verilen karar, Şekil 2.11'deki ağaç yapısıyla gösterilmiştir. Bu şekilde veriler, farklı özelliklere sahip farklı geometrik şekillerden oluşmaktadır. İstenen sınıf özelliği, geometrik şekillerin köşe sayılarına ve geometrik şekillerin yukarıyı gösterme durumlarına göre şekillerin rengidir.



Şekil 2.11 : Örnek KA yapısı [63].

Bazı KA algoritmaları kullanılarak, şekilleri renklerine göre sınıflandırmak için farklı şekil özellikleri üzerinde birkaç test belirlenmiş ve belirlenen testler ağacın en üst kısmından en alt kısmına ulaşılana kadar ardışık olarak uygulanmıştır. Ağacın en alt kısmında bulunan yaprak düğümleri sınıf etiketlerini içerir. Örnekte geometrik şekiller köşe

sayılarına göre kategorize edilebilir. Yukarıyı gösteren üçgen ve daha farklı pozisyonlarda bulunan üçgenler ve farklı köşe sayılarına sahip geometrik şekiller bulunmaktadır. Örnek KA incelendiğinde, yukarıyı gösteren her üçgenin sarı olacağı tahmin edilecektir. Ayrıca, köşe sayısının 3'den büyük olduğu her geometrik şeklin renginin turuncu olacağı tahmin edilecektir.

Görüntü sınıflandırmada KA kullanmanın avantajları ve dezavantajları [64]**:**

Avantajları:

- Anlaşılması basit
- Kullanımının kolay olması
- KA algoritmalarından olan C4.5 algoritmasının diğer makine öğrenmesi algoritmalarına göre sınıflandırma hızının yüksek, hata oranının düşük olması
- Gürültü içeren eğitim verilerinde iyi performans sergilemesi

Dezavantajları:

- Karar ağaçları büyüdükçe kararın güvenilirliği negatif yönde etkinebilir
- Sınıf sayısının fazla olduğu, veri sayısının az olduğu durumlarda sınıflandırma hataları fazla olabilir.

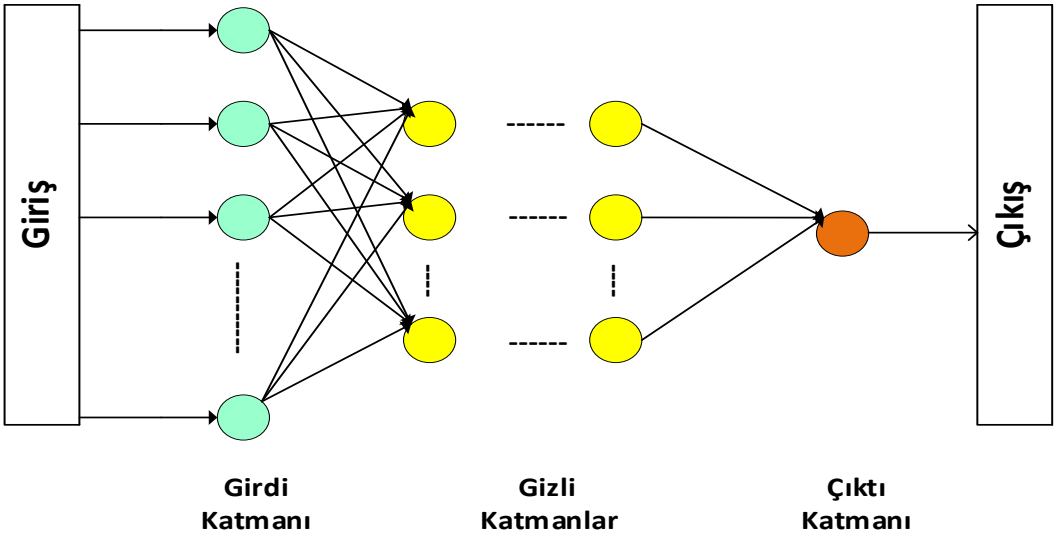
2.5 Nesne Tanıma

Görüntülerde nesne tespiti, BG'deki en temel ve zorlu sorunlardan biridir [65]. Nesne algılama, bir görüntüdeki birden çok nesneyi tanımlamanın, doğrulamanın ve bunları belirli sınıflara ayırmanın yanı sıra, nesnenin bulunduğu yeri bulma sorunuyla da ilgilenir. Bu tanımla, nesne tespitinin iki alt problemle ilgili olduğu sonucuna varılabilir: sınıflandırma ve konumlandırma. Sınıflandırma problemi, bir görüntüdeki bir nesnenin sınıfını tahmin etme sorunuyla ilgilidir. Konumlandırma problemi ise bir görüntüdeki bir veya birden fazla nesnenin bulunduğu konumu ve sınırları belirlemek için sınırlayıcı kutu çizme sorunuyla ilgilidir. İdeal bir sınırlayıcı kutu, bir nesnenin tüm parçalarını içeren bir eksen paralel dikdörtgendir. Bir görüntüde bulunan her bir nesne için çizilen sınırlayıcı kutu içinde bulunan nesnenin hangi sınıfa ait olduğu, olasılıkla ifade edilen güven puanı ile belirlenir. Nesne tanıma algoritması uygulanan görüntüde yer alan nesne veya nesnelerin etrafına, sınırlayıcı kutu çizilerek nesnenin hangi sınıfa ait olduğu bulunur.

3. DERİN ÖĞRENME

Derin öğrenme, bilgisayar sistemlerinin daha basit kavramları daha soyut ve karmaşık kavramlara dönüştürmesine olanak sağlayan, makine öğrenmesinin bir alt kümesi olarak düşünülebilir. Derin öğrenme, insan beyninin işleyişinden ve yapısından esinlenen algoritmalar geliştirmekle ilgilenir [66]. Derin öğrenme ilk olarak 1980'lerde kuramsallaştırılmış olsa da, son zamanlarda en çok rağbet gören ve hala üzerinde birçok kişinin çalıştığı bir araştırma ve uygulama alanıdır.

Derin öğrenmenin temeli yapay sinir ağlarına dayanmaktadır. Yapay sinir ağları; girdi katmanı, çıktı katmanı ve gizli katmandan oluşur. İleri beslemeli yapay sinir ağlarında gizli katman sayısı en fazla 3 olmaktadır. Klasik makine öğrenmesi teknikleriyle sınıflandırmada, özellik vektörü çıkarma aşamaları alanında uzman mühendis aracılığıyla gerçekleştirilirken, derin öğrenme yöntemlerinde özellik vektörü çıkarma aşamaları, işlemci gücü yüksek bilgisayarlar tarafından gerçekleştirilir [67]. Derin sinir ağı (DSA) , girdi ve çıktı katmanları arasında birden fazla katmana sahip yapay bir sinir ağıdır [68]. Girdi katmanı ile çıktı katmanı arasında bulunan gizli katman sayısının çok sayıda olması, derin öğrenme kavramının gelişmesine olanak tanımıştır. Şekil 3.1'de girdi katmanı, çıktı katmanı ve çok fazla gizli katmandan oluşan derin sinir ağı modeli gösterilmiştir. Gizli katman sayısı ne kadar artarsa ağ o kadar derinleşir ve daha fazla özellik çıkarımı yapılır.

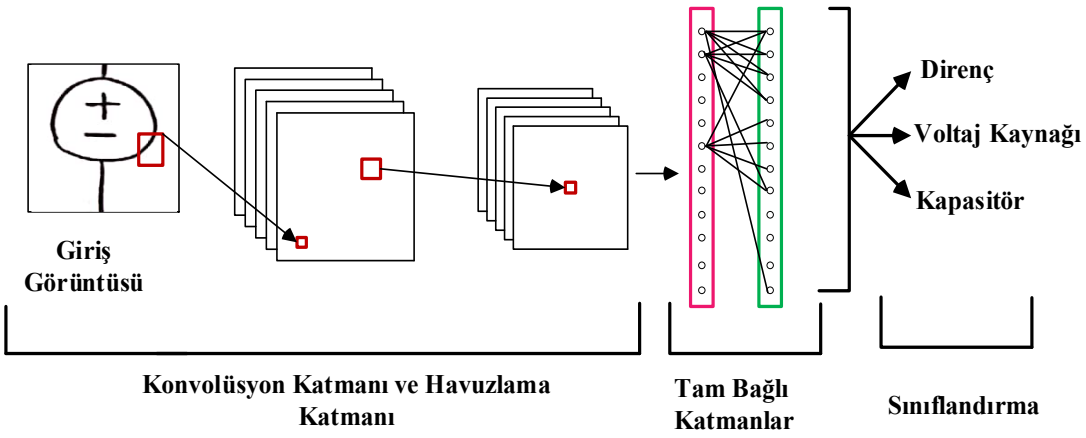


Şekil 3.1 : Derin sinir ağı (DSA) modeli [69].

3.1 Evrişimli Sinir Ağları (ESA)

Evrişimli Sinir Ağları (ESA) kavramının orjinal ve yaygın olarak kullanılan kısaltması CNN'dir. ESA ve CNN eş anlamlı olarak kullanılmıştır. CNN, 1980 yılında Kunihiko Fukushima tarafından ortaya atılmıştır [70]. CNN, aynı zamanda ConvNet olarak da adlandırılır. CNN, insan beynindeki biyolojik süreçlerden esinlenen sıradan sinir ağlarına benzer şekilde çalışan derin öğrenme algoritmasıdır. İlk CNN modeli, Yann LeCun'un elle yazılan rakamları tanımak için 1988 yılında oluşturduğu ve başarılı sonuçlar alınan LeNet mimarisidir [71]. 1990'lı yılların başında CNN konuşma tanıma için kullanılmıştır [72]. Günümüzde, CNN'ler el yazısı tanıma [73,74], yüz tanıma [75]–[77], davranış tanıma [78], konuşma tanıma [79], görüntü sınıflandırma [80,81] çalışmalarına başarıyla uygulanmıştır. CNN mimarisinde, parametre sayısını ve ağ içindeki hesaplamaları azaltmaya yardımcı bir katman kullanılmaktadır. Bundan dolayı, görüntü tanıma ve görüntü sınıflandırma gibi alanlarda CNN'in çok yararlı olduğu görülmüştür. CNN'de parametre sayısını ve ağ içindeki hesaplamaları azaltan katman, havuzlama katmanıdır [82]. CNN'ler çoğunlukla görüntü verileriyle ilgili yapılan çalışmalarda kullanıldığı için ve görüntüler de iki boyutlu veriler olduğu için, görüntülerin işlemci üzerindeki hesaplama maliyeti fazladır. Bunun için, ardışık havuzlama katmanı kullanılmaktadır.

1B-CNN, 2B-CNN ve 3B-CNN olmak üzere 3 farklı CNN mimarisi bulunmaktadır. 2B-CNN hemen hemen tüm BG görevlerinde kullanılır. 2B-CNN genellikle görüntü ve video sınıflandırma çalışmalarında kullanılır. CNN mimarileri oluşturulurken konvolüsyon katmanı, havuzlama katmanı, relu katmanı, düzleştirme katmanı, tam bağlı katman ve çeşitli aktivasyon fonksiyonları kullanılır. Aşağıdaki Şekil 3.2'de örnek olarak tipik bir CNN mimarisi gösterilmiştir. CNN'de bulunan her katmanın mimaride üstlendiği görev farklıdır.



Şekil 3.2 : CNN mimarisinin genel yapısı [83].

Görüntü sınıflandırmada CNN kullanmanın avantajları ve dezavantajları [84]**:**

Avantajları:

- Özellik çıkarma aşamasının, özellik mühendisliğine gerek kalmadan gerçekleşmesi
- Özellik çıkarma aşaması, CNN ile otomatik bir şekilde gerçekleştiği için hesaplama maliyeti azdır.
- DVM ve KNN metodlarına kıyasla, test aşamasının kısa sürmesi

Dezavantajları:

- Eğitimin büyük veri setlerinde uzun zaman alması
- Büyük veri setlerinde bilgisayarlarda büyük belleklere ihtiyaç duyması

3.1.1 Konvolüsyon katmanı

Konvolüsyon katmanında, matris biçiminde ifade edilen giriş verisi, seçilecek bir filtre matrisiyle konvolüsyon işlemine tabi tutulur. Filtre matrisindeki tüm sayıların, sırasıyla giriş verisindeki sayılarla belirli kurallara göre konvolüsyonu yapılır. Bu konvolüsyon işlemi sonucunda özellik haritasına ulaşılır.

3.1.1.1 Tek boyutlu konvolüsyon işlemi

Tek boyutlu konvolüsyon genellikle sinyaller ve doğal dil işleme alanında kullanılır. Tek boyutlu konvolüsyon işleminde, genellikle giriş verisi bir dizidir ve seçilen filtre giriş verisi üzerinde 1 boyut kaydırılır. Tek boyutlu konvolüsyon işleminin matematiksel ifadesi, Denklem 3.1'de verilmiştir [85].

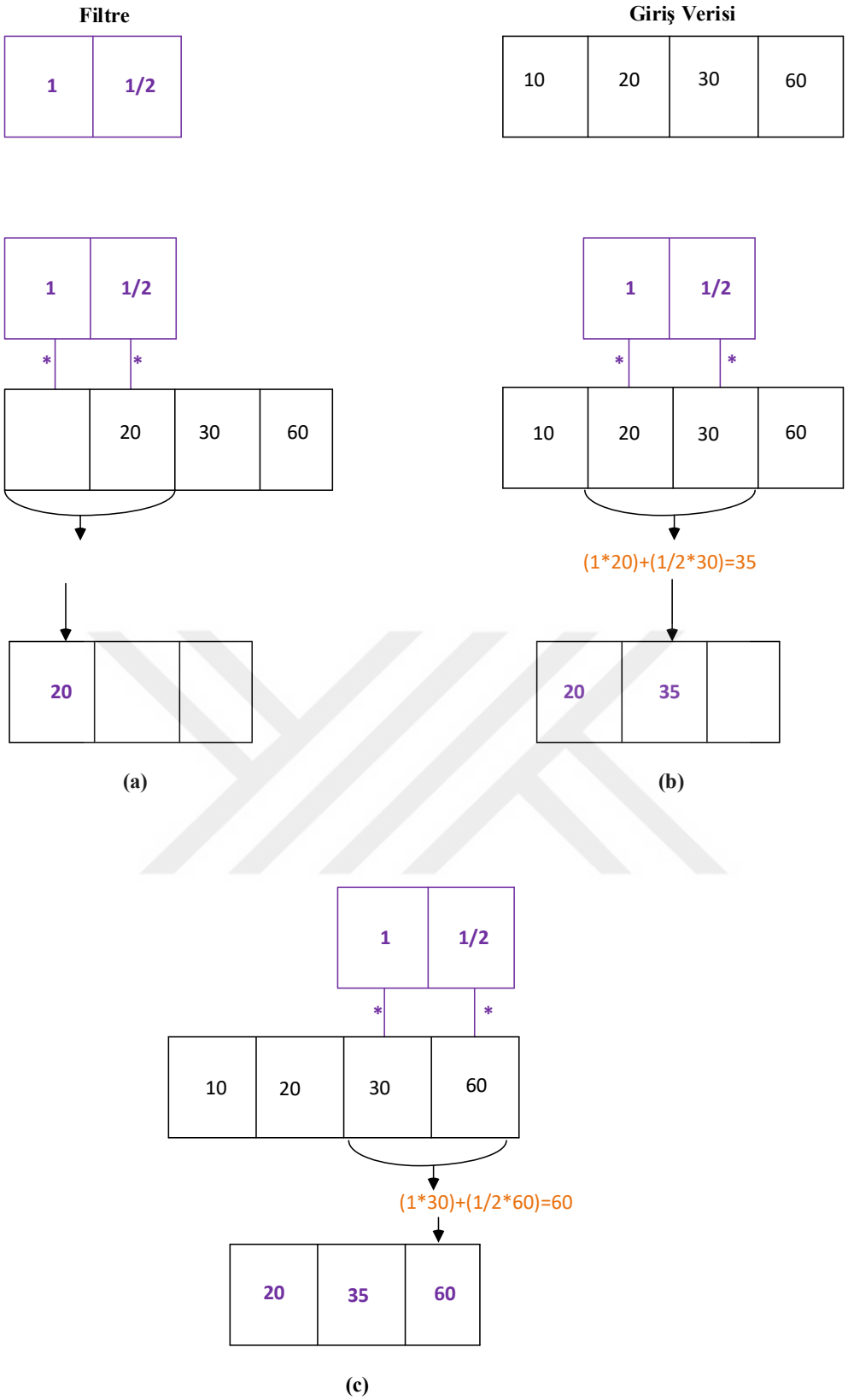
$$v_{ij}^{x} = f\left(b_{ij} + \sum_{m} \sum_{p=0}^{P_i - 1} w_{ijm}^{p} v_{(i-1)m}^{(x+p)}\right) \tag{3.1}$$

Denklem 3.1 incelendiğinde, i. katmandaki j. özellik haritasındaki (x) konumundaki değer v_{ij}^{x} olarak verilmiştir. b_{ij}, özellik haritası için bias değeridir. P_i, filtre boyutunu belirtir. p, filtre endeksidir. i, giriş katmanı dizinidir. m, özellik katmanı dizinidir. j, çıkış katmanı dizinidir. w, özellik haritasının sayısına ve boyut sayısına göre filtrenin değerini ifade eder.

Şekil 3.3 (a)'da gösterilen $1x2$ boyutunda seçilen filtrenin ilk elemanı, $1x4$ boyutunda verilen giriş verisinin ilk elemanıyla çarpılır. Daha sonra filtrenin ikinci elemanı giriş verisinin ikinci elemanıyla çarpılır. İlk çarpım sonucu, ikinci çarpım sonucuyla toplanır.

Toplam sonucu, özellik haritasının ilk elemanını oluşturur. Örnek olarak verilen giriş verisinde 4 adet eleman bulunmaktadır. Verinin ilk 2 elemanı filtrenin elemanlarıyla konvolüsyon işlemine tabi tutulmuştur ve geriye verinin 2 elemanı kalmıştır. Bunun için filtre 1 adım kaydırılır yani filtre 1 alt satıra kaydırılarak konvolüsyon işlemi yapılır. Verinin tüm elemanlarının filtredeki elemanlar ile konvolüsyon işleminin yapılması için sırayla filtre veri üzerinde kaydırılarak, Şekil 3.3 (b) ve Şekil 3.3 (c)'deki adımların gerçekleşmesi sonucu özellik haritası çıkarılır. Giriş verisinin boyutları WxH, filtre boyutu ise KxL olduğu düşünüldüğünde özellik haritasının boyutu Denklem 3.2'deki gibi formülize edilir [86].

$$(WxH) * (KxL) = (W - K + 1) * (H - L + 1) \tag{3.2}$$



Şekil 3.3 : Tek boyutlu konvolüsyon işleminin bloklarla gösterimi: (a) özellik haritasının ilk elemanının bulunması, (b) özellik haritasının ikinci elemanının bulunması, (c) özellik haritasının üçüncü elemanının bulunması.

3.1.1.2 İki boyutlu konvolüsyon işlemi

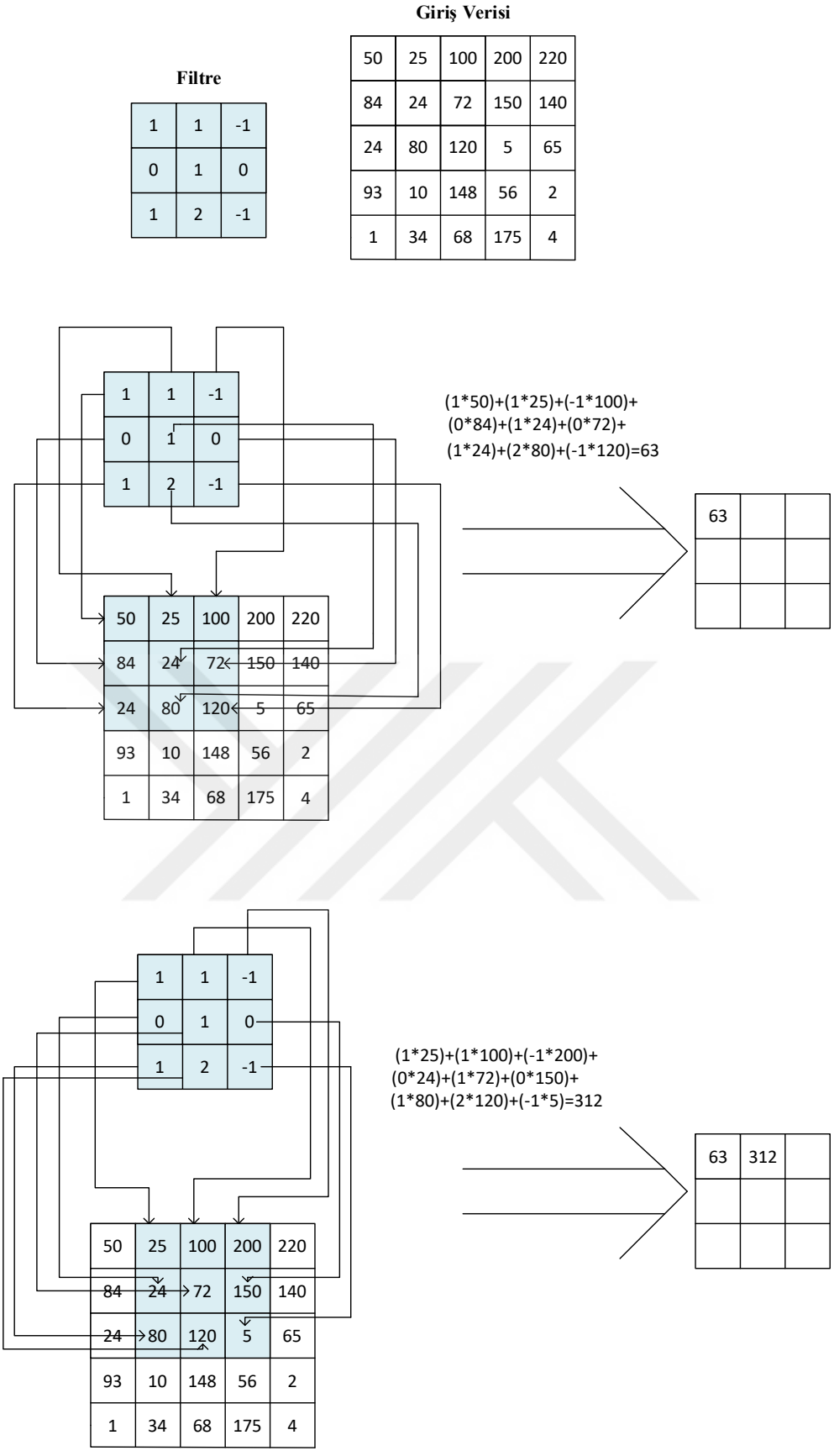
İki boyutlu konvolüsyon işlemi genellikle görüntüler için uygulanır. İki boyutlu konvolüsyonun tek boyutlu konvolüsyondan farkı, konvolüsyon için seçilen filtrenin iki boyutta hareket ettirilmesidir. İki boyutlu konvolüsyon işleminin matematiksel ifadesi, Denklem 3.3'de verilmiştir [85].

$$v_{ij}^{xy} = f\left(b_{ij} + \sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right) \tag{3.3}$$

Denklem 3.3 incelendiğinde, i. katmandaki j. özellik haritasındaki (x, y) konumundaki çıktı değeri, v_{ij}^{xy}, özellik haritası için bias değeridir. P_i, Q_i filtre boyutunu belirtir. p, q filtre endeksidir. i, giriş katmanı dizinidir. m, özellik katmanı dizinidir. j, çıkış katmanı dizinidir. w, özellik haritasının sayısına ve boyut sayısına göre filtrenin değerini ifade eder.

Konvolüsyon katmanında, görüntü verilerinin boyutuna göre filtre seçilir. Kullanıcı tarafından seçilen filtre, her bir görüntü verisi üzerinde soldan sağa ve yukarıdan aşağıya olmak üzere iki boyutta piksel piksel kaydırılır ve sırasıyla filtredeki sayısal değerler ile görüntüdeki piksel değerleri çarpılır. Çarpım sonucu çıkan değerler toplanır ve filtredeki toplam değer sayısına bölünerek elde edilen değer yeni oluşan görüntü üzerine yazılır. Konvolüsyon işlemi sonrası yeni oluşan görüntünün boyutu, ham görüntü boyutundan daha küçüktür. Şekil 3.4'de $5x5$ boyutunda matris olarak ifade edilen örnek bir giriş verisi ve $3x3$ boyutunda örnek bir filtre verilmiştir. Filtrenin giriş verisi üzerinde 1 adım kaydırılarak konvolüsyon işleminin nasıl gerçekleştiği gösterilmiştir. Konvolüsyon işlemi sonucu oluşan özellik haritasının boyutu $3x3$ tür. Giriş verisinin boyutları WxH, filtre boyutu KxL ve filtrenin adım sayısı s olduğu düşünüldüğünde özellik haritasının boyutu aşağıdaki Denklem 3.4'deki gibi formülize edilir [86].

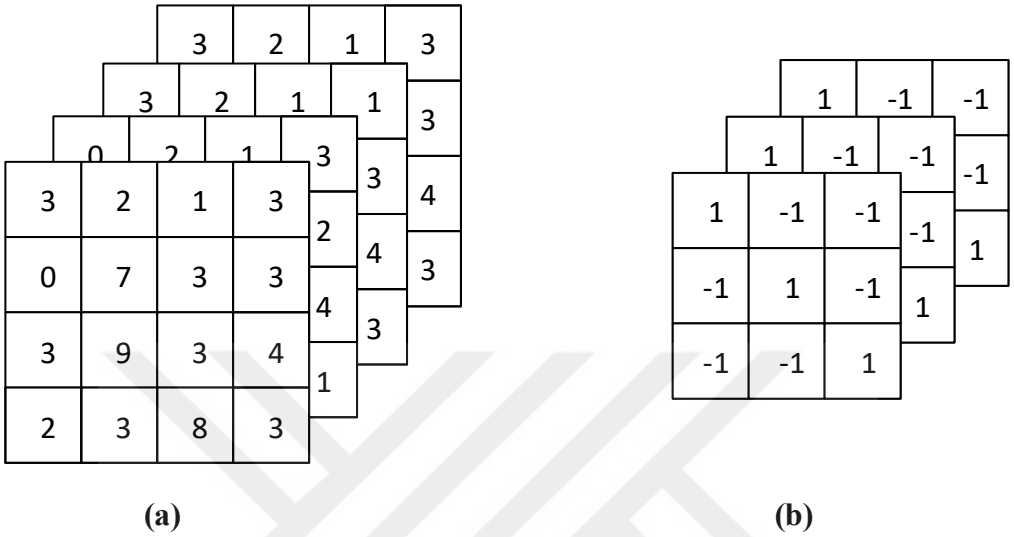
$$(WxH) * (KxL) = ((W-K)/s+1) * ((H-L)/s+1) \tag{3.4}$$



Şekil 3.4 : İki boyutlu konvolüsyon işleminin gösterimi.

3.1.1.3 Üç boyutlu konvolüsyon işlemi

Üç boyutlu konvolüsyon işlemi genellikle Manyetik Rezonans Görüntüleme (MRG) taramaları gibi tıbbi görüntüleme çalışmalarında, insan hareketi tanıma ve video sınıflandırma çalışmalarında uygulanır [87]. Esas olarak, 2B-CNN ile aynıdır. 2B-CNN'den farkı, filtrenin üç boyutta hareket etmesi ve konvolüsyon işlemi sonrası çıktının boyutudur. Üç boyut, yükseklik, uzunluk ve derinliği ifade eder. Şekil 3.5 (a)'da $4x4x4$ boyutunda örnek bir giriş verisi ve Şekil 3.5 (b)'de $3x3x3$ boyutunda örnek bir filtre verilmiştir.



Şekil 3.5 : Üç boyutlu giriş verisi ve üç boyutlu örnek fitrenin matris gösterimi: (a) $4x4x4$, (b) $3x3x3$.

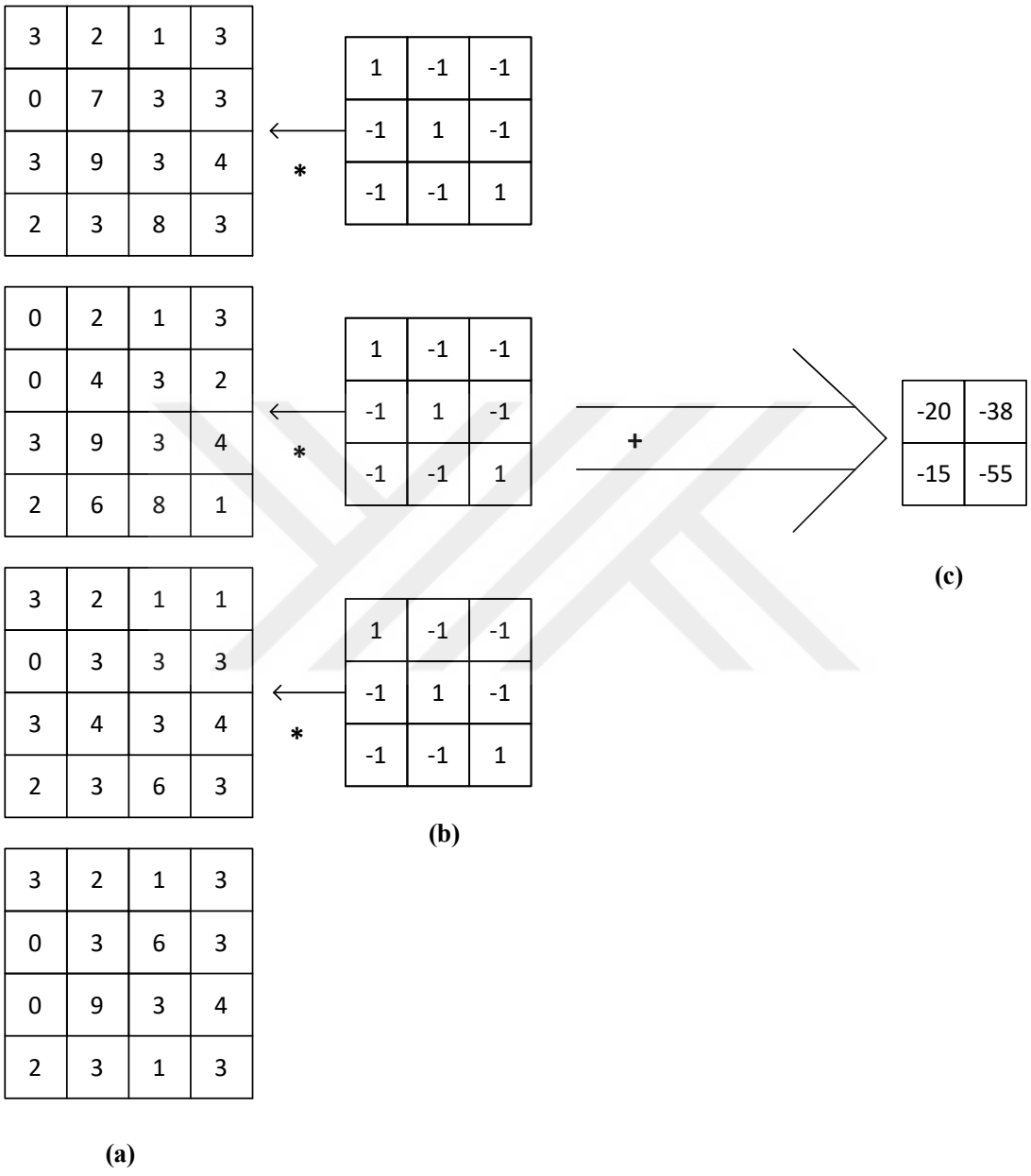
Üç boyutlu konvolüsyon işleminin matematiksel ifadesi, Denklem 3.5'de verilmiştir [85].

$$v_{ij}^{xyz} = f(b_{ij} + \sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} \, v_{(i-1)m}^{(x+p)(y+q)(z+r)}) \qquad (3.5)$$

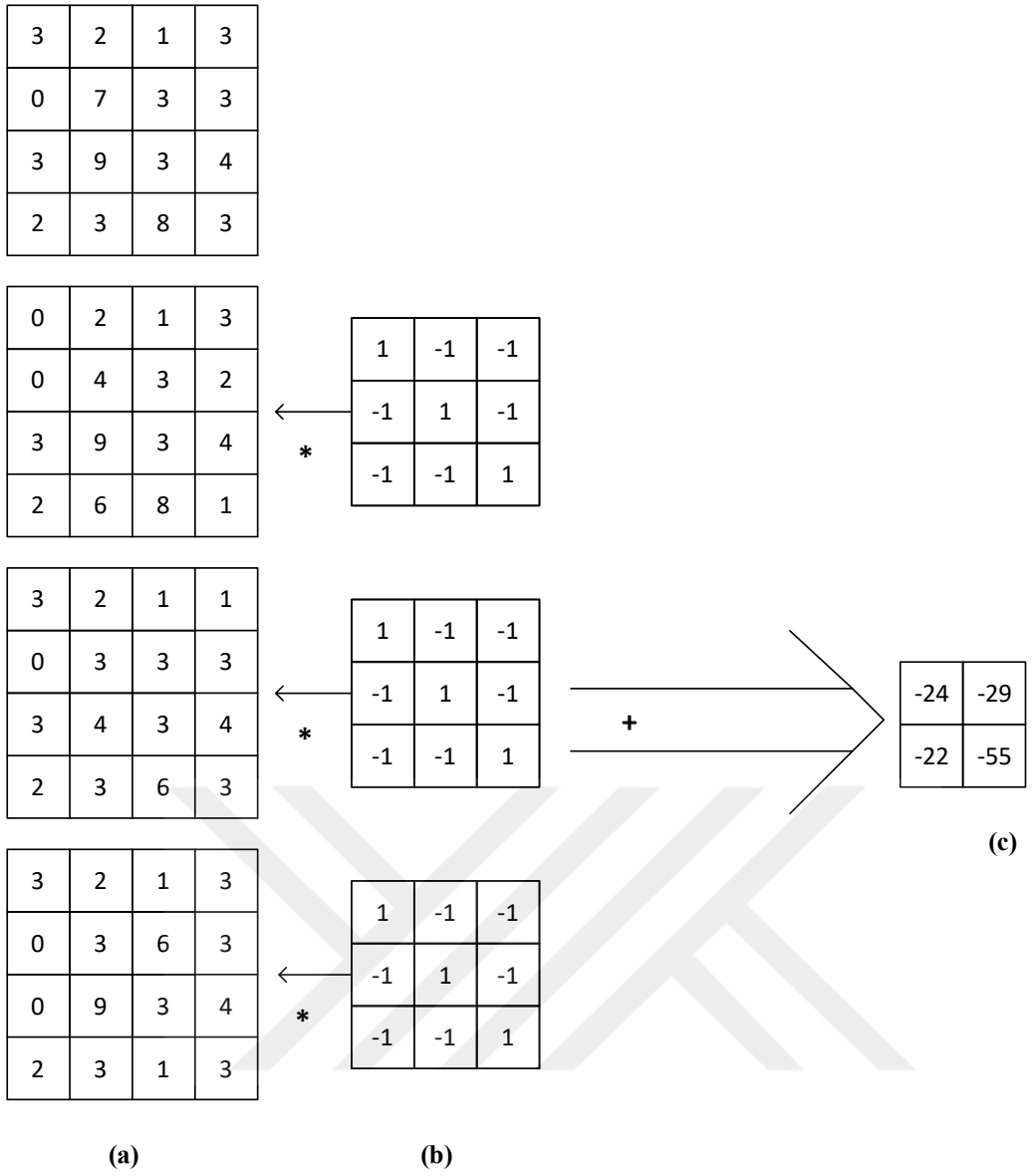
Denklem 3.5 incelendiğinde, i. katmandaki j. özellik haritasındaki (x, y, z) konumundaki çıktı değeri, v_{ij}^{xyz} olarak verilmiştir. b_{ij}, özellik haritası için bias değeridir. P_i, Q_i, R_i filtre boyutunu belirtir. p, q, r filtre endeksidir. i, giriş katmanı dizinidir. m, özellik katmanı dizinidir. j, çıkış katmanı dizinidir. w, özellik haritasının sayısına ve boyut sayısına göre filtrenin değerini ifade eder.

3 boyutta bulunan giriş verisi ve 3 boyutlu filtrenin matematiksel konvolüsyon işlemini anlaşılır bir şekilde açıklamak için, matrisler parçalara ayrılarak Şekil 3.6, Şekil 3.7

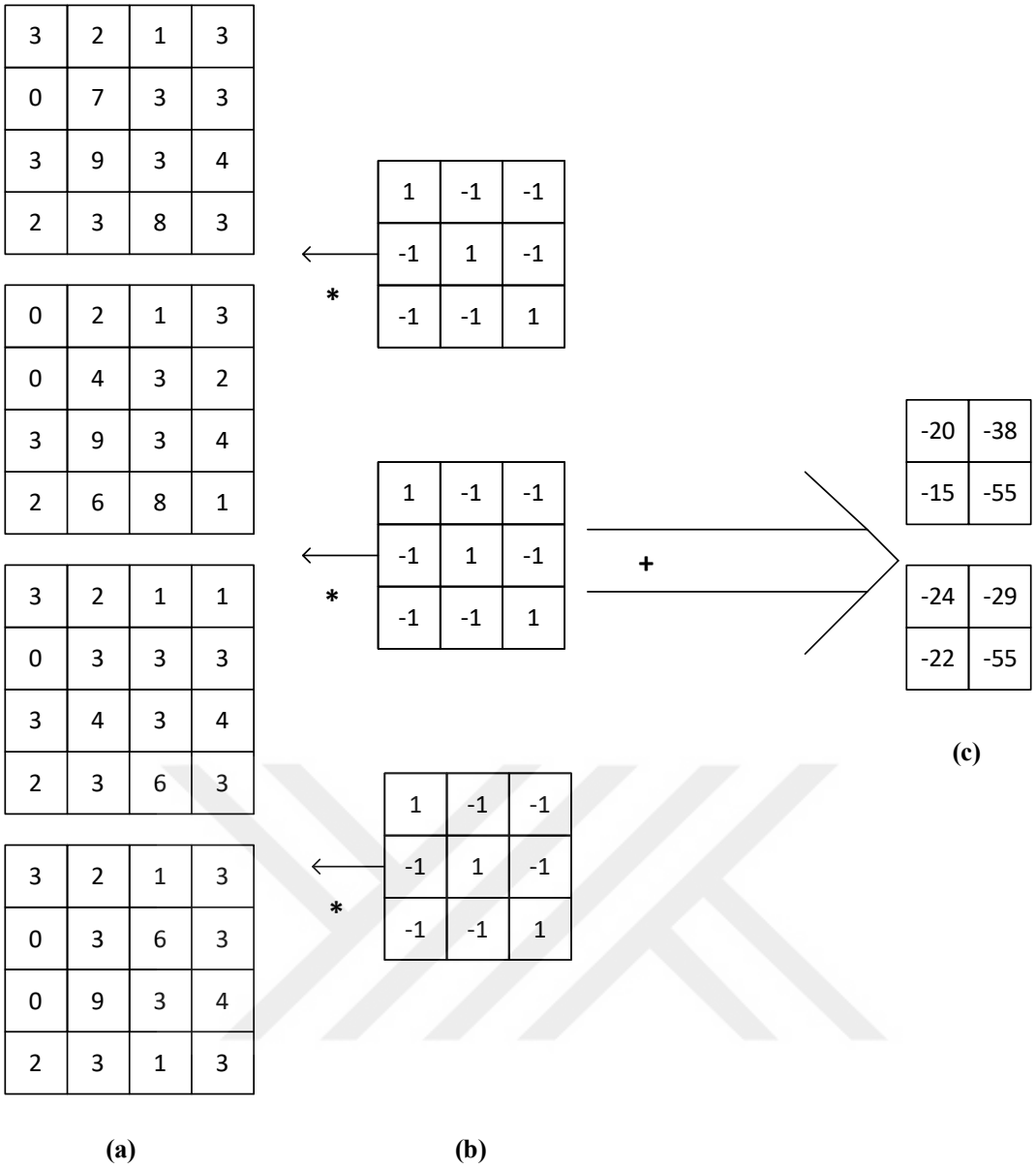
ve Şekil 3.8'de gösterilmiştir. Şekil 3.6'dan görüleceği üzere, 3 farklı $3x3$ boyutunda matris ile gösterilen filtreler, sırasıyla aynı 2B-CNN'de olduğu gibi giriş verisine uygulanır ve özellik haritasının ilk matrisi bulunur. Daha sonra Şekil 3.7'de, filtreler bir adım aşağıya kaydırılarak aynı işlem tekrar edilir ve özellik haritasının ikinci matrsisi bulunur. Son olarak, Şekil 3.8 (c)'de, üç boyutlu konvolüsyon işleminin tamamlanması sonucu elde edilen özellik haritası gösterilmiştir.



Şekil 3.6 : Üç boyutlu konvolüsyon işleminin gösterimi: (a) $4x4x4$ boyutunda giriş verisi, (b) $3x3x3$ boyutunda filtre, (c) filtrelerin, giriş verisi matrisi ile konvolüsyonu sonucu oluşan özellik haritasının ilk matrisi [88].



Şekil 3.7 : Üç boyutlu konvolüsyon işleminin gösterimi: (a) $4x4x4$ boyutunda giriş verisi, (b) $3x3x3$ boyutunda filtre, (c) filtrelerin, giriş verisi matrisi ile konvolüsyonu sonucu oluşan özellik haritasının ikinci matrisi [88].



Şekil 3.8 : Üç boyutlu konvolüsyon işleminin gösterimi: (a) $4x4x4$ boyutunda giriş verisi, (b) $3x3x3$ boyutunda filtre, (c) filtrelerin, giriş verisi matrisi ile konvolüsyonu sonucu oluşan $2x2x2$ boyutundaki özellik haritası [88].

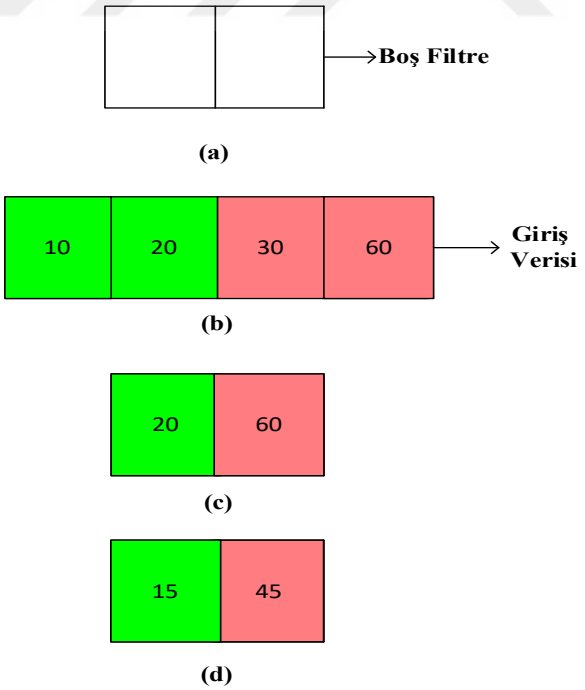
3.1.2 Havuzlama katmanı

Havuzlama katmanı, konvolüsyon çıktısının yani özellik haritasının boyutunu azaltmak için genellikle konvolüsyon katmanından sonra kullanılır. Popüler olarak kullanılan iki tür havuzlama katmanı vardır; maksimum havuzlama ve ortalama havuzlama. Maksimum havuzlama katmanında seçilen boş bir filtre, giriş verisi üzerinde adım adım gezdirilerek, filtrenin kapsadığı herbir alanda bulunan elemanlardan matematiksel olarak en büyük elemanın değeri seçilir ve özellik haritasına bu değer yazılır. Ortalama havuzlama

katmanında, seçilen boş bir filtre, giriş verisi üzerinde adım adım gezdirilerek filtrenin kapsadığı herbir alanda bulunan elemanlarının aritmetik ortalaması hesaplanır ve elde edilen değer özellik haritasına yazılır. Havuzlama katmanı kullanmaya bağlı olarak, ağırlık sayıları azalır, sinir ağının hesaplama maliyeti azalır ve bellekteki işlem yoğunluğu azalır.

3.1.2.1 Tek boyutlu havuzlama işlemi

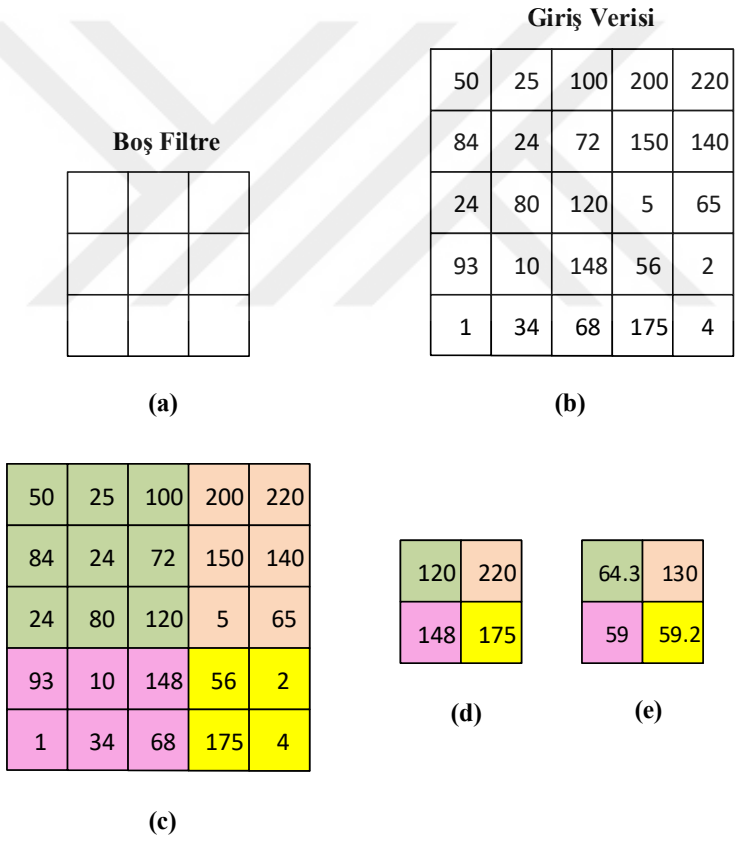
Şekil 3.9 (a)'da $1x2$ boyutunda boş filtre gösterilmiştir. Şekil 3.9 (b)'de $1x4$ boyutunda giriş verisi gösterilmiştir. Filtrenin, giriş verisi üzerinde denk geldiği bölgedeki elemanlar içerisinde, matematiksel olarak büyük olan değer seçilerek maksimum havuzlama katmanının ilk elemanı bulunur. Maksimum havuzlama katmanının ikinci elemanını bulmak için filtre, giriş verisi üzerinde bir adım kaydırılır. Filtrenin, giriş verisi üzerindeki yeni konumunda denk geldiği bölgedeki elemanlar içerisinde, matematiksel olarak büyük olan değer seçilerek maksimum havuzlama katmanının ikinci elemanı bulunur. Maksimum havuzlama işlemi Şekil 3.9 (c)'de gösterilmiştir. Filtrenin, giriş verisi üzerinde denk geldiği bölgedeki elemanların aritmetik ortalaması alınarak, ortalama havuzlama katmanının ilk elemanı bulunur. Ortalama havuzlama katmanının ikinci elemanını bulmak için filtre, giriş verisi üzerinde bir adım kaydırılır. Filtrenin, giriş verisi üzerindeki yeni konumunda denk geldiği bölgedeki elemanların aritmetik ortalaması alınarak, ortalama havuzlama katmanının ikinci elemanı bulunur. Maksimum havuzlama işlemi Şekil 3.9 (d)'de gösterilmiştir.



Şekil 3.9 : Tek boyutlu havuzlama işleminin bloklarla gösterimi: (a) filtre, (b) giriş verisi, (c) $1x2$ maksimum havuzlama, (d) $1x2$ ortalama havuzlama.

3.1.2.2 İki boyutlu havuzlama işlemi

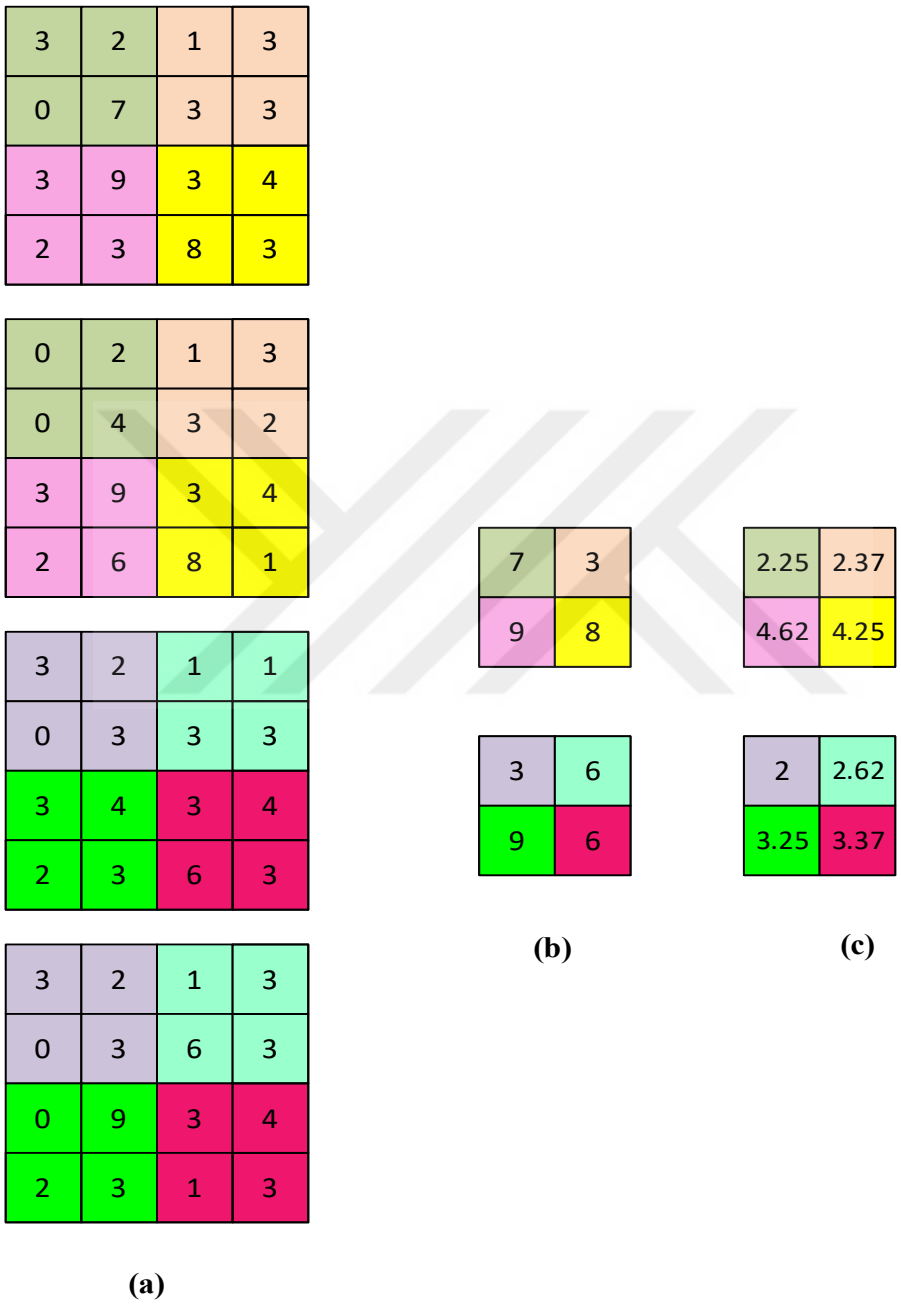
Tek boyutlu havuzlama işleminden farklı olarak, seçilen boş filtre, giriş verisi üzerinde soldan sağa ve yukarıdan aşağıya belirlenen adım sayısına bağlı olarak iki boyutta kaydırılır. Kullanıcı maksimum havuzlama katmanını tercih ederse, filtre giriş verisi üzerinde kapsadığı alandaki elemanlar içerisinde en büyük sayıyı alır ve özellik haritasına atar. Kullanıcı ortalama havuzlama katmanını tercih ederse, filtre giriş verisi üzerinde kapsadığı alandaki elemanların aritmetik ortalamasını alır ve özellik haritasına atar. Tüm bu işlemler sırasıyla, filtrenin giriş verisi üzerinde kaydırılarak yerleştiği her yeni konumu için tekrarlanır. Şekil 3.10 (a)'da $3x3$ boyutunda boş bir filtre verilmiştir. Şekil 3.10 (b)'de $5x5$ boyutunda giriş verisi gösterilmiştir. Filtre, giriş verisi üzerinde bir adım soldan sağa ve yukarıdan aşağıya hareket ettirilerek, Şekil 3.10 (d)'de verilen maksimum havuzlama işlemi gerçekleştirilmiştir. Daha sonra, Şekil 3.10 (e)'de verilen ortalama havuzlama işlemi gerçekleştirilmiştir.



Şekil 3.10 : İki boyutlu havuzlama işleminin gösterimi: (a) filtre matrisi, (b) giriş verisi,(c) filtrenin, giriş verisi üzerinde kapsadığı alanların renkli gösterimi, (d) maksimum havuzlama, (e) ortalama havuzlama.

3.1.2.3 Üç boyutlu havuzlama işlemi

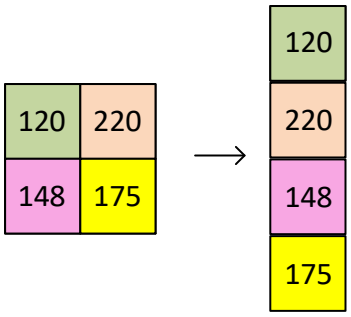
Üç boyutlu havuzlama işlemi, iki boyutlu havuzlama işlemine benzerdir. Kullanıcı tarafından seçilen boş filtre, giriş verisi üzerinde soldan sağa ve yukarıdan aşağıya belirlenen adım sayısına bağlı olarak üç boyutlu olarak kaydırılır. Şekil 3.11'de $4x4x4$ boyutunda verilen giriş verisi için hem maksimum havuzlama hemde ortalama havuzlama katmanları oluşturulmuştur.



Şekil 3.11 : Üç boyutlu giriş verisi üzerinde havuzlama işlemlerinin gösterimi: (a) $4x4x4$ boyutunda verilen giriş verisi, (b) $2x2x2$ maksimum havuzlama, (c) $2x2x2$ ortalama havuzlama.

3.1.3 Düzleştirme katmanı

Havuzlama katmanından sonra ortaya çıkan matris şeklindeki iki boyutlu özellik haritasını, tek boyutlu bir sütun formatına dönüştürmek için uygulanan basit bir adımdır. Şekil 3.12'de $2x2$ boyutunda verilen özellik haritasının, düzleştirme katmanından geçtikten sonra oluşan yeniden biçimlendirilmiş hali gösterilmiştir.



Şekil 3.12 : Düzleştirme işleminin gösterimi.

3.1.4 Tamamen bağlı katman

Tamamen bağlı katman, ileri beslemeli YSA mimarisinin kullanıldığı ve sınıflandırmanın gerçekleştiği katmandır. Tamamen bağlı katman, CNN mimarisinin son birkaç katmanını oluşturur. CNN mimarisinde, çıkarılan özelliklerin sınıflandırılması gerekmektedir. Sınıflandırma işleminin yapılabilmesi için düzleştirme katmanı sonucu sütun vektörüne dönüştürülen özellik haritası, tamamen bağlı katmana aktarılır. Tamamen bağlı katmanda, özellik haritasının sınıflandırma işlemi tamamlandıktan sonra çeşitli aktivasyon fonksiyonları kullanılır.

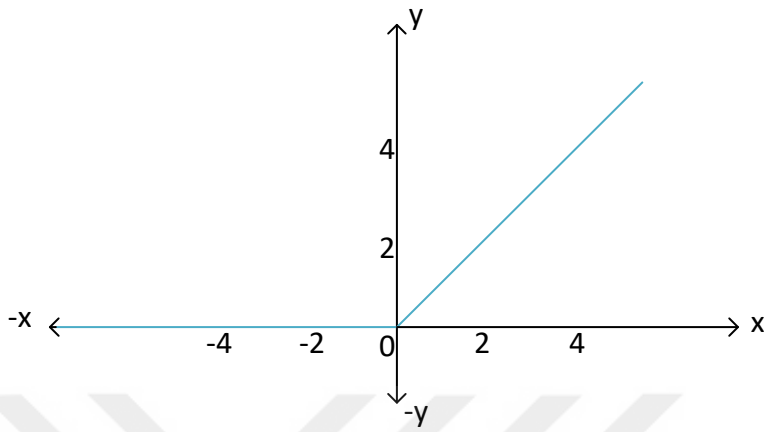
3.1.5 Aktivasyon fonksiyonları

Tanım gereği, bir aktivasyon fonksiyonu, bir nöronun aktive edilip edilmeyeceğine karar verir. Aktivasyon fonksiyonu, her bir nöronun çıktısını optimize eder. Aktivasyon fonksiyonları olmayan bir sinir ağı, sadece doğrusal bir regresyon modelidir. Aktivasyon fonksiyonları nöronun çıktısına, doğrusal olmama özelliği getirir. Çünkü, doğrusal olan bir fonksiyon sınırlı değildir. Doğrusal olmama özelliği, fonksiyonu belirli aralıklarda sınırlar. Sinir ağlarında sınıflandırma ve tespit görevleri için kullanılan bazı popüler aktivasyon fonksiyonları aşağıda ayrı başlıklar altında verilmiştir.

3.1.5.1 ReLU fonksiyonu

Derin öğrenmede en yaygın kullanılan aktivasyon fonksiyonlarından biridir. ReLU fonksiyonu, Denklem 3.6'da gösterildiği gibi hesaplanır [89].

$$ReLU(x) = max(0, x) \tag{3.6}$$



Şekil 3.13 : ReLU fonksiyonu [89].

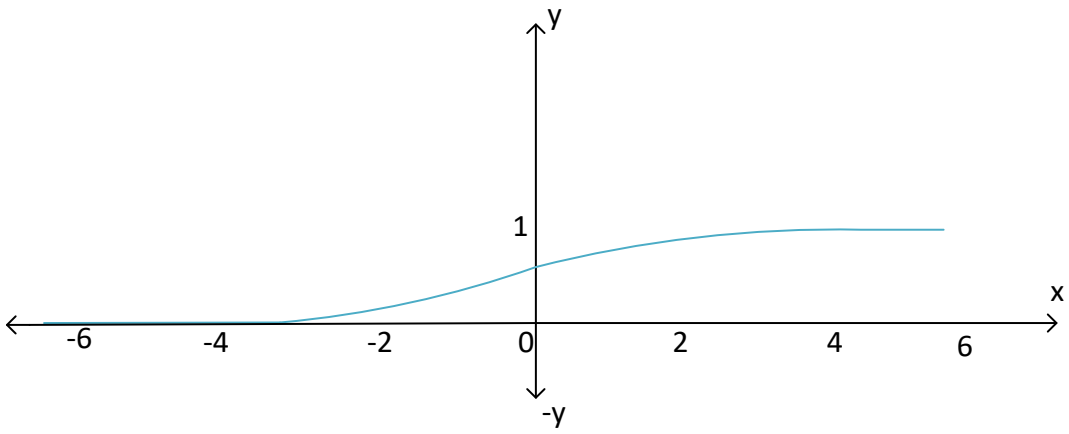
Doğrusal olmayan bir ağın oluşturulması için oldukça popülerdir. Şekil 3.13'de ReLU fonksiyonunun grafiği verilmiştir. ReLU fonksiyonu, 0 dışında her yerde türevlenebilir. Ayrıca, sigmoid ve hiperbolik tanjant fonksiyonlarından daha hızlı yakınsar [90].

3.1.5.2 Sigmoid fonksiyonu

Giriş olarak gerçek değerleri alıp [0, 1] aralığındaki değerleri çıkarır. Sigmoid fonksiyonu şu şekilde hesaplanır [89]:

$$sig(x) = \frac{1}{1 + e^{-x}} \tag{3.7}$$

Daha büyük negatif girdi değerleri 0'a yakın olma eğilimindeyken, daha büyük pozitif girdiler 1'e daha yakındır. Şekil 3.14'de verilen sigmoid fonksiyonunun grafiği incelendiğinde, fonksiyonun "S" şeklinde ve türevlenebilir olduğu görülmektedir.

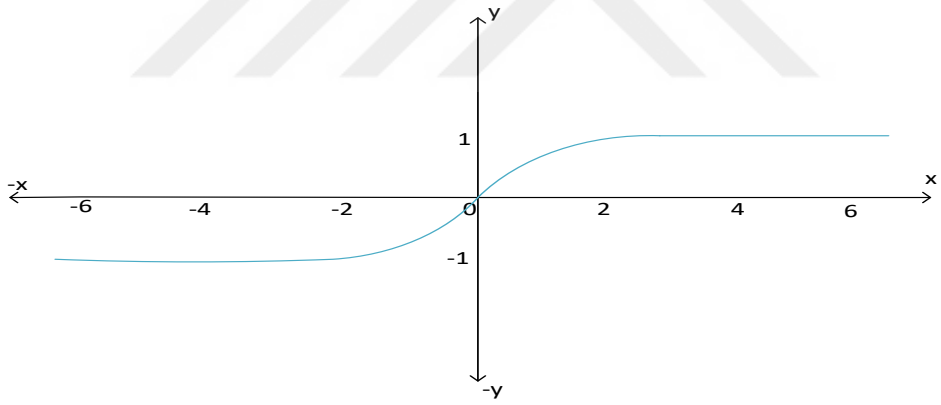


Şekil 3.14 : Sigmoid fonksiyonu [89].

3.1.5.3 Tanh fonksiyonu

Tanh fonksiyonu, sigmoid fonksiyonuna çok benzemektedir. Fakat sigmoid fonksiyonu 0 ile 1 arasında sınırlı iken tanh fonksiyonu Şekil 3.15'den de görüleceği üzere -1 ile 1 arasında sınırlıdır. Tanh fonksiyonu şu şekilde hesaplanır [89]:

$$tanh(x) = 2\,sig(x) - 1 = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{3.8}$$



Şekil 3.15 : Tanh fonksiyonu [89].

3.1.5.4 Softmax fonksiyonu

Softmax Fonksiyonu genellikle sınıflandırma görevi için oluşturulmuş çok katmanlı sinir ağlarının çıktı katmanında kullanılır. Fonksiyon her sınıfın olasılığını hesaplar. Kesin bir seçim yapmak yerine softmax, sınıflar arasında girdinin hangi sınıfa ait olduğunu tahmin eder. M'nin sınıfları temsil ettiği düşünülürse, softmax fonksiyonu Denklem 3.9'da gösterildiği gibi hesaplanır.

$$softmax(x_k) = \frac{e^{x_k}}{\sum_{i=1}^{M} e^{x_i}} \quad (3.9)$$

3.2 Nesne Tanıma İçin Bölge Tabanlı Evrişimli Sinir Ağları

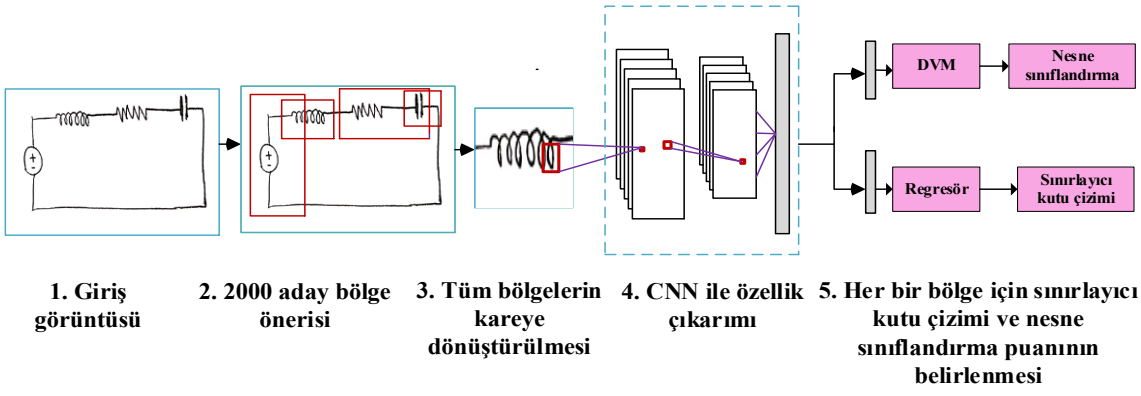
Sınıflandırmada genellikle odak olarak tek bir nesneye sahip bir görüntü vardır ve sınıflandırmanın amacı, görüntüde ne tür bir nesnenin mevcut olduğunu belirlemektir. Ancak sınıflandırma ile görüntüde bulunan nesnenin bulunduğu konumu belirlemek mümkün değildir. Nesne algılama, görüntüde farklı sınıflara ait nesnelerin tespit edilmesinde kullanılan yöntemdir. Nesne algılama iki aşamadan oluşur. Biri, nesne içerebilecek bölgeleri bulmak olan bölge önerisi aşamasıdır. Bir diğeri, önerilen bölgenin belirli bir nesne içerip içermediğini belirleyen görüntü sınıflandırma aşamasıdır. Geleneksel nesne algılama yöntemleri, ilk olarak bir görüntüde nesne içerebilecek çok sayıda bölgeyi tarar. Daha sonra her bölgeden, özellik çıkarma algoritmalarından olan SIFT [91], HOG [92] ve Haar benzeri [93] algoritmalar kullanılarak özellikler çıkarılır. Son olarak, çıkarılan özellikler, DVM [52], AdaBoost [94] veya DPM [95] gibi sınıflandırıcılarda değerlendirilerek, sınıflandırma işlemi gerçekleştirilir.

Derin öğrenmenin gelişmesiyle birlikte, nesne algılama alanı son zamanlarda derin öğrenme temelli yöntemlerin hakimiyetine girmiştir. Derin öğrenme tabanlı nesne tanıma yöntemleri, eğitim için çok sayıda ve doğru şekilde etiketlenmiş görüntü içeren veri kümeleri gerektirir. PASCAL VOC 2012 veri kümesi [96], ImageNet veri kümesi [97] ve COCO [98] gibi çeşitli veri kümeleri mevcuttur. PASCAL VOC 2012 veri kümesi, 27450 nesnenin yaklaşık 11540 görüntüsünü içeren nispeten daha küçük bir veri kümesidir ve bunların küçük bir kısmı nesne bölümleme bilgisine sahiptir. ImageNet veri kümesi, 22K sınıfın 1,5 milyondan fazla etiketli görüntüsünü içerir. 200K sınıfa sahip COCO veri kümesindeki görüntü sayısı, ImageNet'ten daha azdır. Buna rağmen tüm görüntüler nesne bölümleme bilgilerini içerir. Derin öğrenmenin bir alt alanı olan CNN kullanılarak, üç farklı bölge tabanlı nesne algılama yöntemi geliştirilmiştir. Bu yöntemler: R-CNN, Fast R-CNN ve Faster R-CNN.

3.2.1 Bölge tabanlı evrişimli sinir ağları (B-ESA)

Bölge Tabanlı Evrişimli Sinir Ağları (B-ESA) kavramının orjinal ve yaygın olarak kullanılan kısaltması, R-CNN'dir. B-ESA ve R-CNN eş anlamlı olarak kullanılmıştır.

R-CNN ilk olarak 2014 yılında Ross Girshick, Jeff Donahue, Trevor Darrell ve Jitendra Malik [99] tarafından tanıtılmıştır. Nesnelerin konumlarını tespit etmenin ilk adımı, test edilecek bir dizi potansiyel sınırlayıcı kutu veya ilgi alanı oluşturmaktır. R-CNN modeli üç modülden oluşur. İlk modül, giriş görüntüsünü olası nesneler için harici bir algoritma kullanarak tarayarak bölge önerileri üretir. Nesnelik [100], Seçici Arama [101], Kategoriden Bağımsız Nesne Teklifleri [102], Sınırlandırılmış Parametrik Min-Kesmeler (CPMC) [103], Çok Ölçekli Kombinatoryal Gruplama [104] gibi bölge önerileri oluşturmaya yönelik çeşitli yöntemler vardır. En yaygın olarak kullanılan bölge önerisi oluşturma yöntemi, görüntü üzerinde 2000 adet ilgi bölgesi (ROI) önerisi yapan seçici arama algoritmasıdır. Seçici arama, bölge önerileri için geleneksel görüntü işleme tekniklerini kullanır. Bölge önerileri farklı şekil ve boyutlarda olabilir. Bu 2000 aday bölge önerisi, bir kareye dönüştürülür ve çıktı olarak 4096 boyutlu bir özellik vektörü veren bir CNN'e aktarılır. Çıkarılan her özellik, bir DVM kullanılarak her sınıf için puanlanır. Son olarak, nesnenin bulunduğu konumu belirlemek için sınırlayıcı kutu çizimi gereklidir. Bunun için regresyon modeli kullanılır. R-CNN mimarisinde gerçekleşen tüm adımlar, Şekil 3.16'da sırasıyla gösterilmiştir. R-CNN yönteminde, önerilen tüm bölgeler için CNN uygulandığı için eğitim uzun sürmektedir ve yavaştır. R-CNN yönteminde eğitim, 84 saatte tamamlanır. Tek bir görüntü için test süresi ise 47 saniyedir [105]. R-CNN'de eğitim süresi ve test süresi uzun olduğu için, gerçek zamanlı uygulamalarda kullanımı zordur. R-CNN modelini tasarlayan bilim insanları R-CNN'nin dezavantajlı olduğu noktaların farkına varıp Fast R-CNN adında yeni bir model oluşturmuşlardır [106].

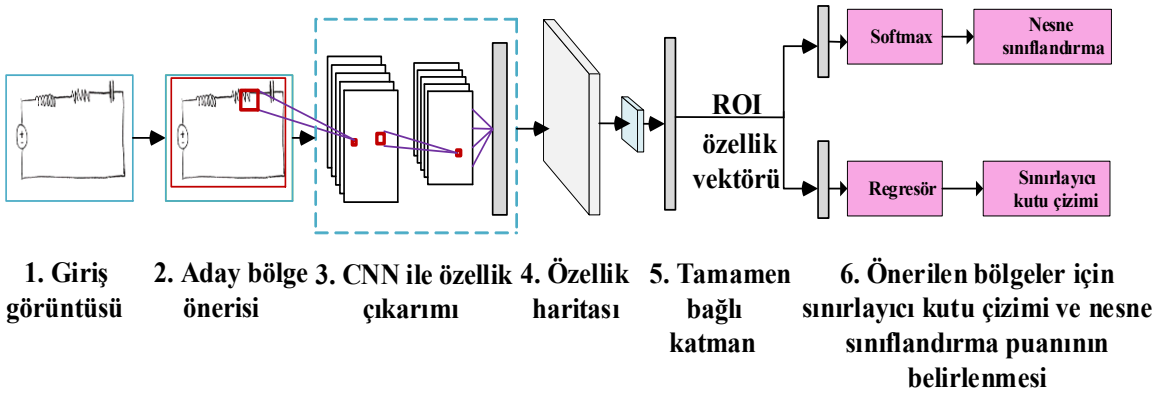


Şekil 3.16 : R-CNN mimarisi [107].

3.2.2 Hızlı bölge tabanlı evrişimli sinir ağları (Hızlı B-ESA)

Hızlı Bölge Tabanlı Evrişimli Sinir Ağları (Hızlı B-ESA) kavramının orjinal ve yaygın olarak kullanılan kısaltması, Fast R-CNN'dir. Hızlı B-ESA ve Fast R-CNN eş anlamlı olarak kullanılmıştır. Fast R-CNN, R-CNN mimarisini hızlandırmak, basitleştirmek ve test süresini kısaltmak için oluşturulmuştur [106]. R-CNN ile arasındaki temel fark, ilk aşamada tüm giriş görüntüsüne çeşitli evrişim ve maksimum havuz katmanları içeren bir CNN uygulanması ve bir tane evrişimli özellik haritasının oluşturulmasıdır. Daha sonra, Fast R-CNN'de R-CNN'den farklı olarak, görüntüye CNN uygulandıktan sonra, seçici arama algoritması kullanılarak 200 farklı ROI önerisi yapılır. Önerilen 200 farklı ROI'nin hepsi farklı boyutlardadır. Farklı boyutlarda bulunan ROI'lerin özellik haritasının, standart ve daha küçük boyuta getirilmesi için, R-CNN'den farklı olarak ROI havuzlama katmanı kullanılmaktadır. ROI havuzlama katmanının girişine verilen özellik haritası, tamamen bağlı katmana iletilerek sabit uzunlukta özellik vektörüne dönüştürülür. Elde edilen bu özellik vektörü, çıkış katmanlarına iletilir. Çıkış katmanlarından birincisi, nesneleri sınıflandıran ve nesne sınıfları için olasılık tahminleri üreten softmax katmanı, ikincisi ise doğrusal regresyon kullanılarak hesaplanan sınırlayıcı kutu koordinatlarını çıkaran gerçek değerli katmandır. Fast R-CNN mimarisinde gerçekleşen tüm adımlar Şekil 3.17'de sırasıyla gösterilmiştir.

Fast R-CNN yönteminde eğitim, 8.75 saatte tamamlanır. Tek bir görüntü için test süresi ise 2.3 saniyedir [108]. Literatüre göre Fast R-CNN, son teknoloji ürünü bir GPU kullanıldığında bir saniyeden daha kısa süren, normal R-CNN'e kıyasla önemli ölçüde daha kısa sınıflandırma süresi sağlar [106]. Bunun başlıca nedeni, her ROI için aynı özellik haritasının kullanılmasıdır.



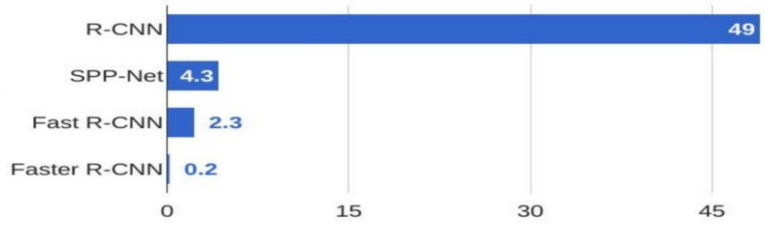
Şekil 3.17 : Fast R-CNN mimarisi [106].

3.2.3 Daha hızlı bölge tabanlı evrişimli sinir ağları (Daha Hızlı B-ESA)

Daha Hızlı Bölge Tabanlı Evrişimli Sinir Ağları (Daha Hızlı B-ESA) kavramının orjinal ve yaygın olarak kullanılan kısaltması, Faster R-CNN'dir. Daha Hızlı B-ESA ve Faster R-CNN eş anlamlı olarak kullanılmıştır. R-CNN'den, Fast R-CNN'e kadar tek bir görüntü için test süresi önemli ölçüde azalmasına rağmen, her iki modelde de bölge önerileri için seçici arama algoritması kullanılmıştır. Seçici arama algoritması yavaş ve zaman alıcıdır ve ağın davranışını etkiler. Bu nedenle, Shaoqing Ren ve arkadaşları [109], seçici arama algoritmasını ortadan kaldıran ve ağın bölge önerilerini öğrenmesine izin veren "Faster R-CNN" adında yeni bir nesne algılama algoritması oluşturmuşlardır. Bölge önerileri için oluşturulan ve seçici arama algoritmasının yerine kullanılan ağ, Bölge Teklif Ağı (RPN)'dir.

RPN, diğer nesne algılama mimarilerinde kullanılan seçici arama algoritmasından farklıdır. RPN, seçici arama algoritmasına göre daha az sayıda bölge önerisi yapar. RPN, giriş görüntüsüne CNN uygulandıktan sonra çıkarılan özellik haritası üzerinde bölge önerileri yapmak için kullanılan tam evrişimli ağdır. RPN, ilk olarak özellik haritası üzerinde bölge önerileri için kayar pencere oluşturur. Kayar pencerenin özellik haritası üzerinde kaydırıldığında, RPN görüntüde nesne olabilecek bölgeler için bir dizi dikdörtgen sınırlayıcı kutu önerisi ve skoru olan bağlantı kutuları kullanır [110]. Bağlantı kutuları, görüntünün tamamına yayılmış, önceden tanımlanmış, sabit boyutlu kutulardır. Bağlantı kutuları, görüntü üzerinde farklı boyutlarda ve farklı şekillerde bulunur. Bu kutuların farklı boyutlarda olması, görüntü üzerinde daha fazla nesne tespitine olanak sağlar [111]. Skoru olan bağlantı kutuları, önerilen bölgenin nesne içerdiğinden ne kadar emin olduğunu puan olarak gösterir. Kayar pencere, özellik haritasında taradığı her bir konumda, nesne bulunma ihtimali olan birkaç bölge önerisi yapar. RPN, eğitim aşamasında bağlantı kutularını kullanarak bölge önerileri oluşturmayı öğrenir. Eğitim işlemi tamamlandıktan sonra RPN, önerilen bölgelerin nesne içerme olasılığının belirlenmesi için sınıflandırma işlemi, sınırlayıcı kutuların koordinatlarının belirlenmesi için regresyon işlemi yapılır. Böylece Faster R-CNN'de, RPN kullanılarak bölge önerme aşaması tamamlanır.

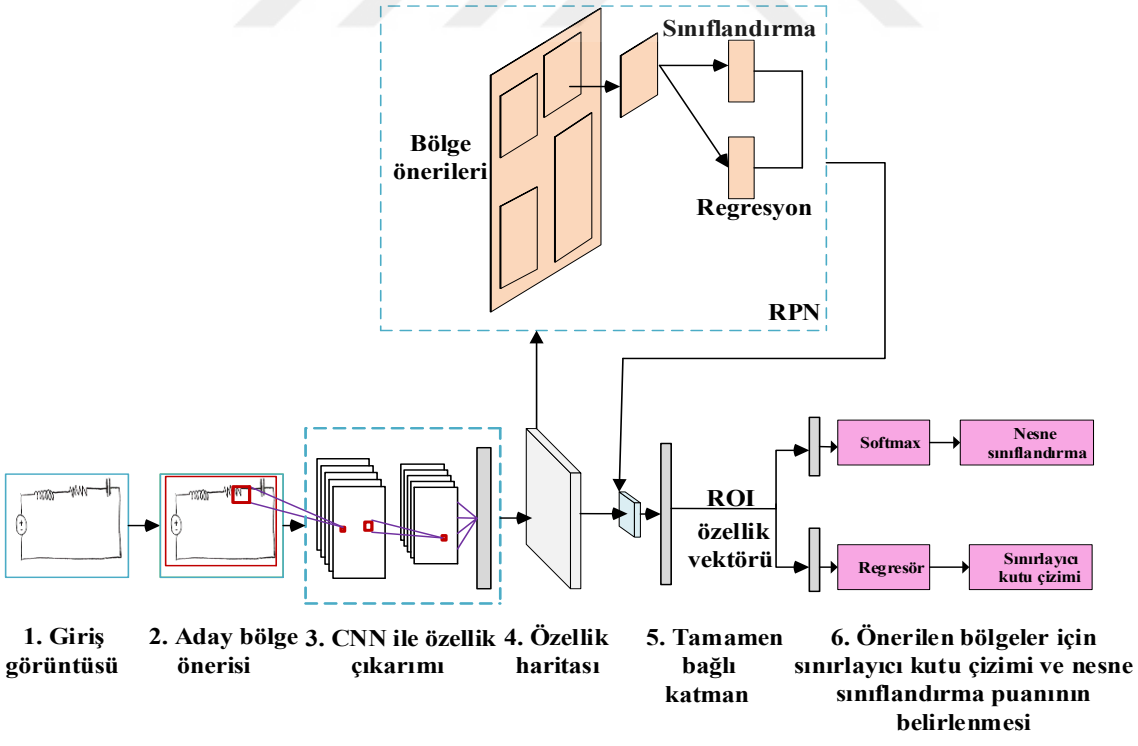
Faster R-CNN, R-CNN'den 250 kat ve Fast R-CNN'den 25 kat daha hızlıdır. Faster R-CNN'in test süresi görüntü başına yaklaşık 0,2 saniyedir [112]. Şekil 3.18'de, CNN tabanlı nesne algılama modellerinin test görüntüsü başına algılama süreleri gösterilmiştir. Şekil 3.18'den de görüleceği üzere, görüntü algılama süresinin en az olduğu yani en hızlı model Faster R-CNN modelidir.



Şekil 3.18 : Bölge tabanlı CNN mimarilerinin görüntü başına algılama süresinin karşılaştırılması [113].

Tez çalışması kapsamında, elle çizilmiş devreler üzerinde bulunan devre elemanlarının tespiti için, eğitim ve test sürelerinin kısa olması sebebiyle Faster R-CNN modeli tercih edilmiştir.

Faster R-CNN, RPN ve bir nesne detektörü olan Fast R-CNN'in kombinasyonu olarak düşünülebilir. Bundan dolayı, Faster R-CNN mimarisi iki ayrı bölüme ayrılarak incelenir. İlk bölümde RPN, görüntü üzerinde tespit edilmesi istenen nesnelerin, görüntüde hangi bölgelerde bulunabileceği ile ilgili bölge önerileri yapar. Faster R-CNN mimarisinin ikinci bölümünde, Şekil 3.19'da verilen mimariden de görüleceği üzere, bölge önerileri tamamlandıktan sonraki aşamalar Fast R-CNN ile aynıdır. Şekil 3.19'da Faster R-CNN mimarisine ait çalışma adımları ve giriş görüntüsü olarak, tez çalışması kapsamında kullanılan elle çizilmiş devre görüntüsü verilmiştir.



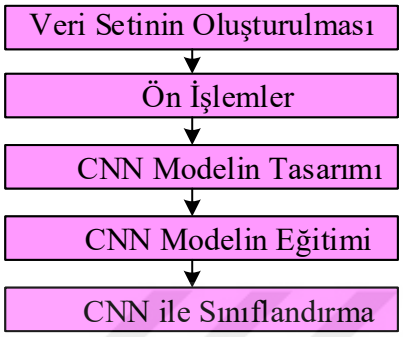
Şekil 3.19 : Faster R-CNN mimarisi [109,114].

Faster R-CNN mimarisinin girişine verilen görüntünün tamamına CNN uygulanır. Görüntüye uygulanan CNN'de, konvolüsyon ve havuzlama katmanları bulunmaktadır. Şekil 3.19'da belirtilen 3.aşamada gösterildiği gibi, görüntüden özellik çıkarma aşaması tamamlanır. Özellik çıkarma aşamasının tamamlanmasıyla, özellik haritası elde edilir. Faster R-CNN'in ilk bölümü olarak incelenen RPN algoritması, özellik haritası üzerinde çalışmaya başlar. RPN algoritmasının kullanılmasıyla bölge önerme aşaması hızlı bir şekilde tamamlanır. Daha sonra, Faster R-CNN'de ikinci bölüme geçilir. Önerilen bölgeler için, Fast R-CNN'de kullanılan ROI havuzlama katmanı kullanılır. ROI havuzlama katmanının girişine verilen özellik haritası, yeniden şekillendirilerek sabit boyutlu özellik haritasına dönüştürülür. Elde edilen yeni özellik haritası, tamamen bağlantılı katmanlara giriş olarak verilir. Tamamen bağlı katmanda, özellik haritası düzleştirilerek özellik vektörü haline getirilerek çıkış katmanına iletilir. Çıkış katmanında ise, önerilen bölgeler için regresör kullanılarak sınırlayıcı kutu çizimi ve softmax kullanılarak bulunan nesnelerin sınıflandırılması işlemi gerçekleşir.

4. DENEYSEL ÇALIŞMALAR VE BULGULAR

4.1 Deneysel Çalışma I: CNN Kullanarak Temel Devre Elemanlarının Sınıflandırılması

Bu deneysel çalışmada, elle çizilmiş devre elemanlarının görüntülerini tanıyarak birbirinden ayırt etmek ve yüksek doğruluklarla sınıflara atamak için farklı CNN modelleri geliştirilmiştir. Önerilen CNN modeli ile görüntü sınıflandırma metodu 5 ana aşamadan oluşmaktadır. Bu aşamalar sırasıyla Şekil 4.1'de gösterilmiştir.



Şekil 4.1 : Önerilen yöntemin çalışma adımları.

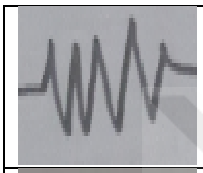
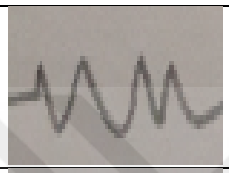
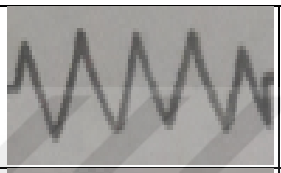
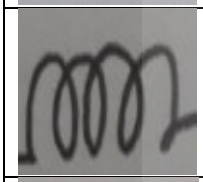
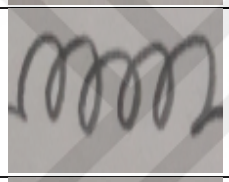
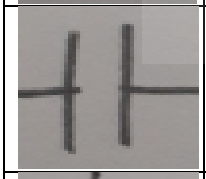
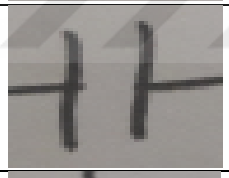
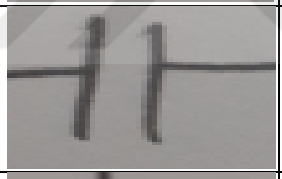
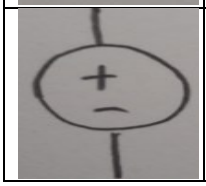
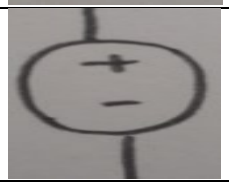
Şekil 4.1'den de görüleceği üzere, ilk aşama veri setinin oluşturulması, ikinci aşama ise veri setinde yapılan ön işlemlerdir. Sonraki aşamalarda derin öğrenme mimarisi olan CNN modeli kullanılmıştır. Sırasıyla model tasarımı, modelin eğitimi ve sınıflandırma işlemleri gerçekleştirilmektedir. Derin öğrenme uygulamalarında, eğitim ve test işlemlerinin daha hızlı tamamlanabilmesi için genellikle GPU ekran kartına sahip bilgisayarlar kullanılır. Her bilgisayarda GPU bulunmamaktadır. Ekran kartı yetersiz bilgisayarlar için, Google şirketi, internet üzerinden ücretsiz NVIDIA Tesla K80 GPU ekran kartını kullanabilme imkanı sağlayan Google Colaboratory (Colab) isimli bulut servisini oluşturmuştur. Python dilini destekleyen Colab bulut servisi; Keras, TensorFlow, OpenCV ve diğer bir çok kütüphane için hiçbir ek kurulum gerektirmeden kullanılabilir. Çalışmada, CNN modeliyle eğitim ve test aşamalarında ekran kartı ihtiyacına çözüm sunduğu için Colab uygulaması tercih edilmiştir.

4.1.1 Veri seti ve ön işlemler

Deneysel çalışmada, Munzur Üniversitesi Mühendislik Fakültesi öğrencilerinin elle çizdikleri elektronik devre elemanlarının görüntülerinden yeni bir veri seti oluşturulmuş ve tüm işlemlerde bu veri seti kullanılmıştır. Çizelge 4.1'de veri setinde bulunan örnek

görüntüler yer almaktadır. Birçok kişinin kağıtlara farklı stillerle çizdiği el çizimi elektronik devre bileşenlerinin, fotoğraf makinesi yardımıyla alınan görüntüleri için iki ayrı dosya oluşturulur. Bu dosyalardan birincisi eğitim veri seti dosyası, ikincisi ise test veri seti dosyasıdır. Çalışmada elle çizilen direnç, indüktör, kapasitör ve DC voltaj kaynağı olmak üzere 4 farklı bileşenin sınıflandırılması yapıldığı için her eğitim veri seti dosyası içerisinde 4 ayrı klasör oluşturulur. Bu klasörler; direnç, indüktör, kapasitör ve DC voltaj kaynağı klasörleridir. Aynı işlem test veri seti dosyası içinde 4 ayrı klasör oluşturularak tekrarlanır. Böylece her bir sınıfa ait toplanan görüntü verileri, ayrı klasörlerde düzenlenerek karmaşıklığın önüne geçilir. Eğitim veri setinde toplamda 709 adet görüntü bulunmaktadır. Test veri setinde ise 154 adet görüntü bulunmaktadır. Veri setine ait detaylı bilgiler Çizelge 4.2'de verilmiştir.

Çizelge 4.1 : Veri setinde bulunan örnek görüntüler.

			**Direnç**
			**İndüktör**
			**Kapasitör**
			**DC Voltaj Kaynağı**

Görüntü ön işleme, modeli eğitmeden önce yapılması gereken bir hazırlık aşaması şeklinde düşünülebilir. Ön işlemenin amacı, görüntü özelliklerini iyileştirmek, görüntü boyutunu değiştirmek, görüntünün sahip olduğu renk özelliğini değiştirmek gibi görüntülerle ilgili bir takım düzenlemelerdir.

Çalışmada ilk olarak, elle çizilmiş devre elemanlarının görüntüleri farklı boyutlarda olduğu için görüntü verileri yeniden boyutlandırılarak, 150*150 boyutuna getirilmiştir. Veri setindeki tüm görüntüler aynı boyuta indirgendikten sonra gri tonlamaya dönüştürülmüştür.

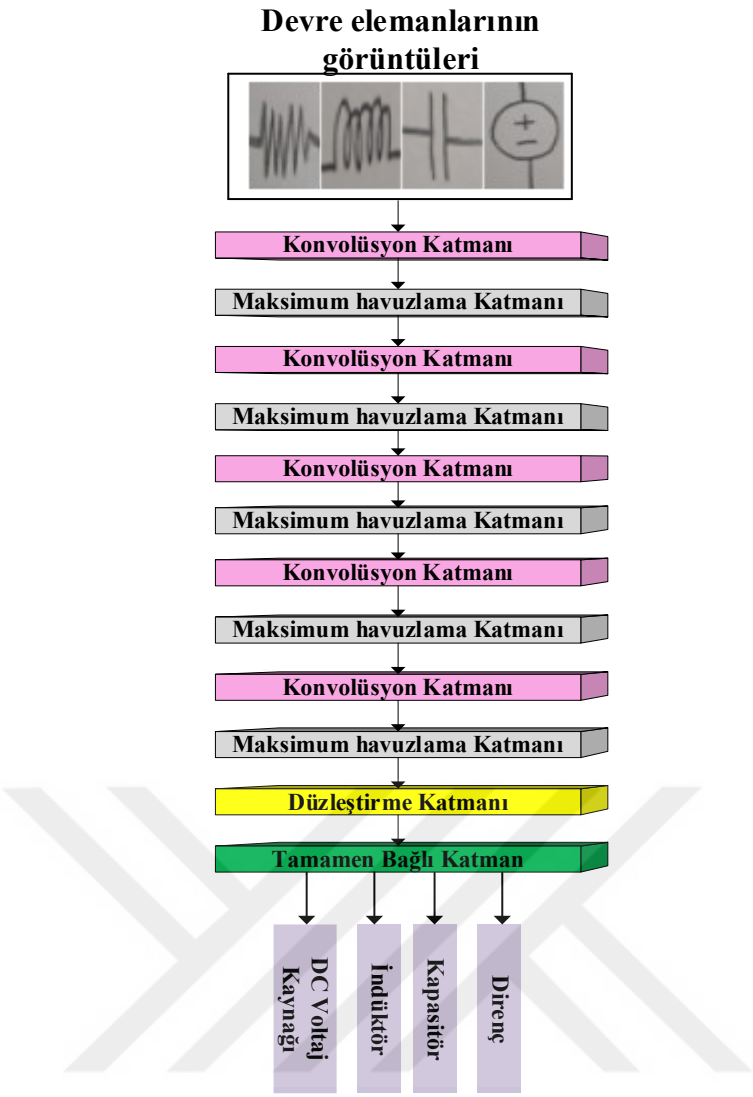
Renkli görüntülerin gri tonlamaya dönüştürülmesinin başlıca nedeni, bir görüntüyü tanımak ve yorumlamak için renk bilgisinin gerekli olmamasıdır. Renkli görüntüler üç kanalla temsil edildiği için gri tonlamalı resimlere göre daha fazla bilgi içerdiğinden işlemci belleğinde gereksiz karmaşıklık oluşturur ve bellekte daha fazla yer kaplar. İşlemci üzerindeki hesaplama karmaşıklığını azaltmak, hesaplama alanını azaltmak ve zamandan tasarruf etmek amacıyla veri setinde bulunan tüm görüntüler gri tonlamaya dönüştürülmüştür.

Çizelge 4.2 : Veri seti ile ilgili detaylar.

Veri setindeki Dosyalar	**Görüntülerin Boyutları**	**Eğitim (Train) veri setindeki görüntü adedi**	**Test veri setindeki görüntü adedi**
Direnç	150*150	282	32
İndüktör	150*150	170	65
Kapasitör	150*150	152	33
DC Voltaj Kaynağı	150*150	105	24

4.1.2 Çalışmada önerilen CNN modeli

CNN modeli tasarım sürecinde, dört farklı yeni CNN modeli geliştirilmiştir. Katman sayıları farklı olan modellerin arasından, Şekil 4.2'de verilen model seçilmiştir. Tercih edilen bu CNN modelinde, 12 katman bulunmaktadır. Model 5 adet konvolüsyon katmanı, 5 adet havuzlama katmanı, 1 adet düzleştirme katmanı, 1 adet tamamen bağlı katman ve softmax aktivasyon fonksiyonu ile tasarlanmıştır.

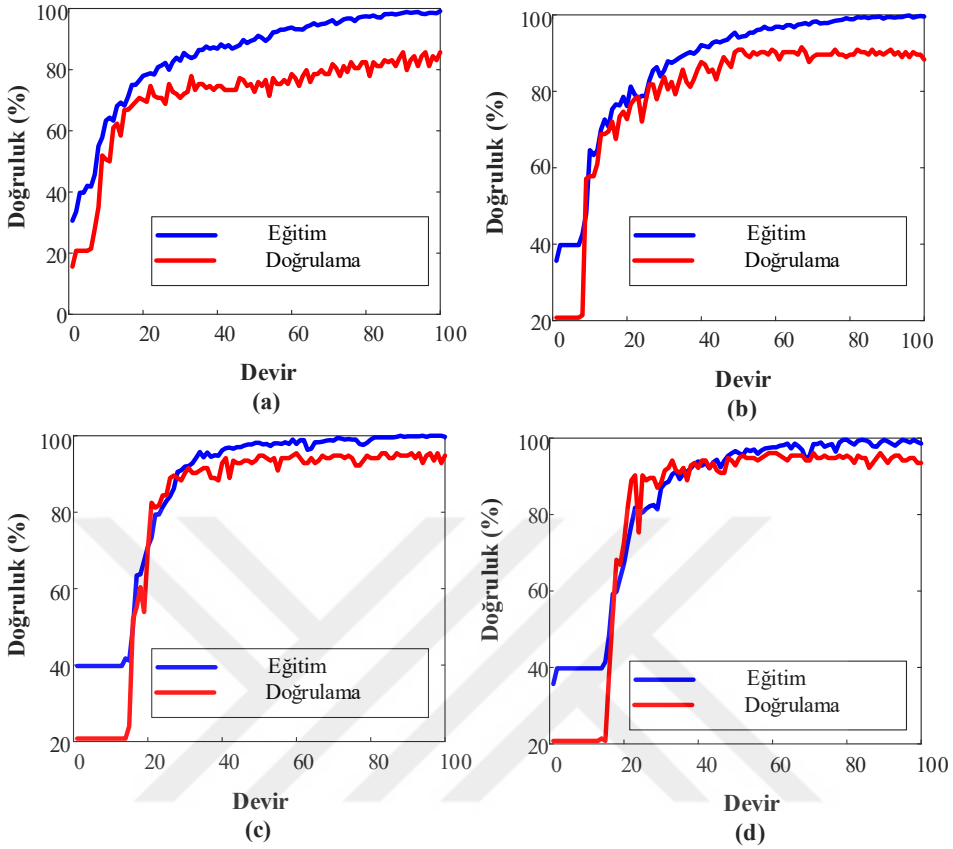


Şekil 4.2 : Önerilen CNN-4 modelinin mimarisi.

Konvolüsyon katmanında, görüntü verilerinin boyutuna göre filtre seçilir. 3*3 boyutlarında seçilen fitre, her bir görüntü verisi üzerinde soldan sağa ve yukarıdan aşağıya kaydırılır ve sırasıyla filtredeki sayısal değerler ile görüntüdeki piksel değerleri çarpılır. Çarpım sonucu çıkan değerler toplanır ve filtredeki toplam değer sayısına bölünerek elde edilen değer yeni oluşan görüntü üzerine yazılır. Konvolüsyon işlemi sonrası yeni oluşan görüntünün boyutu, ham görüntü boyutundan daha küçüktür. Maksimum havuzlama katmanında, boş bir filtre seçilir ve boyutu belirlenir. 2*2 boyutlarında seçilen boş filtre, görüntü üzerinde soldan sağa ve yukarıdan aşağıya hareket ettirilir, görüntü üzerinde dolaşan filtre içinde kalan 4 değerden en büyük değer alınarak, yeni oluşan görüntü üzerine yazılır. Düzleştirme katmanında, matris formatında olan görüntü verileri, düz bir dizi formatına dönüştürülür. Son katman olan tamamen bağlı katmanda ise sınıflandırma işlemi için softmax fonksiyonu kullanılır.

4.1.3 Deneysel sonuçlar

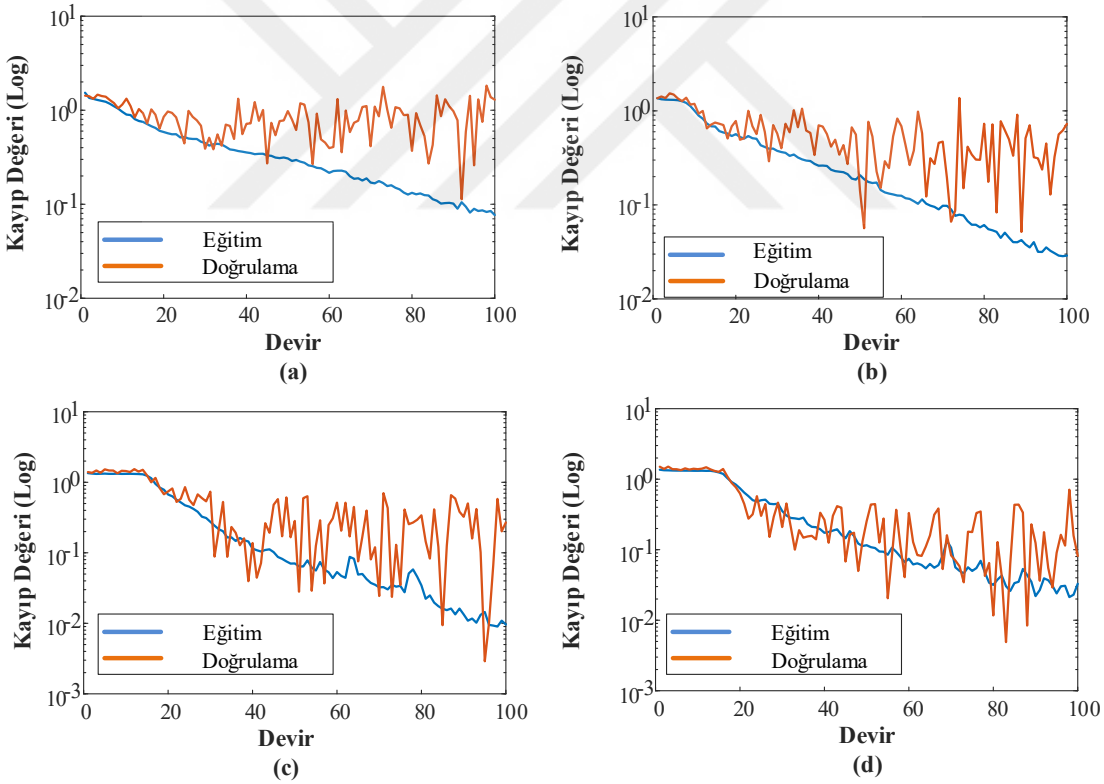
Çalışma için oluşturulan ve 863 adet görüntü içeren veri seti kullanılarak, katman sayıları birbirinden farklı 4 CNN modeli, 100 devir (epoch) için eğitilmiştir. Şekil 4.3'de katman sayıları farklı 4 ayrı CNN modeline ait doğruluk grafikleri verilmiştir.



Şekil 4.3 : Katman sayıları farklı CNN modellerine ait doğruluk grafikleri: (a) CNN-1, (b) CNN-2, (c) CNN-3, (d) CNN-4.

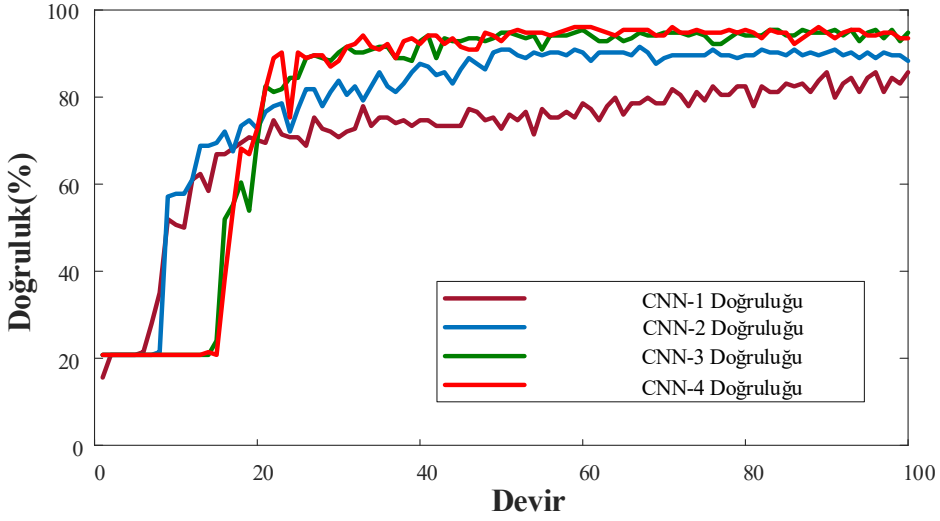
2 adet konvolüsyon katmanı, 2 adet maksimum havuzlama katmanı, 1 adet düzleştirme katmanı, 1 adet tamamen bağlı katman ve softmax aktivasyon fonksiyonu ile tasarlanan ilk CNN modeli olan CNN-1'e ait doğruluk grafiği Şekil 4.3 (a)'da, kayıp grafiği ise Şekil 4.4 (a)'da verilmiştir. 3 adet konvolüsyon katmanı, 3 adet maksimum havuzlama katmanı, 1 adet düzleştirme katmanı, 1 adet tamamen bağlı katman ve softmax aktivasyon fonksiyonu ile tasarlanan ikinci CNN modeli olan CNN-2'ye ait doğruluk grafiği Şekil 4.3 (b)'de, kayıp grafiği ise Şekil 4.4 (b)'de verilmiştir. 4 adet konvolüsyon katmanı, 4 adet maksimum havuzlama katmanı, 1 adet düzleştirme katmanı, 1 adet tamamen bağlı katman ve softmax aktivasyon fonksiyonu ile tasarlanan üçüncü CNN modeli olan CNN-3'e ait doğruluk grafiği Şekil 4.3 (c)'de, kayıp grafiği ise Şekil 4.4 (c)'de verilmiştir. 5 adet konvolüsyon katmanı, 5 adet maksimum havuzlama katmanı, 1 adet düzleştirme katmanı, 1

adet tamamen bağlı katman ve softmax aktivasyon fonksiyonu ile tasarlanan son CNN modeli olan CNN-4'e ait doğruluk grafiği Şekil 4.3 (d)'de, kayıp grafiği ise Şekil 4.4 (d)'de verilmiştir. Şekil 4.3 incelendiğinde, Şekil 4.3 (c) ve Şekil 4.3 (d) grafiklerindeki eğitim performansı diğer iki grafiğe göre daha yüksek çıkmıştır. Şekil 4.3 (c) ve Şekil 4.3 (d)'deki grafiklerin eğitim performansları birbirine yakın çıkmıştır. Grafiklerin doğrulama performansı göz önüne alındığında, Şekil 4.3 (d)'deki doğrulama performansı daha yüksek çıkmıştır. Şekil 4.4'de bulunan kayıp değeri grafikleri incelendiğinde Şekil 4.4 (d)'de kayıp grafiğindeki doğrulama performansı diğer üç grafiğe kıyasla en düşük çıkmıştır. CNN-4 modelinin diğer modellere kıyasla, doğruluk grafiğinde en yüksek performansı elde etmesinin sebebi, konvolüsyon ve havuzlama katmanı sayısının diğer modellere göre daha fazla olmasıdır. Benzer şekilde, kayıp değeri grafikleri incelendiğinde CNN-4 modeline ait kayıp değerinin diğer CNN modellerine kıyasla daha az olmasının sebebi, konvolüsyon ve havuzlama katmanı sayısının diğer modellere göre daha fazla olmasıdır. Grafikler değerlendirildiğinde, katman sayısı artırmanın CNN modellerindeki olumlu etkisi görülmektedir.

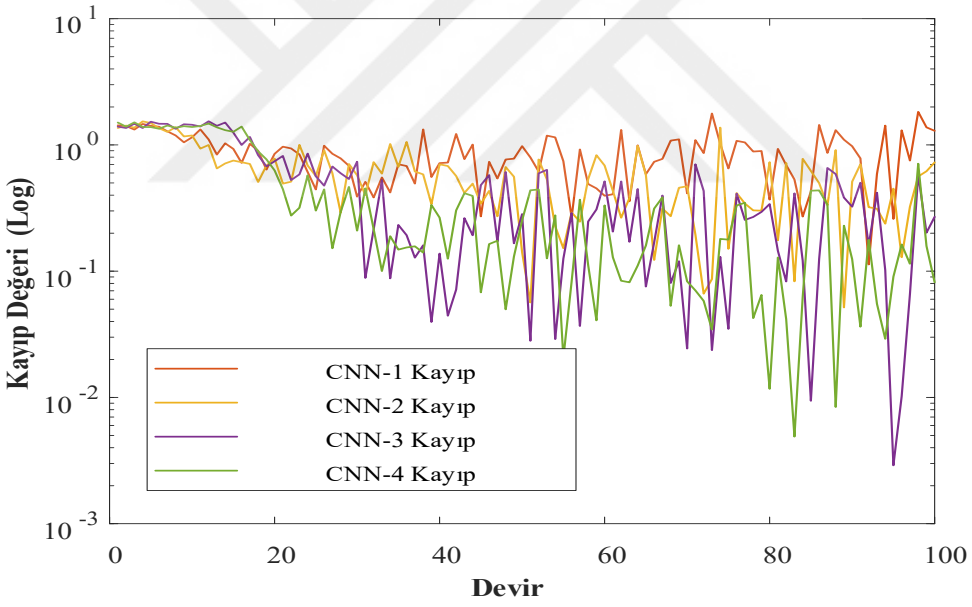


Şekil 4.4 : Katman sayıları farklı CNN modellerine ait kayıp değeri grafikleri: (a) CNN-1, (b) CNN-2, (c) CNN-3, (d) CNN-4.

Over fitting problemi, modelin eğitim aşamasında iyi bir performans gösterip doğrulama aşamasında, kötü performans sergilediği durumdur. Şekillerden görüldüğü gibi modellerin başarım grafiklerinde over fitting problemi oluşmamıştır.



Şekil 4.5 : Tüm CNN modellerinin doğrulama (validation) seti üzerindeki doğruluk oranlarını gösteren grafik.



Şekil 4.6 : Tüm CNN modellerinin doğrulama (validation) seti üzerindeki kayıp oranlarını gösteren grafik.

CNN-1, CNN-2, CNN-3 ve CNN-4 modellerinin son devirdeki doğruluk oranları sırasıyla %85.71, %88.31, %94.81, %93.51 elde edilmiştir. CNN-3 modelinin son devirdeki doğruluk oranı CNN-4 modelinden yüksektir. Fakat son devirdeki doğruluk oranı tam olarak modelin başarısını göstermez. CNN-4 modelinin başarım grafiği diğer modellere göre daha

yüksek çıkmıştır. Bu farkı gözlemlemek için tüm modellere ait doğrulama setleri üzerindeki performans grafikleri, Şekil 4.5 ve Şekil 4.6'da verilmiştir. Şekil 4.5'de görüldüğü gibi katman sayısı en fazla olan CNN-4 modelinin doğruluğu diğer 3 modele kıyasla en yüksek oranda çıkmıştır. Şekil 4.6 incelendiğinde, katman sayısı en fazla olan CNN-4 modelinin kaybı diğer 3 modele kıyasla en düşük oranda çıkmıştır. Eğitim aşamasında en iyi performansı gösteren CNN-4 modelinin her bir sınıf için sınıflandırmadaki başarısını gösteren karmaşıklık matrisi ise Şekil 4.7'de verilmiştir.

Doğruluk: 84.416%

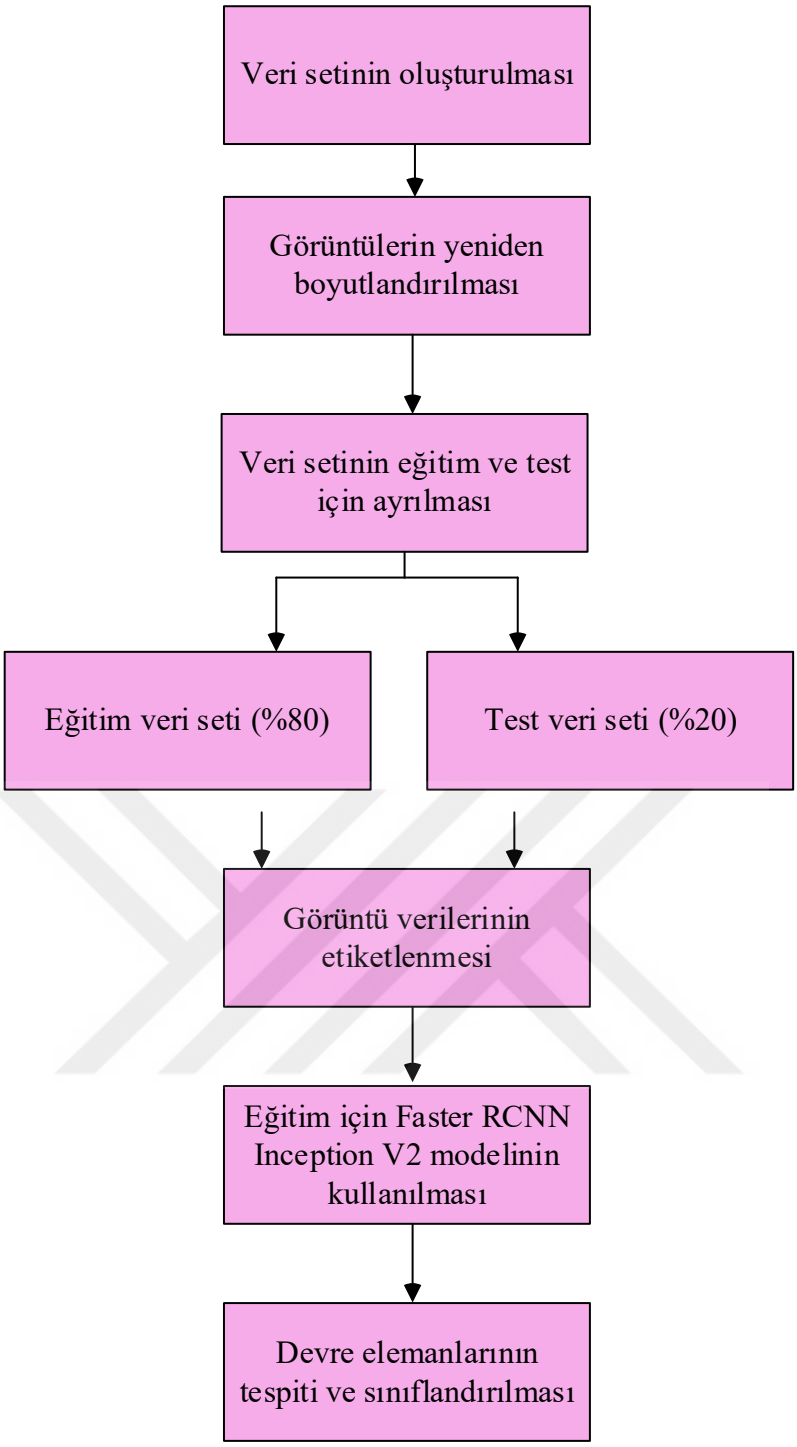
		Tahmin Edilen Sınıf			
		İndüktör	**Direnç**	**Kapasitör**	**DC Voltaj Kaynağı**
Gerçek Sınıf	**İndüktör**	100.0% 56	11.1% 4	12.2% 5	0.0% 0
	Direnç	0.0% 0	72.2% 26	14.6% 6	0.0% 0
	Kapasitör	0.0% 0	16.7% 6	65.9% 27	0.0% 0
	DC Voltaj Kaynağı	0.0% 0	0.0% 0	7.3% 3	100.0% 21

Şekil 4.7 : CNN-4 modeline ait karmaşıklık matrisi.

Buna göre CNN-4 modelinin sınıflandırmadaki genel başarısının %84.416 olduğu görülmüştür. Karmaşıklık matrisinden görüldüğü üzere modelin indüktör ve DC voltaj kaynağını sınıflandırma başarı oranı %100 dir. Modelin direnci doğru sınıflandırma başarısı %72.2 olurken kapasitörü doğru sınıflandırma başarısı %65.9 olmuştur.

4.2 Deneysel Çalışma II: Faster R-CNN Kullanarak El Çizimi Devre Üzerinde Bulunan Devre Elemanlarının Tespiti

Bu deneysel çalışmada, elle çizilmiş bir devre görüntüsünde bulunan devre elemanlarının tespiti için derin öğrenme tabanlı nesne tespiti için geliştirilmiş bir yöntem olan Faster R-CNN yöntemi önerilmiştir. Önerilen yöntem için çalışma adımları, Şekil 4.8'de verilmiştir.



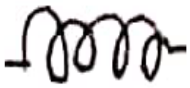
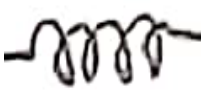
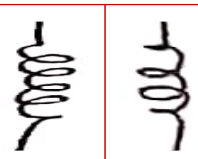
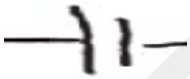
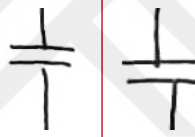
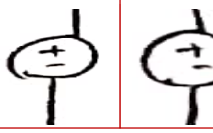
Şekil 4.8 : Elle çizilmiş devrelerde bulunan devre elemanlarının tanınmasında sırasıyla izlenen adımlar.

4.2.1 Veri seti ve ön işlemler

İlk olarak, elle çizilmiş devre görüntüleri toplanarak yeni bir veri seti oluşturulmuştur. İlk yapılan deneysel çalışmadaki veri seti kullanılmamıştır. Veri seti iki şekilde elde edilir. Birincisi, çeşitli web platformlarında hazır olarak bulunan veri setidir. İkincisi ise uygulamayı yapacak olan kişilerin kendilerinin oluşturdukları veri setidir. Bu

deneysel çalışma, elle çizilmiş devre görüntüleri üzerinde gerçekleştiğinden, elle çizilmiş devre görüntülerini içeren veri seti, herhangi bir web platformunda bulunamamıştır. Bu nedenle, veri seti manuel olarak farklı kişilerin elle çizdikleri devrelerin taranan görüntüleri ile oluşturulmuştur. Birçok kişinin farklı stillerle çizdiği elle çizilmiş devreler, beyaz bir A4 kağıdı üzerine çizilmiştir. Çalışmada kullanmak için 800 adet elle çizilmiş devrenin taranan görüntüsü toplanmıştır. Elle çizilmiş devre görüntülerinde tespit edilmesi gereken 4 farklı kategori belirlenmiştir. Bunlar; voltaj kaynağı, direnç, indüktör ve kapasitördür. Çizelge 4.3, veri setinde bulunan devre elemanlarından örnek görüntüler içermektedir.

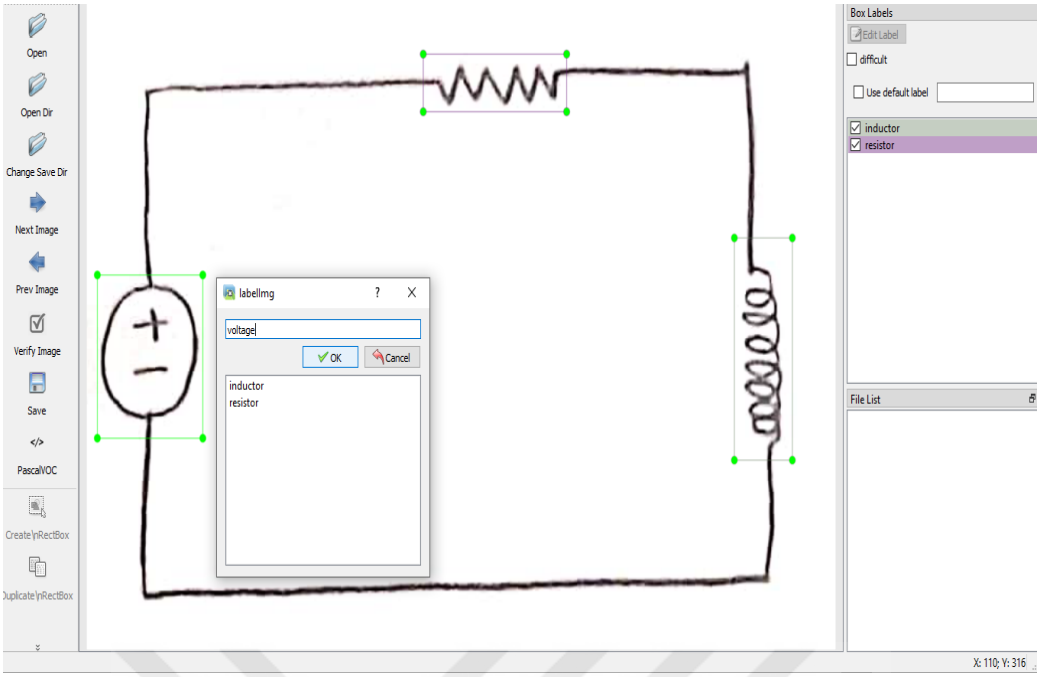
Çizelge 4.3 : Veri setinde bulunan sınıfların örnek görüntüleri.

SINIFLAR				
				İndüktör
				Direnç
				Kapasitör
				Voltaj

Veri seti oluşturduktan sonra belirli ön işlemler uygulanmıştır. Toplanan ve taranan bu görüntüler yüksek boyutlu olduğu için, öncelikle görüntüler yeniden boyutlandırılarak küçültülmüştür. Veri setinde bulunan tüm görüntüler 200*200 piksel olarak yeniden boyutlandırılmış ve tüm görüntüler aynı boyuta getirilmiştir. Oluşturulan veri seti ile eğitim ve test yapabilmek için iki ayrı dosya oluşturulur. Veri setinde bulunan görüntülerin %80'i train klasörüne geri kalan %20'si ise test klasörüne aktarılmıştır.

Çalışmadaki amaç görüntüler üzerindeki devre bileşenlerini tespit olduğu için sınıflandırma çalışmalarından farklı olarak devre bileşenlerinin konumuna ihtiyaç vardır. Bu sebeple, label map programı kullanılarak veri setindeki görüntülerde bulunan devre

bileşenleri etiketlenmiştir. Şekil 4.9, LabelImg programı ile veri setinde bulunan örnek bir görüntüdeki elemanların etiketleme işlemini göstermektedir.



Şekil 4.9 : LabelImg ile veri setindeki görüntüleri etiketleme.

LabelImg programı ile veri setinde bulunan tüm görüntüler etiketlenmiştir. Train ve test klasöründe yer alan her bir görüntü için xml dosyası oluşturulur. Xml dosyası, nesnelerin sınıf bilgilerini ve koordinat bilgilerini içerir. Train ve test klasöründeki tüm xml dosyaları tek bir csv dosyasına dönüştürülür. Uygulama için gerekli ön hazırlıklar tamamlandıktan sonra Faster R-CNN modelini kullanmak için Google Colab [115] bulut ortamı tercih edilmiştir.

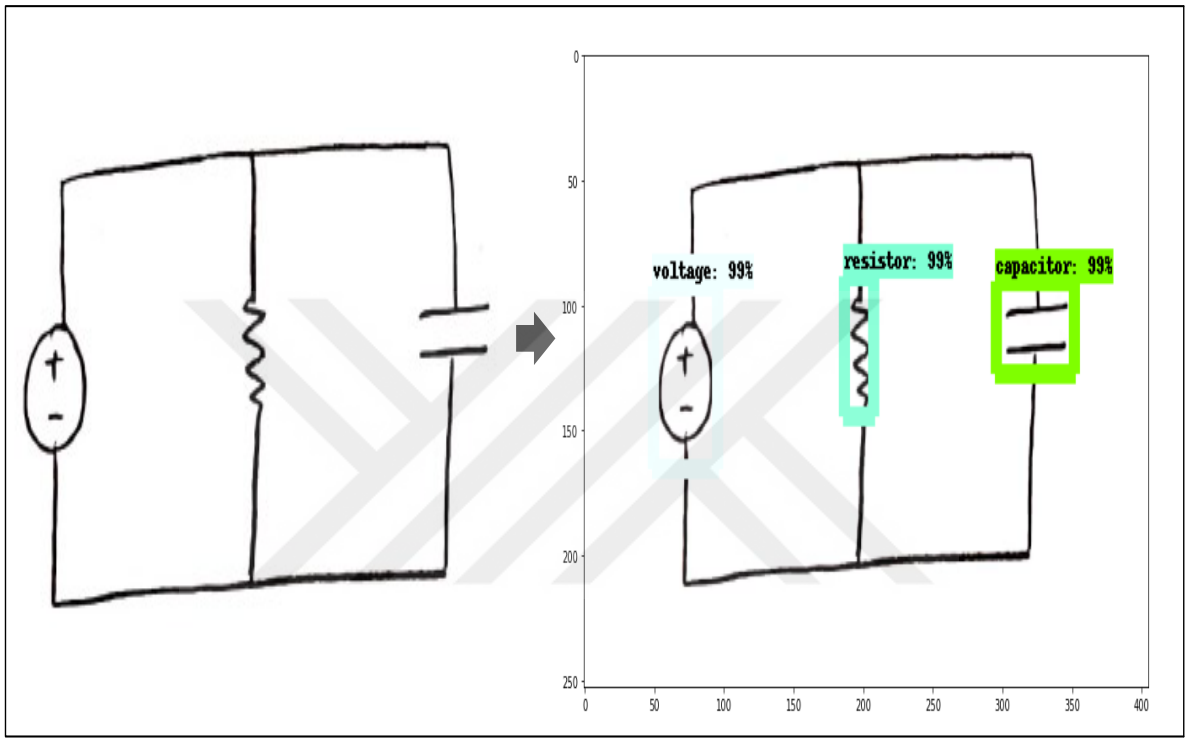
4.2.2 Önerilen Faster R-CNN modeli ile devre elemanlarının tespiti

Bu çalışmada, el çizimi devreler üzerindeki devre elemanlarını tespit etmek için önceden eğitilmiş Faster R-CNN modellerinden olan Inception V2 modeli [116] tercih edilmiştir. Önerilen el çizimi devre elemanı tanıma sisteminin uygulaması, python yazılım dili kullanılarak önceden eğitilmiş Inception V2 model üzerinde gerekli ayarlamalar ve düzenlemeler yapılarak eğitim gerçekleşmiştir. Ekran kartı olarak Google Colab'ın ücretsiz sunduğu NVDIA Tesla K80 GPU kullanılmıştır. Model, 50000 adımda eğitilmiştir.

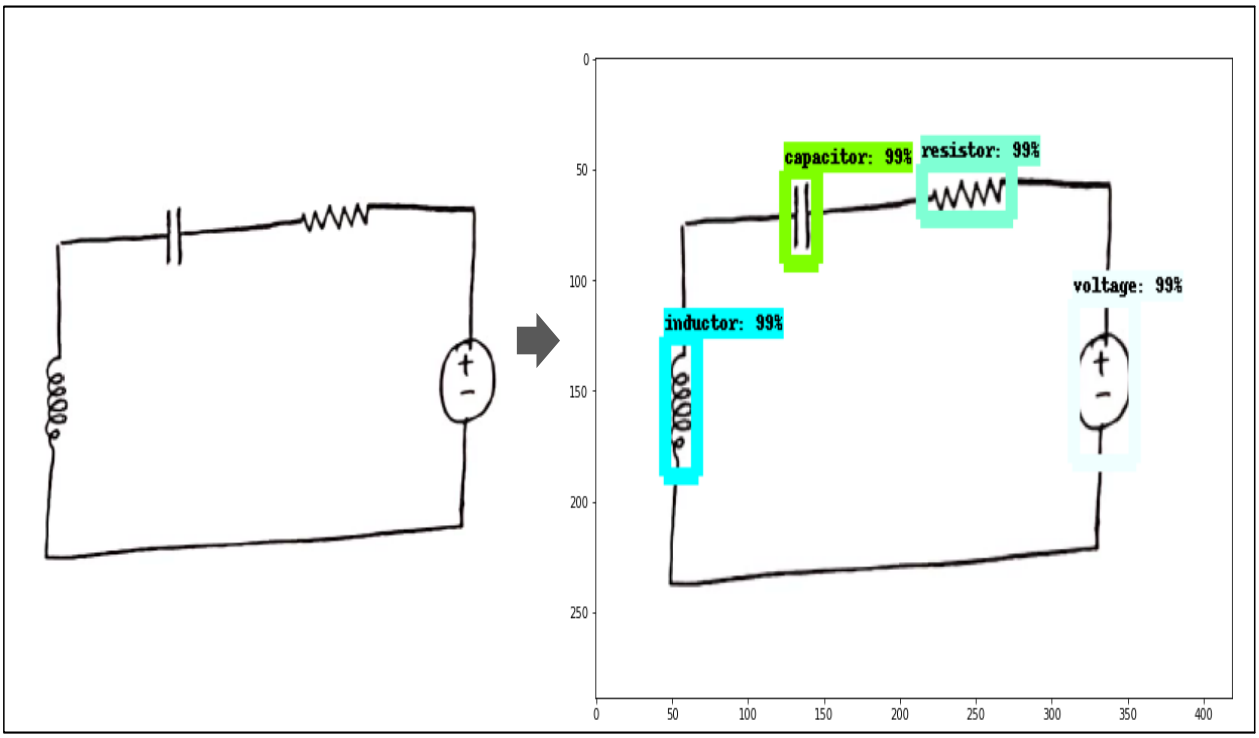
Farklı stillerle çizilmiş 4 farklı kategorideki devre elemanlarının devre üzerinde bulunduğu konumlar değişebilir. Ayrıca, devrenin amacına göre her devrede bu 4 eleman bulunmayabilir. Çünkü tasarlanan her bir devrenin ve devrede bulunan her bir elemanın

gerçekleştireceği görevi vardır. Devre elemanlarının devre üzerinde farklı konumlarda bulunması durumu göz önüne alınarak, eğitilen model elle çizilmiş çeşitli devreler üzerinde analiz edilmiştir.

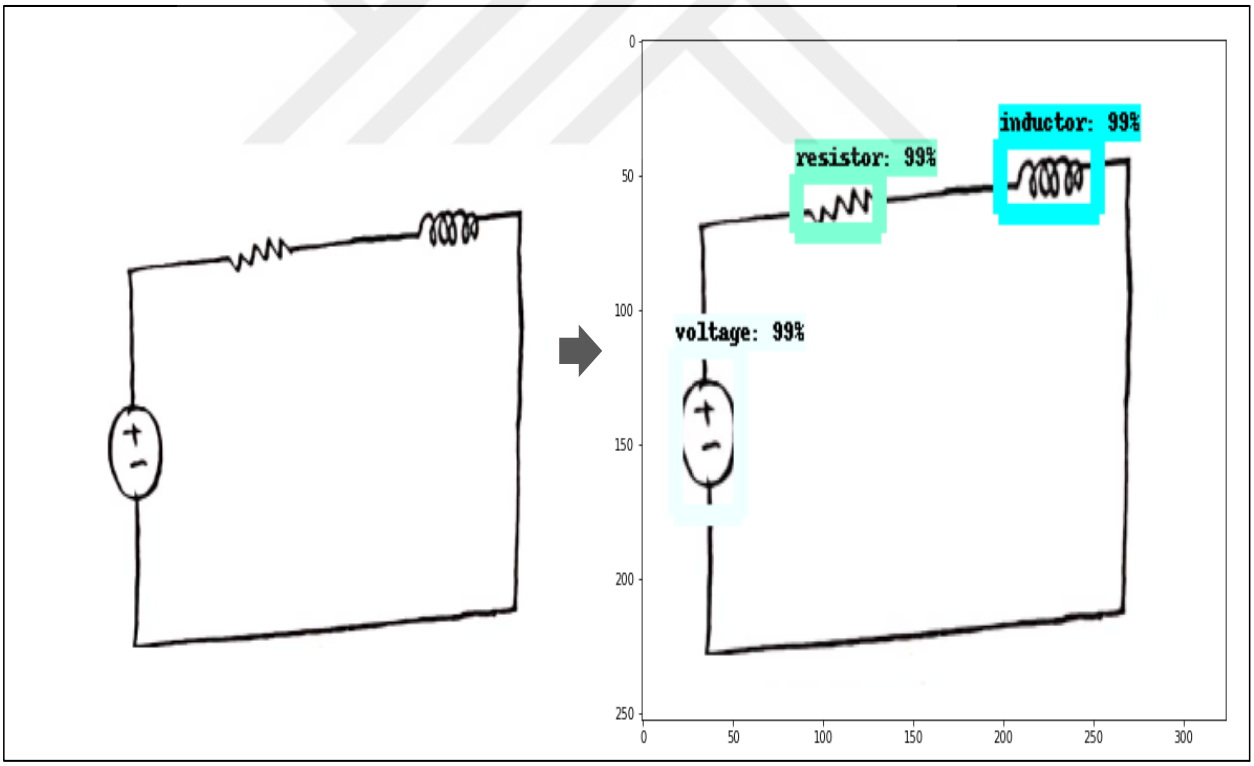
Şekil 4.10, Şekil 4.11, Şekil 4.12 ve Şekil 4.13, 4 farklı elle çizilmiş devre görüntüsü üzerinde bulunan devre elemanlarının gerçek zamanlı tespitini göstermektedir. Aşağıdaki şekiller incelendiğinde eğitilen model, devreler üzerinde bulunan elemanları değişik renk tonları ile dikdörtgen çerçeve çizerek tespit etmiştir. Ayrıca model, çizilen her bir dikdörtgen çerçeve ile birlikte tespit edilen elemanın etiketini % olarak tahmin etmiştir.



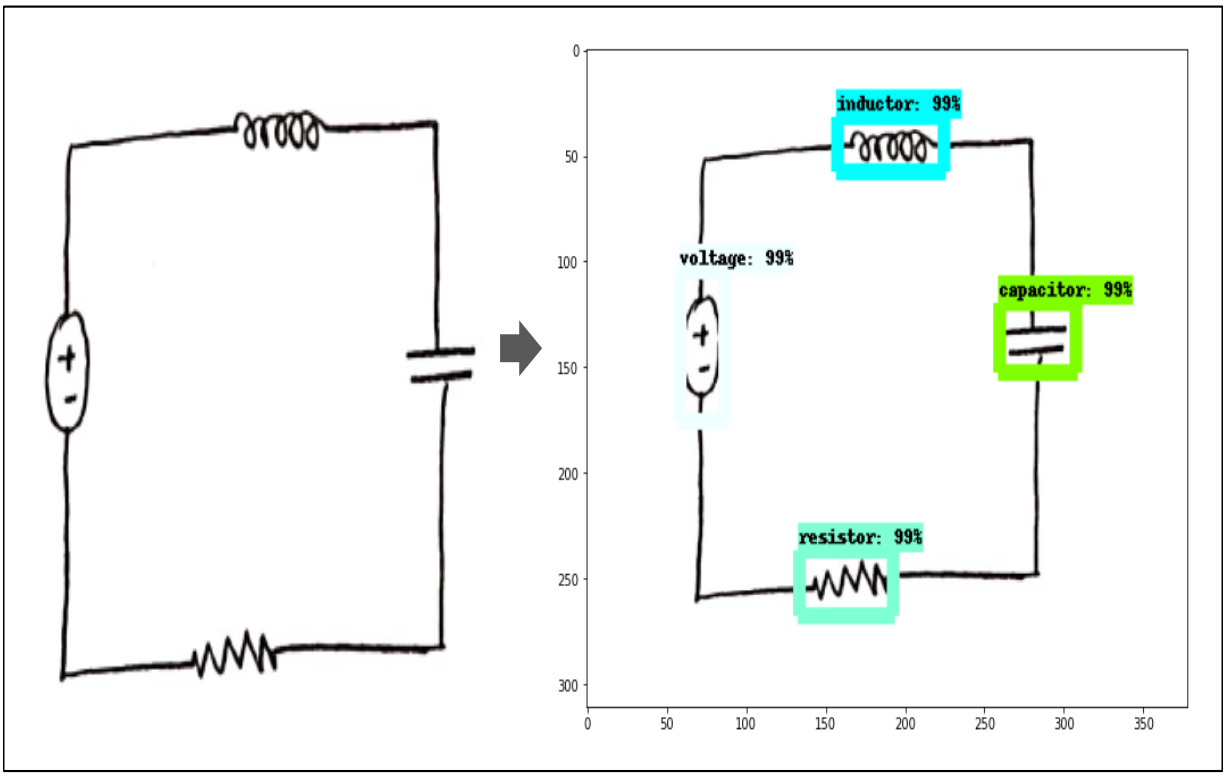
Şekil 4.10 : Faster R-CNN ile örnek bir devrede gerçek zamanlı devre elemanlarının tespiti test-1.



Şekil 4.11 : Faster R-CNN ile örnek bir devrede gerçek zamanlı devre elemanlarının tespiti test-2.



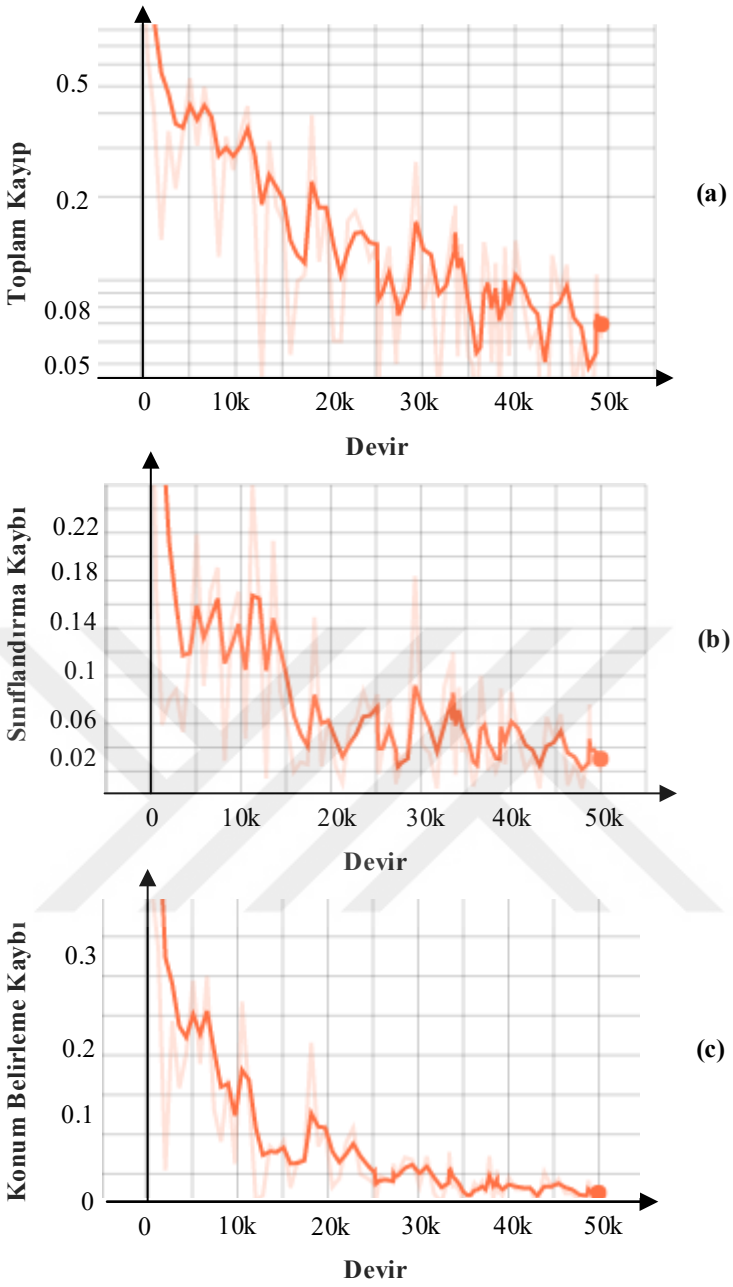
Şekil 4.12 : Faster R-CNN ile örnek bir devrede gerçek zamanlı devre elemanlarının tespiti test-3.



Şekil 4.13 : Faster R-CNN ile örnek bir devrede gerçek zamanlı devre elemanlarının tespiti test-4.

Eğitilen Faster R-CNN Inception V2 modelinin elle çizilmiş devreler üzerinde bulunan elemanları tanıyabilmesindeki performansını değerlendirmek için çalışmada kayıp grafikleri incelenmiştir. Kayıp grafiklerinde, kayıp ne kadar azalırsa modelin başarısı o kadar artar. Bundan dolayı modelin başarısını değerlendirirken adım sayısı arttıkça kaybın kademeli olarak azalması beklenir. Şekil 4.14'de 50000 devir eğitilen elle çizilmiş devrede bulunan elemanları tespit eden modele ait çeşitli kayıp fonksiyonları verilmiştir. Şekil 4.14 (a)'da modele ait toplam kayıp fonksiyonu verilmiştir. Toplam kayıp fonksiyonu, modelin genel kaybını göstermektedir. Şekil 4.14 (a) incelendiğinde, 50000 adım sonunda toplam kayıp 0.06 'ya düşmüştür. Şekil 4.14 (b)'de modele ait sınıflandırma kayıp fonksiyonu verilmiştir. Sınıflandırma kayıp fonksiyonu, modelin sınıflandırma aşamasındaki kaybını göstermektedir. Şekil 4.14 (b) incelendiğinde, 50000 adım sonunda sınıflandırmadaki kayıp 0.03 'e düşmüştür. Şekil 4.14 (c)'de, modelin devre elemanlarının konumlarını belirleyebilmesine ait kayıp fonksiyonu verilmiştir. Şekil 4.14 (c) incelendiğinde, 50000 adım sonunda konum belirleme kaybı 0.02 'ye düşmüştür. Modelin başarısını gösteren kayıp grafikleri incelendiğinde, kayıpta kademeli olarak azalma görülmektedir ve bir süre sonra kayıp sabit kalmaya başlamıştır. Toplam kayıp grafiği göz önüne alındığında, model kaybının minimum 0.048 olmuştur. Devir sayısı arttıkça modelin kaybının 0.1'in altında

kalması, modelin gerçek zamanlı tespitlerde yüksek başarım oranına sahip olacağını göstermektedir.



Şekil 4.14 : Test edilen farklı devre görüntülerindeki elemanların gerçek zamanlı tespiti: (a) Toplam kayıp, (b) Sınıflandırma kaybı, (c) Konum belirleme kaybı.

5. SONUÇLAR

Bu tez çalışmasının deneysel çalışmaları, iki aşamada gerçekleşmiştir. Yapılan ilk deneysel çalışmada, popüler araştırma konularından biri olan derin öğrenme mimarilerinden evrişimsel sinir ağları mimarisi kullanılarak, veri setindeki görüntülerin eğitimi gerçekleştirilmiş ve hangi görüntünün hangi elektronik devre elemanı sınıfına ait olduğu yüksek başarı oranıyla belirlenmiştir. Çalışmada, direnç, indüktör, kapasitör ve DC voltaj kaynağı görüntülerini doğru sınıflara atamak için 4 faklı CNN modeli eğitilmiştir. Bu modeller yazarlar tarafından oluşturulan özel veri seti üzerinde test edilerek sonuçlar incelenmiştir. Bu modellerin eğitim ve doğrulama sonuçları ayrı ayrı karşılaştırılmıştır. Eğitilmiş modeller test edildiğinde, en yüksek doğruluk elde eden modelde konvolüsyon katmanı ve havuzlama katmanı sayısı diğer modellere göre daha fazladır. Bu sebeple, CNN'de konvolüsyon katmanı ve havuzlama katmanı sayısını artırmanın, modelin performansı üzerindeki olumlu etkisi görülmüştür.

Yapılan ikinci deneysel çalışmada, ilk deneysel çalışmadan farklı olarak elle çizilmiş devreler üzerinde bulunan devre elemanlarının hem tespiti hem de sınıflandırılması işlemi gerçekleştirilmiştir. Elle çizilen devrelerin taranan görüntüleri üzerinde farklı konumlarda bulunan devre elemanlarının tanınması için, son yıllarda gelişmekte olan ve özellik çıkarma aşaması CNN tarafından otomatik olarak yapılan Faster R-CNN nesne algılama yöntemi, kullanılmıştır. Önerilen yöntemi kullanmak için ilk olarak, farklı kişilerin el çizimleri olan 800 adet devre görüntüsü toplanılarak veri seti oluşturulmuştur. Veri setinde bulunan görüntülerin etiketlenmesi ve gerekli diğer ön işlemler tamamlandıktan sonra, önceden eğitilmiş Faster R-CNN Inception V2 modelinde ince ayar ve düzenlemeler yapılarak kullanılmıştır. Gerekli düzenlemeler yapıldıktan sonra model 50000 adımda eğitilmiştir. Eğitilen Faster R-CNN modeli, devre elemanlarının farklı konumlarda dizayn edildiği birçok elle çizilmiş devre üzerinde test edilmiştir. Elde edilen sonuçlara göre kullanılan yöntemin, elle çizilmiş devre elemanlarının seri ve paralel RLC devrelerinden ayrı ayrı tespit edilerek tanımlanmasında yüksek başarı oranına sahip olduğu görülmüştür.

KAYNAKLAR

[1] **Liu, Y., & Xiao, Y.** (2013). *Circuit Sketch Recognition*. Retrieved March 15, 2020, from https://stacks.stanford.edu/file/druid:bf950qp8995/Liu_Xiao.pdf.

[2] **Moetesum, M., Waqar Younus, S., Ali Warsi, M., & Siddiqi, I.** (2018). Segmentation and Recognition of Electronic Components in Hand-Drawn Circuit Diagrams. *ICST Transactions on Scalable Information Systems*, *5*(16), 154478.

[3] **Angadi, M., & R, L. N.** (2014). Handwritten Circuit Schematic Detection and Simulation using Computer Vision Approach. *3*(6), 754–761.

[4] **Dewangan, A., & Dhole, A.** (2018). KNN based Hand Drawn Electrical Circuit Recognition. *International Journal for Research in Applied Science and Engineering Technology(IJRASET)*, *6*(6), 1111–1115.

[5] **Lakshman Naika, R., Dinesh, R., & Santoshnaik.** (2018). Handwritten electronic components recognition: An approach based on HOG + SVM. *Journal of Theoretical and Applied Information Technology*, *96*(13), 4020–4028.

[6] **Patare, M. D., & Joshi, M. S.** (2016). Hand-drawn Digital Logic Circuit Component Recognition using SVM. *International Journal of Computer Applications*, *143*(3), 24–28.

[7] **Datta, R., Mandal, P. D. S., & Chanda, B.** (2016). Detection and identification of logic gates from document images using mathematical morphology. *2015 5th National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics, NCVPRIPG 2015*, 19–22. https://doi.org/10.1109/NCVPRIPG.2015.7490040.

[8] **Sala, P.** (2004). A recognition system for symbols of electronic components in hand-written circuit diagrams. *CSC 2515-Machine Learning Project Report*.

[9] **Rabbani, M., Khoshkangini, R., Nagendraswamy, H. S., & Conti, M.** (2016). Hand Drawn Optical Circuit Recognition. *Procedia Computer Science*, *84*, 41–48. https://doi.org/10.1016/j.procs.2016.04.064.

[10] **Edwards, B., & Chandran, V.** (2000). Machine recognition of hand-drawn circuit diagrams. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *6*, 3618–3621. https://doi.org/10.1109/ICASSP.2000.860185.

[11] **Lee, H., & Kwon, H.** (2017). Going Deeper with Contextual CNN for Hyperspectral Image Classification. *IEEE Transactions on Image Processing*, *26*(10), 4843–4855. https://doi.org/10.1109/TIP.2017.2725580.

[12] **Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W.** (2016). CNN-RNN: A Unified Framework for Multi-label Image Classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 2285–2294. https://doi.org/10.1109/CVPR.2016.251.

[13] **Li, Q., Cai, W., Wang, X., Zhou, Y., Feng, D. D., & Chen, M.** (2014). Medical image classification with convolutional neural network. *2014 13th International Conference on Control Automation Robotics and Vision, ICARCV 2014*,

2014(December), 844–848. https://doi.org/10.1109/ICARCV.2014.7064414.

[14] **Wang, H., Pan, T., & Ahsan, M. K.** (2020). Hand-drawn electronic component recognition using deep learning algorithm. *International Journal of Computer Applications in Technology*, *62*(1), 13–19.

[15] **Jain, A., Singh, S. K., & Singh, K. P.** (2020). Handwritten signature verification using shallow convolutional neural network. *Multimedia Tools and Applications*, *79*(27–28), 19993–20018. https://doi.org/10.1007/s11042-020-08728-6.

[16] **Boufenar, C., Kerboua, A., & Batouche, M.** (2018). Investigation on deep learning for off-line handwritten Arabic character recognition. *Cognitive Systems Research*, *50*, 180–195. https://doi.org/10.1016/j.cogsys.2017.11.002.

[17] **Ahlawat, S., Choudhary, A., Nayyar, A., Singh, S., & Yoon, B.** (2020). Improved handwritten digit recognition using convolutional neural networks (Cnn). *Sensors (Switzerland)*, *20*(12), 1–18. https://doi.org/10.3390/s20123344.

[18] **Wu, Y. C., Yin, F., & Liu, C. L.** (2017). Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. *Pattern Recognition*, *65*(December 2016), 251–264. https://doi.org/10.1016/j.patcog.2016.12.026.

[19] **Sardoğan, M., Özen, Y., & Tuncer, A.** (2020). Faster R-CNN Kullanarak Elma Yaprağı Hastalıklarının Tespiti. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, *8*, 1110–1117. https://doi.org/10.29130/dubited.648387.

[20] **Cömert, O., Hekim, M., & Adem, K.** (2019). Faster R-CNN Kullanarak Elmalarda Çürük Tespiti. *Uluslararası Muhendislik Arastirma ve Gelistirme Dergisi*, *11*(1), 335–341. https://doi.org/10.29137/umagd.469929.

[21] **Ren, Y., Zhu, C., & Xiao, S.** (2018). Deformable faster R-CNN with aggregating multi-layer features for partially occluded object detection in optical remote sensing images. *Remote Sensing*, *10*(9). https://doi.org/10.3390/rs10091470.

[22] **Julca-Aguilar, F. D., & Hirata, N. S. T.** (2018). Symbol detection in online handwritten graphics using faster R-CNN. *Proceedings - 13th IAPR International Workshop on Document Analysis Systems, DAS 2018*, 151–156. https://doi.org/10.1109/DAS.2018.79.

[23] **Yang, J., Ren, P., & Kong, X.** (2019). Handwriting Text Recognition Based on Faster R-CNN. *Proceedings - 2019 Chinese Automation Congress, CAC 2019*, 2450–2454. https://doi.org/10.1109/CAC48633.2019.8997382.

[24] **Albahli, S., Nawaz, M., Javed, A., & Irtaza, A.** (2021). An improved faster-RCNN model for handwritten character recognition. *Arabian Journal for Science and Engineering*, *0123456789*. https://doi.org/10.1007/s13369-021-05471-4.

[25] **Cao, C., Wang, B., Zhang, W., Zeng, X., Yan, X., Feng, Z., Liu, Y., & Wu, Z.** (2019). An Improved Faster R-CNN for Small Object Detection. *IEEE Access*, *7*(August),106838–106846.https://doi.org/10.1109/ACCESS.2019.2932731.

[26] **Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E.** (2018). Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience*, *2018*. https://doi.org/10.1155/2018/7068349.

[27] **V., D. S.** (2019). Computer Vision for Human-Machine Interaction-Review. *Journal of*

Trends in Computer Science and Smart Technology, *2019*(02), 131–139. https://doi.org/10.36548/jtcsst.2019.2.006.

[28] **Li, G., Huang, Y., Chen, Z., Chesser, G. D., Purswell, J. L., Linhoss, J., & Zhao, Y.** (2021). Practices and applications of convolutional neural network-based computer vision systems in animal farming: A review. *Sensors*, *21*(4), 1–42. https://doi.org/10.3390/s21041492.

[29] **Petrou, M., & Kamata, S.** (2021). Image processing: dealing with texture. *John Wiley & Sons*. https://doi.org/10.1002/9781119618621.

[30] **Nixon, M., & Aguado, A. S.** (2012). Feature Extraction & Image Processing for Computer Vision. In *Feature Extraction & Image Processing for Computer Vision*. Academic Press. https://doi.org/10.1016/C2011-0-06935-1.

[31] **Reiter, A., Allen, P. K., & Zhao, T.** (2012). Feature classification for tracking articulated surgical tools. *International Conference on Medical Image Computing and Computer-Assisted Intervention.*, 592–600.

[32] **Samuel, A. L.** (1959). Some Studies in Machine Learning. *IIBM Journal of Research and Development*, *3*(3), 210–229.

[33]**Url-33**<*https://emerj.com/ai-glossary-terms/what-is-machine-learning/*>,date retrieved 12.01.2021.

[34] **Rameez, A., Nizzad, M., & Kunarakulan, K.** (2020). Training Computers to Recognize Facial Expressions of Humans : A Machine Learning Approach. *5*(1), 1–13.

[35] **Berg, A., & Forsberg, D.** (2019). Identifying DNS-tunneled traffic with predictive models. *ArXiv*, 1–14.

[36]**Url-36**<*https://towardsdatascience.com/machine-learning-vs-deep-learning 62137a1c9842*>, date retrieved 07.06.2021.

[37] **Kingma, D. P., Rezende, D. J., Mohamed, S., & Welling, M.** (2014). Semi-supervised learning with deep generative models. *Advances in Neural Information Processing Systems*, *4*(January), 3581–3589.

[38] **Joachims, T.** (2000). Transductive inference for text classification using support vector machines. *Proceedings of the 20th International Conference on Machine Learning*.

[39] **Şahin, E.** (2018). *Makine Öğrenme Yöntemleri Ve Kelime Kümesi Tekniği İle İstenmeyen E-Posta / E-Posta Sınıflaması*. (Master's thesis). Retrieved from https://tez.yok.gov.tr/UlusalTezMerkezi/.

[40]**Url-40**<*https://www.geeksforgeeks.org/ml-classification-vs-regression/*>, date retrieved 05.06.2021.

[41] **Goldberg, X.** (2009). Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *6*, 1–116. https://doi.org/10.2200/S00196ED1V01Y200906AIM006.

[42] **Wei, Q.** (2018). Understanding of the naive Bayes classifier in spam filtering. *AIP Conference Proceedings*, *1967*(May). https://doi.org/10.1063/1.5038979.

[43] **Url-43**<*https://machinelearningmastery.com/bayes-theorem-for-machine-learning/*>, date retrieved 01.06.2021.

[44]**Url-44**<*https://medium.com/kaveai/naive-bayes-ve-uygulamaları-d7d5a56c689b*>, date retrieved 01.06.2021.

[45] **Hu, W.** (2018). Towards a Real Quantum Neuron. *Natural Science*, *10*(03), 99–109. https://doi.org/10.4236/ns.2018.103011.

[46] **Ceylan, H.** (2014). An artificial neural networks approach to estimate occupational accident: A national perspective for turkey. *Mathematical Problems in Engineering*, *2014*. https://doi.org/10.1155/2014/756326.

[47] **Url-47** <*https://learnopencv.com/understanding-feedforward-neural-networks/*>, date retrieved 31.05.2021.

[48] **Url-48** <*https://towardsdatascience.com/a-gentle-introduction-to-neural-networks-series-part-1-2b90b87795b*>, date retrieved 12.01.2021.

[49] **Kotsiantis, S. B.** (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, *31*, 249–268. https://doi.org/10.1007/s10751-016-1232-6.

[50] **Baharudin, B., Lee, L. H., & Khan, K.** (2010). A Review of Machine Learning Algorithms for Text-Documents Classification. *Journal of Advances in Information Technology*, *1*(1), 4–20. https://doi.org/10.4304/jait.1.1.4-20.

[51] **Foody, G. M., & Mathur, A.** (2004). Toward intelligent training of supervised image classifications: Directing training data acquisition for SVM classification. *Remote Sensing of Environment*, *93*(1–2), 107–117.

[52] **Cortes, C., & Vapnik, V.** (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297.

[53] **Kavzoglu, T., & Colkesen, I.** (2010). Destek Vektör Makineleri ile Uydu Görüntülerinin Sınıflandırılmasında Kernel Fonksiyonlarının Etkilerinin İncelenmesi. *Harita Dergisi*, *144*, 73–82.

[54] **Osuna, E., Freund, R., & Girosi, F.** (1997). Support Vector Machines : Training and Applications. *Massachusetts Institute of Technology*, *9217041*(1602).

[55] **Ganyun, L. V., Haozhong, C., Haibao, Z., & Lixin, D.** (2005). Fault diagnosis of power transformer based on multi-layer SVM classifier. *Electric Power Systems Research*, *74*(1), 1–7.

[56] **Ramteke, R. J., & Y, K. M.** (2012). Automatic Medical Image Classification and Abnormality Detection Using K- Nearest Neighbour. *International Journal of Advanced Computer Research*, *2*(4), 190–196.

[57] **Li, Y., & Cheng, B.** (2009). An improved k-nearest neighbor algorithm and its application to high resolution remote sensing image classification. *2009 17th International Conference on Geoinformatics, Geoinformatics 2009*, 1–4. https://doi.org/10.1109/GEOINFORMATICS.2009.5293389.

[58] **Jiang, L., Cai, Z., Wang, D., & Jiang, S.** (2007). Survey of improving K-nearest-neighbor for classification. *Proceedings - Fourth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2007*, *1*, 679–683. https://doi.org/10.1109/FSKD.2007.552.

[59]**Url-59**<*https://www.datacamp.com/community/tutorials/k-nearest-neighbor classification-scikit-learn*>, date retrieved 31.05.2021.

[60] **Jadhav, S. D., & Channe, H. P.** (2016). Comparative Study of K-NN, Naive Bayes and Decision Tree Classification Techniques. *International Journal of Science and Research (IJSR)*, *5*(1), 1842–1845.

[61] **Ho, Y., Kyeong, J., & Hie, S.** (2002). A personalized recommender system based on web usage mining and decision tree induction. *Expert systems with Applications*, *23*(3) 329–342.

[62] **Safavian, S. R., & Landgrebe, D.** (1991). A Survey of Decision Tree Classifier Methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, *21*(3), 660–674.

[63] **Arnold, T.** (2014). *A Machine Learning Approach for Coreference Resolution* (Master's thesis). Retrieved from https://www.ke.tu-darmstadt.de/lehre/arbeiten/master/2014/Arnold_Thomas.pdf.

[64] **Ruggieri, S.** (2002). Efficient C4. 5. *IEEE Transactions on Knowledge and Data Engineering*, *14*(2), 438–444.

[65] **Yu, X., Yang, J., Lin, Z., Wang, J., & Wang, T.** (2015). Subcategory-Aware Object Detection. *IEEE Signal Processing Letters*, *22*(9), 1472–1476.

[66] **Url-66**<*https://machinelearningmastery.com/what-is-deep-learning/*>, date retrieved 15.01.2021.

[67] **Lecun, Y., Bengio, Y., & Hinton, G.** (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539.

[68] **Schmidhuber, J.** (2015). Deep Learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003.

[69] **Musiol, M. J.** (2016). *Speeding up Deep Learning Computational Aspects of Machine Learning Speeding up Deep Learning Computational Aspects of Machine Learning*. *January*.

[70] **Fukushima, K.** (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*(4), 193–202. https://doi.org/10.1007/BF00344251.

[71] **Cun, Y. Le, Guyon, I., Jackel, L. D., Henderson, D., Boser, B., Howard, R. E., Denker, J. S., Hubbard, W., & Graf, H. P.** (1989). Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning. *IEEE Communications Magazine*, *27*(11), 41–46. https://doi.org/10.1109/35.41400.

[72] **Waibel, A., & Lee, K.-F.** (1990). Readings in Speech Recognition. In *Readings in Speech Recognition*. Elsevier. https://doi.org/10.1016/c2009-0-27652-5.

[73] **Maitra, D. Sen, Bhattacharya, U., & Parui, S. K.** (2015). CNN based common approach to handwritten character recognition of multiple scripts. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, *2015-Novem*, 1021–1025. https://doi.org/10.1109/ICDAR.2015.7333916.

[74] **Chen, L., Wang, S., Fan, W., Sun, J., & Naoi, S.** (2016). Beyond human recognition: A CNN-based framework for handwritten character recognition. *Proceedings - 3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015*, 695–699. https://doi.org/10.1109/ACPR.2015.7486592.

[75] **Lawrence, S., Giles, C. L., Tsoi, A. C., & Back, A. D.** (1997). Face Recognition: A Convolutional Neural-Network Approach. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, *8*(1), 98–113.

[76] **Khalajzadeh, H., Mansouri, M., & Teshnehlab, M.** (2014). Face Recognition Using Convolutional Neural Network and Simple Logistic Classifier. *Soft Computing in Industrial Applications. Springer International Publishing*, *223*, 197–207. https://doi.org/10.1007/978-3-319-00930-8.

[77] **Yin, X., & Liu, X.** (2018). Multi-Task Convolutional Neural Network for Pose-Invariant Face Recognition. *IEEE Transactions on Image Processing*, *27*(2), 964–975 https://doi.org/10.1109/TIP.2017.2765830.

[78] **Yan, S., Teng, Y., Smith, J. S., & Zhang, B.** (2016). Driver behavior recognition based on deep convolutional neural networks. *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, ICNC-FSKD 2016*, *1*, 636–641. https://doi.org/10.1109/FSKD.2016.7603248.

[79] **Abdel-Hamid, O., Mohamed, A., Jiang, H., Deng, L., Penn, G., & Yu, D. Y.** (2014). Convolutional Neural Networks for Speech Recognition. *IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, *22*(10), 1533–1545. https://doi.org/10.1109/ijcnn.1999.835942.

[80] **Wei, Y., Xia, W., Lin, M., Huang, J., Ni, B., Dong, J., Zhao, Y., & Yan, S.** (2015). HCP: A flexible CNN framework for multi-label image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(9), 1901–1907.

[81] **Maggiori, E., Tarabalka, Y., Charpiat, G., & Alliez, P.** (2017). Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(2), 645–657.

[82]**Url-82**<*https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad*>, date retrieved 05.06.2021.

[83] **Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L.** (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. In *Journal of Big Data* (Vol. 8, Issue 1). Springer International Publishing. https://doi.org/10.1186/s40537-021-00444-8.

[84] **Kamilaris, A., & Prenafeta-Boldú, F. X.** (2018). A review of the use of convolutional neural networks in agriculture. *Journal of Agricultural Science*, *156*(3), 312–322.

[85] **Li, R.** (2016). *Machine Learning Methods for Medical and Biological Image Computing* (Doctoral dissertation). Retrieved from https://digitalcommons.odu.edu/cgi/viewcontent.cgi?article=1015&context=computerscience_etds.

[86] **Cihan, M.** (2020). *Hiperspektral Görüntüleme Yöntemi Kullanılarak Yenidoğan Sağlık Durumlarının Derin Öğrenme Metotları İle Sınıflandırılması* (Master's thesis). Retrieved from https://tez.yok.gov.tr/UlusalTezMerkezi/.

[87]**Url-87**<*https://towardsdatascience.com/understanding-1d-and-3d-convolution-neural-network-keras-9d8f76e29610*>, date retrieved 02.06.2021.

[88]**Url-88**<*https://towardsdatascience.com/step-by-step-implementation-3d-convolutional-neural-network-in-keras-12efbdd7b130*>, date retrieved

01.06.2021.

[89] **Wang, Y., Li, Y., Song, Y., & Rong, X.** (2020). The influence of the activation function in a convolution neural network model of facial expression recognition. *Applied Sciences (Switzerland)*, *10*(5). https://doi.org/10.3390/app10051897.

[90] **Zafar, M. W.** (2018). *Object Detection and Segmentation using Region-based Deep Learning Architectures*. (Master's thesis). Retrieved from https://patrec.cs.tu-dortmund.de/pubs/theses/ma_zafar.pdf.

[91] **Lowe, D. G.** (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, *60*(2), 91–110.

[92] **Dalal, N., & Triggs, B.** (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, *I*, 886–893. https://doi.org/10.1109/CVPR.2005.177.

[93] **Lienhart, R., & Maydt, J.** (2002). An extended set of Haar-like features for rapid object detection. *IEEE International Conference on Image Processing*, *1*, 900–903. https://doi.org/10.1109/icip.2002.1038171.

[94] **Freund, Y., & Schapire, R. E.** (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139. https://doi.org/10.1006/jcss.1997.1504.

[95] **Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D.** (2013). Visual object detection with deformable part models. *Communications of the ACM*, *56*(9), 97–105. https://doi.org/10.1145/2500468.2494532.

[96] **Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A.** (2015). The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, *111*(1), 98–136. https://doi.org/10.1007/s11263-014-0733-5.

[97] **Jia Deng, Wei Dong, Socher, R., Li-Jia Li, Kai Li, & Li Fei-Fei.** (2009). *ImageNet: A large-scale hierarchical image database*. 248–255. https://doi.org/10.1109/cvprw.2009.5206848.

[98] **Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P., & Zitnick, C. L.** (2014). Microsoft COCO: Common objects in context. *European Conference on Computer Vision*, 740–755.

[99] **Girshick, R., Donahue, J., Darrell, T., & Malik, J.** (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 142–158. https://doi.org/10.1109/TPAMI.2015.2437384.

[100] **Alexe, B., Deselaers, T., & Ferrari, V.** (2012). Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(11), 2189–2202. https://doi.org/10.1109/TPAMI.2012.28.

[101] **Uijlings, J. R. R., Van De Sande, · K E A, Gevers, · T, & Smeulders, · A W M.** (2013). Selective Search for Object Recognition. *Int J Comput Vis*, *104*, 154 171. https://doi.org/10.1007/s11263-013-0620-5.

[102] **Endres, I., & Hoiem, D.** (2010). Category independent object proposals. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial*

Intelligence and Lecture Notes in Bioinformatics), *6315 LNCS*(PART 5), 575–588. https://doi.org/10.1007/978-3-642-15555-0_42.

[103] **Carreira, J., & Sminchisescu, C.** (2012). CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(7), 1312–1328. https://doi.org/10.1109/TPAMI.2011.231.

[104] **Arbeláez, P., Pont-Tuset, J., Barron, J., Marques, F., & Malik, J.** (2014). Multiscale combinatorial grouping. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *500*, 328–335. https://doi.org/10.1109/CVPR.2014.49.

[105]**Url-105**< *https://www.alegion.com/faster-r-cnn*>, date retrieved 02.06.2021.

[106] **Girshick, R.** (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 1440–1448. https://doi.org/10.1109/ICCV.2015.169.

[107]**Url-107**<*https://neurohive.io/en/popular-networks/r-cnn/*>, date retrieved 06.06. 2021.

[108]**Url-108**<*https://dzone.com/articles/from-r-cnn-to-faster-r-cnn-the-evolution-of-object*>, date retrieved 02.06.2021.

[109] **Ren, S., He, K., Girshick, R., & Sun, J.** (2017). Faster R-CNN: Towards Real- Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.

[110] **Soydaş, M.** (2019). *Aircraft Detection From Large Scale Remote Sensing Images With Deep Learning Techniques*. (Master's thesis). Retrieved from https://tez.yok.gov.tr/UlusalTezMerkezi/.

[111]**Url-111**<*https://thebinarynotes.com/region-proposal-network-faster-r-cnn/*>, date retrieved 03.06.2021.

[112] **Rella, S.** (2020). *Distributed Collaborative Framework For Deep Learning In Object Detection*. (Master's thesis). Retrieved from https://mospace.umsystem.edu/xmlui/handle/10355/74349.

[113] **Girshick, R., Donahue, J., Darrell, T., Malik, J., Berkeley, U. C., & Malik, J.** (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*, 580–587. https://doi.org/10.1109/CVPR.2014.81.

[114] **Lin, G., Zhang, Y., Xu, G., & Zhang, Q.** (2019). Smoke Detection on Video Sequences Using 3D Convolutional Neural Networks. *Fire Technology*, *55*(5), 1827–1847. https://doi.org/10.1007/s10694-019-00832-w.

[115]**Url-115**<
https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=5fCEDCU_qrC0>, date retrieved 05.03.2021.

[116]**Szegedy, C., Vanhoucke, V., Loffe, S., Shlens, J., & Wojna, Z.** (2016). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.

ÖZGEÇMİŞ

Ad-Soyad

ÖĞRENİM DURUMU:

- **Lisans :** 2016, Melikşah Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü
- **Yüksek Lisans :** 2021, İnönü Üniversitesi, Elektrik Elektronik Mühendisliği Anabilim Dalı

YÜKSEK LİSANS TEZİNDEN TÜRETİLEN ÇALIŞMALAR

- **Günay, M., Köseoğlu, M., Yıldırım, Ö. (2020).** Classification of Hand-Drawn Basic Circuit Components Using Convolutional Neural Networks. *2020 2nd Int. Congr. Human-Computer Interact. Optim. Robot. Appl. Proc.* (HORA) (pp. 1–5). Ankara: IEEE.
- **Günay, M., Köseoğlu, M. (2020).** Classification of Hand-Drawn Circuit Components by Considering the Analysis of Current Methods. *2020 4th Int. Symp. Multidiscip. Stud. Innov. Technol.* (ISMSIT) (pp. 1–5). İstanbul: IEEE.