



**DERİN ÖĞRENME YÖNTEMLERİ KULLANARAK ADLİ BİLİŞİM
İNCELEMELERİNDE DELİL ÇIKARIMININ GERÇEKLEŞTİRİLMESİ**

Arş. Gör. Mustafa ERİŞ

Yüksek Lisans Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Mustafa KAYA

TEMMUZ-2018

**T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**DERİN ÖĞRENME YÖNTEMLERİ KULLANARAK ADLİ BİLİŞİM
İNCELEMELERİNDE DELİL ÇIKARIMININ GERÇEKLEŞTİRİLMESİ**

YÜKSEK LİSANS TEZİ

Mustafa ERİŞ

(151129102)

Tezin Enstitüye Verildiği Tarih : 03 Temmuz 2018

Tezin Savunulduğu Tarih : 18 Temmuz 2018

Tez Danışmanı : Dr. Öğr. Üyesi Mustafa KAYA (F.Ü)

Diğer Jüri Üyeleri : Doç. Dr. Galip AYDIN (F.Ü)

Doç. Dr. Muhammed Fatih TALU (İnönü Ü.)

TEMMUZ-2018

ÖNSÖZ

Bu tez çalışması boyunca benden hiçbir yardımını ve desteğini esirgemeyen değerli danışmanım Sayın Dr. Öğr. Üyesi Mustafa KAYA'ya şükranlarımı sunarım.

Tez çalışmasının her aşamasında yanımda olan maddi manevi katkısını hissettiren dert ortağım sevgili eşim Fatma GÜNEŞ ERİŞ'e ve gelişile hayatımı renklendiren gücüm güç katan kızım Amine Zehra'ya teşekkür ederim.

Ayrıca yaşamımın tüm safhalarında desteklerini benden esirgemeyen babam Behzat ERİŞ, annem Perihan ERİŞ, ve kardeşim Abdullah Musab ERİŞ'e minnet ve şükranlarımı sunarım.

Mustafa ERİŞ
ELAZIĞ - 2018

İÇİNDEKİLER

Sayfa No

ÖNSÖZ.....	I
İÇİNDEKİLER	II
ÖZET.....	IV
SUMMARY	V
ŞEKİLLER LİSTESİ.....	VI
TABLolar LİSTESİ.....	IX
KISALTMALAR	X
1. GİRİŞ.....	1
2. ADLİ BİLİŞİM	3
2.1 Adli Bilişimin Tarihi.....	4
2.2 Adli Bilişimin Alt Dalları	5
2.2.1 Bilgisayar Adli Bilişimi.....	5
2.2.2 Ağ ve İnternet Adli Bilişimi	6
2.2.3 Mobil Adli Bilişim.....	6
2.2.4 Sosyal Medya Adli Bilişimi	7
2.3 Adli Bilişim İnceleme Safhaları.....	7
2.3.1 Elde Etme Safhası.....	8
2.3.2 İnceleme Safhası	9
2.3.3 Çözümleme Safhası	10
2.3.4 Raporlama Safhası	10
2.4 Adli Bilişim İncelemelerinde Kullanılan Araçlar	11
2.4.1 Yazma Koruma Cihazları	11
2.4.2 Disk Silme Yazılımları	12
2.4.3 İmaj Alma Yazılım ve Donanımları	13
2.4.3.1 Tableau TD3.....	14
2.4.3.2 CRU Wiebetech Ditto DX Forensic Fieldstation	15
2.4.3.3 Logicube Forensic Dossier	16
2.4.4 Veri Kurtarma Donanımları.....	17
2.4.5 Dijital Delil İnceleme Yazılımları	18
2.5 Adli Bilişim İncelemelerinin Zorlukları ve Önerilen Çözüm	20
3. MAKİNE ÖĞRENMESİ	23
3.1 Genelleştirme	24
3.2 Özellikler.....	24
3.3 Makine Öğrenmesi Tipleri	25
3.3.1 K-Means Algoritması	26
3.3.2 K-NN Algoritması	27
3.3.3 Destek Vektör Makineleri.....	28
3.3.4 Karar Ağaçları	29
3.3.5 Yapay Sinir Ağları	30

3.3.5.1 Yapay Sinir Ağlarının Temelleri.....	31
3.3.5.2 Çok Katmanlı Ağlar	32
3.3.5.3 Geri Yayılım.....	33
3.3.5.4 Aktivasyon Fonksiyonları	35
4. DERİN ÖĞRENME	42
4.1 Bilgisayar Görmesi	43
4.1.1 Nesne Tespiti	44
4.2 Konvolüsyonel Sinir Ağları (CNN)	45
4.2.1 Konvolüsyonel Sinir Ağlarının Yapısı	46
4.2.1.1 Konvolüsyon Katmanı.....	47
4.2.1.2 Havuzlama (Pooling) Katmanı.....	49
4.2.1.3 Aktivasyon ve Tam Bağlı Sinir Ağı Katmanları	50
4.2.2 Konvolüsyonel Sinir Ağlarının Gelişimi	50
4.2.3 Nesne Tespit Uygulamalarında Konvolüsyonel Sinir Ağları	56
4.2.3.1 Regions with CNN features (R-CNN)	57
4.2.3.2 Fast R-CNN.....	58
4.2.3.3 Faster R-CNN.....	60
4.2.3.4 You Only Look Once (YOLO)	61
4.2.3.5 Single Shot Multibox Detector (SSD).....	64
5. DERİN ÖĞRENME YÖNTEMLERİ KULLANARAK ADLİ BİLİŞİM İNCELEMELERİNDE DELİL ÇIKARIMININ GERÇEKLEŞTİRİLMESİ	67
5.1 Transfer Learning.....	68
5.2 Veri Setinin Oluşturulması ve Verilerin Etiketlenmesi	70
5.3 Tensorflow Ortamının Kurulması	73
5.4 Metodoloji.....	74
5.5 Eğitimde Kullanılan Hiper Parametreler	77
5.6 Çakışmaların Giderilmesi	80
5.7 Eğitimde Hata Fonksiyonunun Hesaplanması	82
5.8 Eğitim Sonucunun Değerlendirilmesi	83
6. DENEYSEL SONUÇLAR.....	84
7. YORUM VE SONUÇ.....	90
KAYNAKLAR	92
ÖZGEÇMİŞ.....	97

ÖZET

Günümüzde teknolojinin hızlı gelişimi, dijital ortamların günlük hayatta kullanılmasını kaçınılmaz hale getirmiştir. Bankalar, e-devlet kapısı, e-ticaret siteleri ve sosyal medya ortamlarının bilgiye çok hızlı erişim sağlanmasını mümkün kılması internet ve dijital ortamların kullanımını günlük hayatın değişmez bir parçası haline getirmiştir. Bu durum hayatı kolaylaştırmasının yanında bazı olumsuz durumların da artmasına yol açmıştır. Dijital ortamlarda işlenen dolandırıcılık, veri hırsızlığı, yasal olmayan madde satışı ve terör eylemlerinin yayılması gibi birçok suçun işlenmesi kolaylaşmış ve sayısında artış görülmüştür. Böyle durumların dijital ortamlarda engellenmesi, incelenmesi ve raporlanması adli bilişimin çalışma alanına girmektedir. Ancak, günümüzdeki mevcut adli bilişim inceleme süreçleri artan suçlara cevap veremez duruma gelmektedir.

Bu çalışmada birçok alanda kullanımı yaygınlaşan ve başarılı sonuçlar elde eden derin öğrenme yöntemlerinden adli delil incelemelerinde de yararlanılması amaçlanmıştır. Bu amaçla, derin öğrenme nesne tespiti algoritmaları kullanılarak adli delil incelemelerine yardımcı bir delil ayıklama yöntemi önerilmiştir. Alınan sonuçlar, delilden çıkarılan resim verilerinin içeriklerine göre sınıflandırılması derin öğrenme kullanılarak %93 oranında daha hızlı gerçekleştirilebileceğini göstermektedir. Bunun yanında sınıflandırma sonucunda insan kaynaklı hataların da azalacağı gösterilmiştir. Önerilen yöntemin ilerleyen çalışmalarda görüntü verilerinin yanında ses, yazı ve video verilerine de uygulanarak adli delil inceleme yazılım araçlarına entegre edilmesi amaçlanmaktadır.

Anahtar Kelimeler: Adli Bilişim, Dijital Delil İncelemeleri, Derin Öğrenme, Nesne Tespiti

SUMMARY

Extracting Evidence In Forensic Examinations by Using Deep Learning

Methods

Today, the rapid development of technology has made it inevitable to use digital media in daily life. The fact that banks, e-government portal, e-commerce sites and social media environments make it possible to access the information very quickly has made the use of internet and digital media a constant part of daily life. This situation has led to an increase in some negative situations as well as facilitation of life. Many crimes such as fraud, data theft, the sale of illegal substances, and the spread of terrorist acts committed in digital media have become easier and have increased in number. The examination, blocking and reporting of such situations in the digital environment are the subject of digital forensics science. However, current digital forensic examination processes are unable to respond to increased crime in terms of speed. In this study, it is aimed to use the deep learning methods, which have been successful in the use of many areas, in digital evidence examinations.

In this study, it is aimed to use the deep learning methods which have been successful in many use cases in judicial evidence examinations. For this purpose, a method of extracting evidence to assist in forensic examinations is proposed using deep learning object detection algorithms. The results show that the classification of images obtained from digital evidences according to their contents can be performed 93% faster using deep learning. It has also been shown that human errors will be reduced as a result of classification. It is aimed to integrate the proposed method into the forensic evidence analysis software tools by applying it to audio, text and video data as well as image data in further studies.

Keywords: Digital Forensics, Deep Learning, Digital Evidence Examination, Object Detection

ŞEKİLLER LİSTESİ

Sayfa No

Şekil 2.1. Adli delil inceleme adımları	8
Şekil 2.2. Çeşitli üreticiler tarafından üretilmiş yazma koruma cihazları.....	12
Şekil 2.3. Çeşitli üreticiler tarafından üretilmiş disk silme donanımları	13
Şekil 2.4. Çeşitli üreticiler tarafından üretilmiş disk kopyalama donanımları.....	14
Şekil 2.5. Tableau TD3 adli imaj alma donanımı	15
Şekil 2.6. CRU Wiebetech Ditto DX adli kopya alma cihazı	16
Şekil 2.7. Forensic Dossier Adli Kopyalama Donanımı.....	16
Şekil 2.8. Çeşitli üreticiler tarafından üretilmiş veri kurtarma donanımları	17
Şekil 3.1. K-means algoritması işlem adımları	26
Şekil 3.2. K-NN algoritması	27
Şekil 3.3. DVM sınıflandırıcı.....	28
Şekil 3.4. DVM kernel uygulanma işlemi.....	29
Şekil 3.5. Sınıflandırma ağacı ve regresyon ağacı	30
Şekil 3.6. Yapay ve biyolojik nöron yapısı	31
Şekil 3.7. Çok katmanlı tam bağlı yapay sinir ağı örneği	32
Şekil 3.8. Örnek kayıp fonksiyonu.....	34
Şekil 3.9. Hata grafiğinin eğimi yardımı ile parametrenin güncellenmesi [43].....	34
Şekil 3.10. Lineer aktivasyon fonksiyonu.....	36
Şekil 3.11. Sigmoid aktivasyon fonksiyonu ve türev grafiği.....	37
Şekil 3.12. Tanh fonksiyonu ve türev grafiği.....	38
Şekil 3.13. RELU fonksiyonu ve türev grafiği	39
Şekil 3.14. Leaky ve Parametrik RELU grafiği	40
Şekil 3.15. Swish aktivasyon fonksiyonu grafiği.....	40
Şekil 4.1. Sınıflandırma, lokalizasyon ve nesne tespiti.....	44
Şekil 4.2. Konvolüsyonel sinir ağlarının genel yapısı	46
Şekil 4.3. Konvolüsyon işlemi	48
Şekil 4.4. $p = 1$ için padding sonucu	49
Şekil 4.5. Maksimum ve ortalama havuzlama	50
Şekil 4.6. LeNet yapısı [66].	51

Şekil 4.7. AlexNet yapısı [57].	52
Şekil 4.8. ZFNet yapısı [72].	52
Şekil 4.9. ZFNet sinir ağında eğitim boyunca rasgele seçilen özellik haritalarının görselleştirilmesi [72].	53
Şekil 4.10. VGGNet yapısı [58].	53
Şekil 4.11. Basit inception modül yapısı [59].	54
Şekil 4.12. GoogleNet yapısı [59].	55
Şekil 4.13. Residual blok yapısı [2].	56
Şekil 4.14. 34 Katmanlı ResNet yapısı [2].	56
Şekil 4.15. R-CNN çalışma prensibi [62].	57
Şekil 4.16. Fast R-CNN çalışma prensibi [48].	58
Şekil 4.17. R-CNN ve Fast R-CNN çalışma süreleri [48,62].	59
Şekil 4.18. RPN yapısı [4].	60
Şekil 4.19. R-CNN çalışma süreleri [4,48,62].	61
Şekil 4.20. YOLO çalışma prensibi [56].	62
Şekil 4.21. YOLO ağ mimarisi [56].	63
Şekil 4.22. Fast R-CNN ve YOLO algoritmalarının arka plandan, benzer sınıftan, lokalizasyondan ve diğer nedenlerden kaynaklanan hatalarının ve doğruluk oranlarının karşılaştırılması	64
Şekil 4.23. Varsayılan sınırlayıcı kutuların gerçek sınırlayıcı kutuları tahmin etmesi [55].	65
Şekil 4.24. SSD ve YOLO mimarilerinin karşılaştırılması [55].	65
Şekil 4.25. SSD, Fast R-CNN ve Faster R-CNN doğruluk oranlarının karşılaştırılması [55].	66
Şekil 5.1. Veri Seti büyüklüğüne göre izlenebilecek transfer learning yöntemi [90].	70
Şekil 5.2. Veri setinde bulunan örnek resimler	71
Şekil 5.3. Farklı nesneleri gösteren sınırlayıcı kutular [56].	71
Şekil 5.4. Labelmg programı ile oluşturulan sınırlayıcı kutular	72
Şekil 5.5. Şekil 5.4'te gösterilen ilk resim için oluşturulan xml dosyası	72
Şekil 5.6. Kurulan yazılımların doğrulanması	74
Şekil 5.7. Derin öğrenme kullanılarak oluşturulan adli delil inceleme aşamaları	74
Şekil 5.8. Otomatik anlamlandırma adımı	75
Şekil 5.9. Tensorflow object detectidon api test programının çıktısı	76
Şekil 5.10. COCO veri seti ile eğitilmiş modeller ve hız, doğruluk, çıktı bilgileri	76
Şekil 5.11. Yatay ekseninde çevrilen örnek resim	78
Şekil 5.12. Tüm veri seti ve mini-batch eğitimi hata grafikleri [96].	78

Şekil 5.13. Eğim değışimi ve adım sayısı arasındaki bağlantı [43].	79
Şekil 5.14. Öğrenme hızı seçiminin eğitime etkisi [97].	80
Şekil 5.15. Non-maximum suppression algortiması öncesi ve sonrası [98].	81
Şekil 5.16. IoU değerin hesaplanması.....	81
Şekil 6.1. COCO veri setinde bulunan kategoriler [101].	84
Şekil 6.2. Faster R-CNN tarafından çıkarılan resimler	86
Şekil 6.3. Eğitimde elde edilen hata grafiğı (smoothing: 0.7)	87
Şekil 6.4. Test seti üzerinde elde edilen başarıml (smoothing: 0.7)	87
Şekil 6.5. Eğitilen model tarafından işlenen test resimleri	88
Şekil 6.6. Eğitilen modelin hata yaptığı resimler.....	88

TABLÖLAR LİSTESİ

Sayfa No

Tablo 2.1. Yaygın olarak kullanılan adli delil inceleme yazılılarının özellikleri	19
Tablo 5.1. Veri seti durumuna göre tercih edilebilecek transfer learning yöntemleri	69
Tablo 6.1. Modellerin doğruluk oranları ve sınıflandırma hızı.....	85



KISALTMALAR

SIM	: Subscriber Identity Module
CDR	: Arama Ayrıntı Kayıtları
HPA/DCO	: Host Protected Area ve Device Configuration Overlay
SIFT	: SANS Investigative Forensic Toolkit
FTK	: Forensic Tool Kit
K-NN	: K-Nearest Neighbors
DVM	: Destek Vektör Makinesi
CNN	: Convolutional Neural Network
RELU	: Rectified Linear Unit
DNN	: Deep Neural Network
HOG	: Histogram of Oriented Gradients
SIFT	: Scale-Invariant Feature Transform
ILSVRC	: ImageNet Large Scale Visual Recognition Competition
RCNN	: Regions with CNN Features
RPN	: Region Proposal Network
YOLO	: You Only Look Once
SSD	: Single Shot Multibox Detector
API	: Application Programming Interface
CPU	: Central Processing Unit
GPU	: Graphics Processing Unit
IoU	: Intersection over Union
AP	: Average Precision
mAP	: Mean Average Precision

1. GİRİŞ

Son yıllarda, teknolojiadaki gelişim ile birlikte internet, bilgisayar ve mobil cihazların kullanımı hızlı bir şekilde artmıştır. Günümüzde başta sosyal medya olmak üzere, devlet hizmetleri, bankacılık, eğlence, e-ticaret gibi birçok alanda dijital ortamlar tercih edilmektedir. Bununla birlikte, dijital ortamdaki güvenlik açıkları ve suç oranları da artmıştır. Dolandırıcılık, terör faaliyetleri, bilgi hırsızlığı, yasal olmayan maddelerin satışı, sistemlere izinsiz erişim ve istismar gibi birçok suç dijital ortamda kolaylıkla işlenmektedir. Bu suçlar işlenirken suçlular dijital ortamlarda isteyerek ya da istemeyerek deliller bırakmaktadır. İşlenen bu suçların dijital ortamda bırakılan izler kullanılarak incelenmesi, analiz edilmesi ve raporlanması adli bilişimin çalışma alanına girmektedir.

Adli bilişimin ortaya çıktığı ilk yıllardan bu yana suça yönelik incelenmesi gereken veri miktarı sürekli artmıştır. Artan veri miktarına paralel olarak büyük miktarda veriyi işlemek için, sıralama, indeksleme, metin arama, sayısal özet karşılaştırma, veri kazıma ve dosya tipine göre sınıflandırma gibi özelliklere sahip adli inceleme yazılımları geliştirilmiştir. Ancak adli delil inceleme yazılımları hız bakımından günümüzdeki ihtiyaca tam olarak karşılık verebilecek seviyede değildir. Garfinkel, çalışmasında [1], günümüzdeki adli bilişim araçlarının sadece veri parçalarının bulmak için tasarlandığını ve incelemeciye yardımcı olacak bir özelliğinin bulunmadığını belirtmiştir. Ayrıca, anormal verileri tespit edebilen, incelemeciler gibi veriler üzerinde fikir yürütebilen, doğruluk oranı yüksek adli delil inceleme araçlarının bulunmadığını ve geliştirilmesi gerektiğini vurgulamıştır. Örneğin, günümüzde adli delil inceleme yazılımları delilden elde edilecek tüm resim dosyalarını çıkarabilmektedir. Ancak resim verilerini içerdikleri görüntülere göre yorumlamak tamamen adli delil incelemesini gerçekleştiren uzmanlar tarafından gerçekleştirilmektedir. Henüz böyle bir değerlendirmenin yapıldığı araç bulunmamaktadır. Bu yüzden resim verilerinin incelenmesi adli delil incelemelerinde en fazla zaman alan bölümlerden birisidir.

Konvolüsyonel sinir ağları son yıllarda resim sınıflandırma, nesne tespiti, resim bölütleme gibi resim üzerinde gerçekleştirilen bilgisayar görmesi problemlerinde büyük başarı sağlamıştır [2-4]. Bu yüzden bu tez çalışmasında adli delil inceleme sürecinin geliştirilmesi için adli delil inceleme adımlarında adli bilişim araçlarının yanında resim verilerinin ayrıştırılmasında konvolüsyonel sinir ağlarından yararlanılması amaçlanmıştır.

Konvolüsyonel sinir ağıları kullanılarak resim verileri içeriğine göre ayrıştırılarak inceleyiciye sunulmuş, bu sayede adli delil incelemelerinin daha hızlı ve doğruluk oranı yüksek olarak gerçekleştirilmesi sağlanmıştır.

Bu tez çalışmasında iki farklı çalışma alanının birleşimi sonucunda bir sonuç ortaya konulduğundan dolayı ilk olarak bu alanlara ait teorik bilgiler verilmiştir. Daha sonra da uygulamanın gerçekleştirilmesi anlatılarak alınan sonuçlar yorumlanmıştır. Bölüm 2’de adli bilişim tanımı, adli bilişimin tarihi, adli bilişimde kullanılan yöntemler, adli bilişimde karşılaşılan zorluklar ve önerilen yöntem anlatılmıştır. Bölüm 3’de önerilen yöntemin anlaşılması için gerekli temel makine öğrenmesi başlıkları anlatılmıştır. Bölüm 4’te önerilen yöntemde kullanılan derin öğrenme yöntemleri, konvolüsyonel sinir ağıları ve yapısı, nesne tespiti uygulamalarında literatür özeti ve çalışma prensipleri anlatılmıştır. Bölüm 5’de önerilen yöntemin gerçekleştirilmesi, kullanılan yöntemler ve yapılan hazırlıklar anlatılmıştır. Bölüm 6’da yapılan uygulama ile elde edilen sonuçlar yer almıştır. Son olarak bölüm 7’de sonuç, yorumlar ve gelecekte yapılması düşünülen çalışmalar verilmiştir.

2. ADLİ BİLİŞİM

Bilişim, her tür mekanik hesap ve bilgi işlevini inceleme ve bilgiyi elektronik ortamda makineler aracılığı ile düzenli biçimde işleme bilimidir. İşlemenin yanı sıra günümüzde bilginin aktarılacağı, işleneceği ve depolanacağı ortamların kullanılması, geliştirilmesi ve incelenmesi de bilişim kavramı ile tanımlanabilmektedir. Ancak bilişim ile bilginin işlenmesi sırasında bazı kural ve kısıtların bulunması gerekmektedir. Bunun nedeni bilişimin de çeşitli suçlara alet olabilmesidir. Bu durumda ortaya çıkan suça bilişim suçu adı verilmektedir. Bilişim suçu; bilgilerin işlendiği veya aktarıldığı dijital bir ortamda, kanun dışı, ahlak dışı veya yetkisiz olarak gerçekleştirilen işlemler olarak veya bilgisayar ve iletişim teknolojileri kullanılarak işlenen suçlar şekliyle de tanımlanabilir. Bu gibi durumların engellenmesi amacıyla bilişim hukuku ortaya çıkmıştır. Bilişim suçlarına yönelik Türkiye'de ilk yasal metin, 765 sayılı Türk Ceza Kanununa 1991 yılında eklenen "...bilgileri otomatik işleme tabi tutan sistem..." ibaresidir. Bundan sonra ortaya çıkan ihtiyaçlar neticesinde birçok kanuna bilişim ile ilgili hükümler eklenmiştir. Bilişim suçları ile ilgili en kapsamlı düzenleme 5237 sayılı Türk Ceza Kanununda yer almaktadır. Bilişim suçları ile ilgili en kapsamlı düzenleme 5237 sayılı Türk Ceza Kanunu'nda yer almaktadır. Türk Ceza Kanununun 243. 244. ve 245. maddelerinde bilişim suçları düzenlenmiş ve açıklanmıştır [5].

Günümüzde, kanunlara rağmen bilişim alanında suçlar işlenmekte ve teknolojinin gelişmesiyle birlikte, bilişim suçlarının sayısı gün geçtikçe artmaktadır. Bu suçların incelenmesi ve değerlendirilmesi de adli bilişimin ilgi alanına girmektedir.

Adli bilişim, bir suç için delil niteliği taşıyan veya taşıyabilecek dijital ortamdaki verilerin, korunması, tanımlanması, çıkarılması, analizi, dokümantasyonu ve sunumunu içeren bir bilim dalıdır. Yani, bir olay yeri incelemesinde dijital delillerin ele geçirilmesinden, delillerin değerlendirilmesine ve çıkarımların raporlanmasına kadar olan tüm süreçler adli bilişimin ilgi alanı dâhilindedir. Adli bilişim, hem hukuk hem de bilişimi kapsayan konuları kapsamaktadır [6]. Bunun anlamı, dijital delillerin toplanması ve incelenmesinden sonra çıkarılan sonuçların da davaların işleyişinde belirleyici olduğudur. Ancak, adli bilişimin amacı, herhangi bir kişi veya kurumu suçlamak veya aklamak değildir. Adli bilişimin hukuksal boyuttaki görevi sadece dijital ortamdaki delil niteliği taşıyan verilerin eksiksiz ve objektif olarak adli birimlere teslim edilmesidir. Adli bilişim

uzmanlarının raporları ışığında nihai karar verme işlemini yargı makamları gerçekleştirmektedir. Her alt dal genel bir süreç çatısı altında olmak üzere, birbirinden farklı ortamların incelemesini gerçekleştirmeyi hedeflemektedir.

2.1 Adli Bilişimin Tarihi

Günümüzde adli bilişim, bilgisayarda işlenen suçların (ör. Kimlik avı ve banka sahtekârlığı) yanı sıra, bilgisayarda işlenmese bile kanıtların bir bilgisayarda bulunabileceği suçların çözümünde (ör. Kara para aklama ve çocuk istismarı) önemli bir araçtır. Ayrıca, adli bilişim araçları, siber saldırıların bıraktığı kanıtları yeniden yapılandırma becerileri nedeniyle bilgi güvenliği açısından hayati bir araç haline gelmiştir. Bu duruma gelene kadar adli bilişim çeşitli süreçlerden geçmiş ve günümüzdeki önemine kavuşmuştur.

Adli bilişim yaklaşık kırk yaşındadır. Şu anda adli bilişim teknikleri olarak nitelendirilen teknikler, temel olarak veri kurtarma amacıyla geliştirilmiştir. Örneğin, Wood ve ark. çalışmalarında dikkatsiz bir incelemeci tarafından silinen parçalanmış bir veri tabanı dosyasının tek kopyasını kurtarmak için 70 saat boyunca çalışan iki veri kurtarma uzmanından bahsetmektedir [7]. 1980'lerin sonuna gelindiğinde, "Unformat, Undelete, Diagnose & Remedy" gibi çeşitli veri kurtarma programları çıkarıldı ve yaygın olarak kullanılmaya başlandı. Adli bilişimin bu ilk zamanlarında dosya tiplerinde çoğalma, donanım yazılım ve uygulama çeşitliliğinde çoğalma gibi gelişmeler devam ederken, adli bilişim resmi süreçleri, araç ve gereçler hakkında eğitimsizlik yüksek düzeyde idi. Bu ilk günlerde adli bilişim büyük oranda, kolluk kuvvetleri ile geçici ve tek davalık çalışan bilgisayar uzmanları tarafından gerçekleştirildi. Ayrıca adli bilişime duyulan gereksinim azdı. Bunun nedeni anti-adli bilişim hakkında bilgi olmaması, disklerin küçük olmasına bağlı suç unsurlarının kolayca saptanması gibi nedenlerdi; çok az davada dijital delilin özel olarak incelenmesine gereksinim duyulmaktaydı.

1999-2007 yılları, adli bilişim için bir tür Altın Çağı haline geldi. Bu süre zarfında, adli bilişim, silinmiş olduğu düşünülen verilerin çıkarılabilmesi ile geçmişte görebileceğiniz bir tür sihirli pencere haline geldi. Bu tarihler arasında; Microsoft Windows'un özellikle Windows XP'nin yaygın olarak kullanılması, genellikle araştırmalarda daha az dosya formatı ile karşılaşılması (dokümanlar için Microsoft Office, grafik ve resim dosyaları için JPEG vb., ve video için AVI ve WMV), olayların genellikle tek bir bilgisayar sistemi üzerinde gerçekleşmesi, bilgisayarlardan rahatlıkla ayrılarak incelenebilen disklerin

kullanılması (IDE/ATA diskler gibi) ve silinmiş verilerin elde edilmesini sağlayan birçok yazılımın geliştirilmesi gibi nedenler adli bilişimin hızlı bir şekilde gelişmesini ve yaygınlaşmasını sağlamıştır.

Günümüzde adli bilişim bilgisayar ortamında gerçekleşen veya gerçekleşmeyen her türlü suçun incelenmesinde ve çözülmesinde değişmez bir standart haline gelmiştir. Türkiye’de de özellikle son yıllarda adli bilişim hakkında farkındalık oluşmuş ve devlet kurumları da dâhil olmak üzere birçok kuruluş adli bilişim uzmanlarına ihtiyaç duymaya başlamıştır. Teknolojinin gelişmesi adli bilişimin sürekli yenilenen bir bilim dalı olmasını sağlamış hatta ilk ortaya çıktığında sadece bilgisayarın incelenmesini kapsayan adli bilişim, delillerin çeşitliliğinin artması ve her dijital delilin kendine göre inceleme zorluklarının olması nedeniyle, ilgilenilen delillerin niteliğine göre alt dallara ayrılmıştır.

2.2 Adli Bilişimin Alt Dalları

Adli bilişim bilim dalı 1970’lerde ilk olarak ortaya çıktığında sadece kişisel bilgisayarları konu almaktaydı. Bilgisayarların daha sonra ağlar oluşturması ve internet gibi dünya genelinde bir ağın kişisel kullanım oranının hızla artması sonucu, adli bilişimin sadece kişisel bilgisayarlar ile sınırlı olması olanaksız hale gelmiştir. Bu nedenle adli bilişim adının altında, alt dallar oluşmuştur. Her alt dal genel bir süreç çatısı altında olmak üzere, birbirinden farklı ortamların incelemesini gerçekleştirmeyi hedeflemektedir.

Adli bilişim birçok başlık altında incelenebilir. En çok kullanılanı bilgisayar, ağ ve internet, cep telefonu, GPS, multimedya cihazı, çok oyunculu oyun, oyun konsolu ve sosyal medya adli bilişimidir. Bununla birlikte, bu alt dallarda kendi içinde ayrılmaktadır. Brinson ve ark. yaptıkları çalışmada adli bilişim, alt dalları ve kullanılan araçları özetleyerek adli bilişim ontolojisini ortaya koymuştur [8].

2.2.1 Bilgisayar Adli Bilişimi

Bilgisayar adli bilişimi, adli bilişimi oluşturan en eski alt dallarından birisidir. Bilgisayarlar genellikle bir davaya yön veren dijital delillerin ana kaynağıdır. Bilgisayarlar, içlerinde çok büyük miktarda veri tutmakla birlikte, USB cihaz, cep telefonu, dijital kamera ve harici harddisk gibi bilgisayar ile etkileşime geçen cihazlar hakkında da bilgiler verebilmektedir. Çünkü bütün bağlı cihazlar bir şekilde işletim sistemi üzerinden

bilgisayarlar ile etkileşime geçmektedir. Bir cep telefonun veya dijital kameranin yedekleme dosyasının oluşturulması veya hafızasının boşaltılması için bilgisayara bağlanması durumunda bu bilginin bilgisayar işletim sistemi tarafından saklanması örnek olarak verilebilir. Bilgisayar adli bilişimi genel olarak kullanıcı hesapları, günlük dosyaları, zaman damgaları, çoklu ortam dosyaları, e-postalar gibi bilgisayar harddiskinde depolanan bilgilerle ve bazı durumlarda bilgisayarın RAM donanımındaki canlı verilerin incelenmesini konu almaktadır [9].

2.2.2 Ağ ve Internet Adli Bilişimi

Ağ ve internet adli bilişimi, bir sistemin ağ yapısının, sisteme giren ve çıkan paketlerin, sisteme ait günlük dosyalarının incelenmesini; ayrıca ağ sistemlerinin gerçek zamanlı izlenmesi ve ağ trafiğinin durumuna göre sistemin değerlendirilmesi gibi işlemlerini kapsamaktadır [10]. Bu alt dal genellikle ağ üzerinden saldırılar, ağa izinsiz sızma ve kalıcı zararlı yazılım gizleme gibi ağ üzerinde oluşabilecek istismarları incelemektedir. Ayrıca sisteme herhangi bir saldırının yapılmadığı durumda, pentest, ağ sistem taraması gibi işlemlerin yapılabilmesi için de ağ ve internet adli bilişimi alt dalı hakkında detaylı bilgi sahibi olmak gerekmektedir.

2.2.3 Mobil Adli Bilişim

Mobil adli bilişim, kullanıcıların ve servis sağlayıcıların verilerinin bulunduğu mobil cihazların hafızalarının, arama ayrıntı kayıtları (CDR) gibi bilgilerin bulunabileceği SIM kartların ve cep telefonu servis sağlayıcıları tarafından oluşturulan kayıtların incelenmesini konu alır.

Mobil cihazların incelenmesi, günlük hayatta çok yaygın kullanımları nedeniyle bilgisayar incelemesi kadar yaygınlaşmıştır. Günümüzde aktif olarak kullanılan sadece cep telefonu sayısı yaklaşık olarak dünya nüfusunun yüzde altmışaltısına denk gelmektedir [11]. Mobil cihazlar, fazla miktarda kişisel veri barındırdığı için bilgi miktarı bakımından oldukça zengindir ve incelenmeleri sonucunda, kanıt niteliği taşıyan birçok veri elde edilmektedir. Örneğin, telefon rehberindeki kayıtlar, kısa mesajlar, resim dosyaları, videolar, ses kayıtları ve e-postalar gibi veriler birçok kişisel veri barındırmakta ve bu bilgilerden birçok delil ele geçirilmektedir. Farklı cep telefonu şebekeleri ve servis sağlayıcılarına ek olarak, binlerce

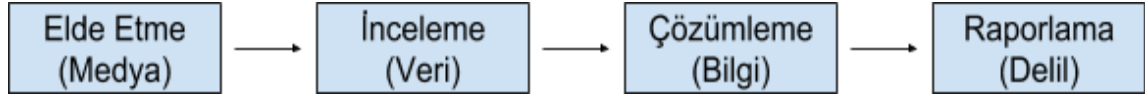
farklı model ve mobil cihaz markası sayesinde, bir mobil cihazdan veri kurtarılabiliyor kurtarılamaması durumu değişiklik gösterebilmektedir.

2.2.4 Sosyal Medya Adli Bilişimi

Son yıllarda sosyal medya kullanımının hızlı bir şekilde yaygınlaşması sonucu insanların kişisel bilgileri ve düşünceleri internet ortamında daha fazla yer almaya başlamıştır. Bu nedenle suçların analizinde de sosyal medya platformlarının analiz edilmesi ihtiyacı oluşmuştur. Bu ortamlarda incelemelerin yapılması veriler arasında ilişki kurulması ve elde edilen bulguların raporlanması sosyal medya adli bilişimin ilgi alanına girmektedir [12]. Sosyal medya ortamlarında, yasal olmayan ve genellikle zararlı olan bazı maddelerin satışı, terör amaçlı kitlesel yönlendirmeler, siber dolandırıcılık gibi tehditlerin takip edilerek suç ve suçluların saptanması mümkün olmaktadır.

2.3 Adli Bilişim İnceleme Safhaları

Bahsedilen tüm bu adli bilişim alt dallarında belirli bir sistematığa bağlı kalınarak delil inceleme aşaması gerçekleştirilir. Bunun nedeni adli incelemelerin uygulanması aşamasında bir standardizasyona ihtiyaç duyulmasıdır. Adli bilişim inceleme aşamasının düzenlenmesi ve en iyi yöntemin bulunması için çeşitli çalışmalar yapılmıştır [13-16]. Her ne kadar standart olarak kabul edilen bir yöntem bulunmasa da tüm önerilen yöntemlerde ortak olan safhalar genel adli bilişim incelemesi safhalarını ortaya çıkarmaktadır. Dijital delil inceleme işlemine genel olarak bakıldığında delilin elde edilmesi, incelenmesi, çözümlenmesi ve raporlanması aşamaları en temel aşamalar olarak düşünülebilmektedir. Bu aşamalar temel bilgisayar adli bilişimi aşamalarıdır. Ortak safhaların bilgisayar adli bilişimi safhalarından oluşması adli bilişim alanında ilk alt dalın bilgisayar adli bilişimi olmasıdır. Geri kalan alt dallar bilgisayar adli bilişimini referans alarak kendi özel durumları için yeri kuralların eklenmesi yoluyla oluşturulmuştur. Buna rağmen bilgisayar adli bilişiminin kendi içinde dahi inceleme aşamaları farklılıklar gösterebilmektedir. Bazı kaynaklarda 3 temel adımdan bahsedilirken, bazı kaynaklarda ise 4, 5 veya 6 adımdan dahi bahsedilebilmektedir [17-20]. Bu tez çalışmasında tüm bakış açılarını kapsayan en genel adımlar 4 adımda incelenmiştir.



Şekil 2.1. Adli delil inceleme adımları

Şekil 2.1’de ele alınan adli bilişim inceleme adımları gösterilmiştir. Şekilde görüldüğü gibi elde etme aşamasında dijital delil ihtiva etmesi muhtemel olan medya ortamlarının toplanması gerçekleştirilir. İnceleme safhasında elde edilen medya ortamlarından manuel olarak, yazılım veya donanım kullanarak elde edilebilecek tüm verilerin çıkarılması işlemi gerçekleştirilir. İnceleme aşamasında ise elde edilen veriler detaylı olarak incelenerek bu verilerden herhangi birinin veya birilerinin araştırılan suçla ilişkisinin olup olmadığı araştırılır. Elde edilen sonuçlar raporlama aşamasında suça ilişkin deliller olarak ilgili makamlara iletilir.

2.3.1 Elde Etme Safhası

Elde etme safhası dijital delillerin incelenmek üzere yasal alt yapıya uygun olarak olay mahallinden toplandığı aşamadır. Toplama aşamasında olay mahallinin de önemi çok büyük olmaktadır. Delillerin delil niteliğinin kaybolmadan olay yerinden toplanabilmesi için olay yerine usule uygun bir şekilde müdahale edilmesi gerekmektedir. İlk olarak olay yerinin güvenliği sağlanmalı ve sadece yetkili kişilerin olay yerine girebilmesi sağlanmalıdır. Bu sayede delillerin karartılması veya bozulması ihtimalinin önüne geçilmiş olmaktadır. Dijital delillerin hassas yapıya sahip olmasından dolayı bozulması ve ortadan kaldırılması diğer delillere göre daha kolay olabilmektedir. Delil toplama işinin adli bilişim uzmanları tarafından yapılması da önemli bir konudur. Yetkinliği olmayan bir kimse tarafından delil niteliği taşıyan bazı objeler saptanamadığından göz ardı edilebilmektedir. Ayrıca olay yerinde bulunan diğer delillerin elde edilmesi sırasında dijital delillerin yok edilmemesine veya zarar görmemesine özen gösterilmelidir. Olay yerinde ilk önlemler alındıktan sonra ilk olarak olay yerinin fotoğrafları çekilmeli ve delillerin bulunduğu ortam ve yerleri hakkında detaylar raporlanmalıdır. Bu adımlarda kriminal ve adli bilişim yöntemleri beraber yürütülür. Olay yerinin güvenliği sağlandıktan ve uygun ortam sağlandıktan sonra elektronik delillerin çalışma biçimleri ve birbirlerini etkileyip etkilemedikleri tespit edilir. Eğer birbirine bağımlı deliller var ise ona göre müdahale edilir. Örneğin; olay yerinde bulunan açık bir bilgisayar kesinlikle direkt olarak kapatılmamalıdır, aynı şekilde kapalı olan bir bilgisayar da açılmamalıdır. Aksi halde, elde edilebilecek verilerin kaybedilmesine veya

delil niteliğini kaybetmesine yol açılabilmektedir. Bu hususlara dikkat edilerek tüm dijital deliller toplandıktan sonra inceleme aşamasına geçilecektir ancak adli bilişim incelemeleri elde edilen donanımlar üzerinde değil, delillerin birebir kopyaları (imajları) üzerinde gerçekleştirilmelidir. Bu nedenle delillerin imajları alınarak orijinal deliller güvenli bir şekilde saklanmalıdır. İmaj alma işlemi sırasında da cihazın yapısı ve imaj alan kişinin bilgisine bağlı olarak veri kaybı yaşanabilmektedir. Ayrıca dikkatli davranılmadığı ve usule uygun gerçekleştirilmediği durumda deliller delil olma özelliğini kaybedebilmektedir. Örneğin bir diskin imajı alınmak istenirken bilgisayara direkt olarak bağlanması disk üzerinde değişikliklere yol açabilir ve böylece delil özelliğini kaybetmiş olabilir. Bu gibi durumlardan kaçınmak amacıyla imaj alma sırasında çeşitli donanım veya yazılımlardan yararlanılmaktadır. Donanım ve yazılımlar sayesinde hem orijinal deliller korunmakta hem de adli delil inceleme safhaları sağlıklı bir şekilde yürütülebilmektedir. Bu nedenle delile müdahale edilmeden önce delilin teknik özelliklerinin dikkate alınması büyük önem taşımakta ve buna uygun doğru müdahaleyi gerçekleştirecek araç ve yöntemler seçilmelidir. Dijital kopyaları da alınan deliller bu safhadan sonra incelemeye hazır hale gelmiş olmaktadır.

2.3.2 İnceleme Safhası

Elde etme safhasından sonra imajı alınan delillerin incelenmesinin amaçlandığı inceleme safhasına geçilir. Elde edilen imaj dosyası, dijital delilin silinmiş, gizlenmiş veya açık olarak bulunan bütün verilerini kapsamaktadır. İnceleme safhasında da elde edilen bu verilere çeşitli işlemler uygulanarak olası suç unsurlarının imaj dosyaları içerisinde çıkarılması ve saptanması amaçlanmaktadır. Kısaca karmaşık veya gizli-kayıp verilerin gözle görülür ve detaylı olarak değerlendirilebilir hale getirilmesi işlemi inceleme safhasının amacıdır. İnceleme safhasında bir dijital delilden elde edilebilecek tüm veri tipleri ortaya çıkarılmaktadır. Bilgisayar adli bilişimi için, ofis dosyaları, resimler, videolar, programlar, e-postalar, web geçmişi, dosya klasörler ve her bir verinin oluşturulma-değiştirilme tarihi, silinmiş veriler örnek olarak verilebilir. Tüm bu verilerin manuel olarak elde edilmesi oldukça zordur ve çok uzun zamanlar alabilir; hatta bazı verilere ulaşlamaya da bilir. Bu yüzden inceleme safhasında verilerin ortaya çıkarılması amacıyla çeşitli yazılımlardan yararlanılmaktadır. Bu yazılımlar otomatik olarak silinen verileri arama ve kurtarma, dosyaları indexleme, gömülü verileri çıkarma, hızlı hash karşılaştırması yapma gibi birçok

özelliğe sahiptir. Kullanılan yazılımlar sayesinde adli delil inceleme aşaması doğruluk ve hız kazanmaktadır. Ancak yazılımlar, sadece kullanıcıların istediği doğrultudaki verilerin tamamını toparlayarak bir yığın haline sunmaktadır. Yani, sadece inceleyicinin anlayabileceği ve hızlı erişebileceği şekilde olası delilleri sunar. Elde edilen tüm bu bulguların incelenmesi, çözümleme aşamasında adli bilişim uzmanları tarafından gerçekleştirilmektedir.

2.3.3 Çözümleme Safhası

Çözümleme safhasında, imaj dosyalarından çıkarılarak inceleyiciye sunulan büyük miktarda veri adli bilişim uzmanları tarafından analiz edilir. Bu aşamada herhangi bir yazılım veya donanım kullanılmamaktadır. Sadece daha önceden kullanılmış olan donanım ve yazılımların sunmuş oldukları verilerin anlamlandırılması ve suç unsuru içerdiği düşünülerek adli makamlara bildirilecek olan verilerin saptanması amaçlanmaktadır. Dolayısı ile bu aşama adli delil inceleme sürecinin en kritik safhasıdır. Çünkü davanın tüm seyrini değiştirebilecek olan, suç unsurlarının bulunduğu ve raporlanmaya hazırlandığı bu safha sadece insan algılarına ve tecrübesine bağlıdır. Delillerin otomatik olarak bulunduğu bir sistem henüz bulunmamaktadır. Bu nedenle adli delil incelemelerinin en uzun süren safhası da genellikle çözümleme safhası olmaktadır. Bu aşamada suç ile ilgisi olmayan tüm veriler temizlenecek ve raporlama safhasına aktarılmayacaktır. Ancak bu safhada elde edilecek sonuç tamamen insan algılarına bağlı olduğundan dolayı insan kaynaklı istemli veya istemsiz hatalarla sık sık karşılaşmaktadır. Bu durum hem incelemeyi yapan kişi hem de mağdur olan kişiler açısından büyük problemlere yol açmaktadır. Bu nedenle dijital delil incelemelerinde üzerinde durulması gereken önemli safhalardan birisi çözümleme safhasıdır.

2.3.4 Raporlama Safhası

Dijital delillerin toplanması, anlaşılır ve analiz edilebilir duruma getirilmesi ve analiz edilerek çözümlenmesinin ardından incelemenin son safhası olan raporlama safhasına geçilir. Bu safhada, çözümleme safhasında dava ile ilgili olduğu belirlenen verilerin adli makamlara sunulabilmesi için gerekli işlemler gerçekleştirilir.

Raporlama safhasında elde edilen deliller referans alınarak, elde edilme yöntemleri, delillerin ayrıntıları, kullanılan adli bilişim yöntemleri ve gerekli teknik bilgilerle birlikte,

bu konularda bilgi sahibi olmayanların da anlayabileceği bir dilde detaylı bir rapor hazırlanır. Hazırlanan raporda delillerle ilgili bilgiler dışında, incelemenin yapıldığı zaman dilimi, kullanılan araç gereçler ve inceleyici hakkında bilgiler de yer almalıdır. Raporun ilgili makamlara iletilmesi ile birlikte dijital delil inceleme süreci sona ermiş olur. İnceleme safhasından sonra deliller muhafaza edilmeli ve daha sonradan gerçekleştirilebilecek bir kontrol veya yeniden inceleme için hazır durumda saklanmalıdır.

2.4 Adli Bilişim İncelemelerinde Kullanılan Araçlar

Adli bilişim incelemelerinde, delillerin elde edilebilmesi, delil inceleme sürecinin hızlandırılması, delil niteliği taşıyan verilerin tamamının elde edilebilmesi gibi birçok nedenden dolayı çeşitli donanım ve yazılım araçları kullanılmaktadır. Bu yazılım ve donanımlar adli bilişim yazılım ve donanımları olarak adlandırılmaktadır. Adli bilişim donanımlarının tamamı ücretli olarak elde edilebilmektedir. Ancak adli bilişim yazılımlarının bir kısmı ücretli olarak edinilebilirken bir kısmı ise lisanssız yani açık kaynak kodlu olarak edinilebilmektedir. Bu araçlar hem dijital delillerin toplanması hem de incelenmesi aşamalarında kullanılmaktadır. Genel olarak kullanım alanlarına göre imaj almak için, canlı analiz için, imaj üzerinden analiz için, veri kurtarmak için, veri silmek için, kullanılan araçlar bulunmaktadır. Bu araçlar teknolojinin gelişimine paralel olarak, adli bilişim alanında gerekli hizmetlerin sağlanabilmesi amacıyla hızla değişmekte ve gelişmektedir. Ancak günümüzde hala istenilen derece çözümlenememiş ve çözüm bulunamayan konular bulunmaktadır. Dijital delil inceleme safhalarında kullanılan cihaz tipleri aşağıdaki gibi sıralanabilir.

2.4.1 Yazma Koruma Cihazları

Yazma koruma cihazları hem delillerin canlı analizi hem de yazılım ile imaj alma işlemleri sırasında kullanılmaktadır. Yazma koruma cihazlarının amacı, delil niteliği taşıma olasılığı olan herhangi bir dijital aletin başka bir cihaza bağlandığında herhangi bir değişime uğramasını engellemektir. Hem donanım hem de yazılım ile yazma koruması sağlanabilmekle beraber en çok tercih edilen yöntem bir donanımın kullanılmasıdır. Dijital deliller üzerinde bir bit verinin dahi değişmesi delil niteliğinin ortadan kalkmasına neden olabilmektedir. Bu yüzden yazma koruma cihazları adli bilişim alanında olmazsa olmaz bir

konumdadır. Yazma koruma cihazları, diğer donanımlardan ayrı olabildiği gibi imaj alma donanımları veya veri çıkarma donanımları gibi cihazların içinde tümleşik olarak da bulunabilmektedir. Çeşitli boyut ve fiyatlarda yazma koruma cihazları bulunabilmektedir. Ayrıca bazı dijital delil inceleme kitleri kendi içinde yazma koruma cihazlarını da sunabilmektedir. En çok kullanılan ve bilinen yazma koruma donanım markaları, Tableau, CRU wiebeteck ve Ultrablock olarak sıralanabilir. Şekil 2.2.'de çeşitli üreticiler tarafından üretilen yazma koruma cihazları gösterilmiştir.



Şekil 2.2. Çeşitli üreticiler tarafından üretilmiş yazma koruma cihazları

2.4.2 Disk Silme Yazılımları

Disklerin silinmesi de adli bilişimde dikkat edilmesi gereken bir husustur. Disklerin silinmesi ilk bakışta anti adli bilişim yaklaşımı olarak algılansa da adli bilişim alanında da kullanımı mevcuttur. Adli delillerin kopyalanması sırasında adli bilişim uzmanları kopyaların kaydedileceği disklere ihtiyaç duymaktadır. Ancak bu disklerin her dava için hiç kullanılmamış yeni diskler olması her zaman mümkün olmamaktadır. Bu durum dijital delillerin yanlış değerlendirmesine yol açabilmektedir. Örneğin daha önceden kullanılmış olan bir diskin içinde kalan veri parçaları yeni davadaki disk ile karışabilir ve diskin içinde aslında bulunmayan verilerin bulunduğu sanılabilir. Bu gibi durumlardan kaçınmak adına adli bilişim alanında kopya için kullanılacak disklerin içinde hiç veri kalmayacak şekilde silinmesi gerekmektedir. Bu işleme wipe adı verilmektedir. Wipe işlemi bir diskin içindeki verileri geri dönüşü olmaksızın silme işlemidir. Bu işlemi gerçekleştiren çeşitli yazılım ve donanımlar bulunmaktadır. Adli bilişim alanında bu işlemler genellikle donanımlar aracılığı ile gerçekleştirilmekle beraber yazılımlar da kullanılabilir. Daha çok donanımların kullanılmasının sebebi, donanımların hız ve kolaylık sağlamasıdır. Sadece disk silmek için geliştirilmiş donanımlar sayesinde aynı anda birden fazla diskin hızlı bir şekilde silinmesi mümkündür. Bunun yanında adli bilişim imaj alma kitleri de disklerin geri dönüşü

olmaksızın silinebileceği modüllere sahiptir. Bunlardan en fazla kullanılanları imaj alma donanımları başlığı altında verilmiştir. Disk silme amaçlı donanımlardan ise en çok kullanılanlar arasında Cru Wiebetech Drive Eraser Ultra, Voom Drivewiper ve Logicube Omniwipe cihazları sıralanabilir. Şekil 2.3.'de çeşitli üreticiler tarafından üretilen disk silme cihazları gösterilmiştir.



Şekil 2.3. Çeşitli üreticiler tarafından üretilmiş disk silme donanımları

2.4.3 İmaj Alma Yazılım ve Donanımları

İmaj alınması dijital delillerin incelenmesinden önce gerçekleştirilen önemli bir adımdır. Delillerin inceleneceği kopyaların oluşturulması bu adımda gerçekleştirilir. Dolayısı ile sağlıklı imaj alma işlemi dijital delil incelemelerinde belirleyici bir etmendir. Bu işlem donanımlar aracılığı ile yapılabildiği gibi yazılımlar ile de gerçekleştirilebilmektedir. Ancak yazılımlar ile gerçekleştirilmesi için dijital delilin yazılımın bulunduğu ortama bağlanması sırasında yazma koruması sağlanamayabilir. Bu işlemi gerçekleştiren birçok lisanslı ve lisanssız yazılım bulunmaktadır. En çok kullanılan imaj alma programları Accessdata FTK Imager, Guidance Software EnCase Forensic Imager, GETDATA Forensic Imager, Linux dd tool ve Guymager Yazılımı olarak sıralanabilir. Bu yazılımlar kullanılırken mutlak suretle bir yazma koruma cihazı kullanılmalı veya yazılım ile yazma koruması sağlanmalıdır.

İmaj alma donanımları ise genellikle imaj alma, klonlama, disk doğrulama, disk silme, disk formatlama, hash (sayısal veri özeti) değeri hesaplama gibi birçok işlemi gerçekleştirebilecek şekilde üretilmiş adli bilişim kitleridir. Adli bilişim alanında en çok kullanılan imaj alma donanımları Tableau TD3/TD2, CRU Wiebetech Ditto/Ditto DX Forensic Fieldstation, Logicube Forensic Dossier, Image Masster Solo gibi donanımlardır. Bu cihazlar Sata, Usb, Firewire, IDE bağlantı portlarını kullanarak çok çeşitli disklerin imajlarını alma özelliğine sahiptir. Ayrıca bazı donanımlar uzak bilgisayar bağlantısı

gerçekleştirerek uzaktaki disklerin imajını alabilmekte ve alınan imajın uzak bilgisayara kaydedilmesine olanak sağlayabilmektedir. Şekil 2.4.'te çeşitli üreticiler tarafından üretilen disk kopyalama cihazları gösterilmiştir.



Şekil 2.4. Çeşitli üreticiler tarafından üretilmiş disk kopyalama donanımları

2.4.3.1 Tableau TD3

Tableau TD3 cihazı en çok kullanılan adli imaj alma cihazları arasında yer almaktadır. Üzerindeki dokunmatik ekran kullanılarak işlemler gerçekleştirilebildiği gibi web tabanlı kullanıcı arayüzü ile de aynı işlemler gerçekleştirilebilmektedir. Ayrıca kullanıcı arayüzü kullanılarak cihaza takılı olan disklerin içi görüntülenebilir ve disk içinden veriler indirilebilir. Kendi üzerinde bulunan USB 3.0 ve SATA bağlantılara ek olarak genişletme modülleri kullanarak IDE/PATA, SAS ve FireWire bağlantı tipine sahip disklerin kopyalarını alabilmektedir. Bu portların hepsi yazma koruma özelliğine sahiptir. Alınan imajlar kendi üzerindeki USB 2.0 ve SATA bağlantılarına ek olarak genişletme modülü ile USB 3.0 tipi bağlantıya sahip cihazlara aktarılabilir. Özel disk kasası yardımı ile aynı anda bir adli kopyayı iki diske birden alma imkânı sağlamaktadır. Alınan bir imajlar veya adli deliller ethernet portu aracılığı ile paylaşım açılabilir. Host Protected Area ve Device Configuration Overlay (HPA/DCO) adı verilen işletim sisteminin okuyamadığı gizli alanlara ulaşım sağlayabilmektedir. MAC cihazların adli kopyalarını oluşturabilmektedir. Ayrıca yapılan tüm işlemlerin log (giriş) kayıtlarını tutmaktadır. Disk klonlama, imaj alma, disk wipe etme, hash değeri hesaplama, disk formatlama, boş disk kontrol etme, adli delil doğrulama gibi işlemlerin tamamını gerçekleştirebilmektedir. Disk wipe etme ve imaj alma işlemlerinde ortalama işlem hızı dakikada 6GB'dır. Oluşturulan imaj dosyaları raw/dd, E01, Ex01 veya dmg formatında olabilmektedir. MD5 ve SHA1 hash değerleri üretmektedir. NTFS, HFS+, exFAT ve FAT32 dosya sistemlerine sahip diskleri hedef disk olarak

tanımlamak ve işlerine adli kopya oluşturmaktadır. Ayrıca cihaza farklı erişim izinlerine sahip profiller tanımlanarak her kullanıcının farklı işlemlere erişimleri kontrol edilebilmektedir. Kullanımının kolay olması cihazın yaygın şekilde kullanılmasını sağlamaktadır. Şekil 2.5.'te Tableau TD3 adli imaj alma cihazı gösterilmiştir.



Şekil 2.5. Tableau TD3 adli imaj alma donanımı

2.4.3.2 CRU Wiebtech Ditto DX Forensic Fieldstation

Ditto DX imaj alma donanımı Ditto donanımının bir yeni versiyonudur. Cihaz üzerinde daha fazla bağlantı noktası olması, işlemci hızının ayarlanabilmesi, performans iyileştirmeleri ve görsel değişiklikler gibi çeşitli geliştirmeler eklenmiştir. Ancak gerçekleştirilebilen işlemler iki cihaz için de ortaktır. Cihaz işlemleri üzerinde bulunan LCD ekrandan takip edilmekte ve üzerinde bulunan dört tane tuş ile yönetilmektedir. Ayrıca üzerinde bulunan ethernet portu kullanılarak web tabanlı kullanıcı arayüzüne bağlanarak işlemlerin gerçekleştirilmesi de mümkündür. Cihaz üzerinde yazma korumalı olarak SATA, eSATA, PATA, USB3.0 / 2.0 bağlantı tipine sahip cihazların bağlanabileceği portlar bulunmaktadır. Bunlara ek olarak genişletme modülleri ile FireWire ve SAS bağlantı tipine sahip cihazlar da bağlanabilmektedir. Disk klonlama, imaj alma, disk wipe etme, hash değeri hesaplama, disk formatlama, adli delil doğrulama, HPA/DCO alanlarını okuma, web portu ile uzak disklere erişim ve paylaşım gibi özelliklere sahiptir. Sata portu işlem hızı 6GB'dır. DD ve E01 formatlarında imaj dosyası oluşturabilmektedir. Lojik imaj işlemi ile sadece ilgilenilen dosyaların imajını oluşturarak zaman kaybı önlenabilmektedir. İşlem hızının yüksek olması ve kullanıcı arayüzünün efektif olması cihazın tercih edilmesini sağlamaktadır. Şekil 2.6.'da CRU Wiebtech Ditto DX adli kopya alma cihazı gösterilmiştir.



Şekil 2.6. CRU Wiebetech Ditto DX adli kopya alma cihazı

2.4.3.3 Logicube Forensic Dossier

Üzerinde bulunan dokunmatik ekran aracılığı ile yönetilen Forensic Dossier donanımı SATA, USB, FireWire ve IDE/PATA bağlantısıyla tipine sahip dijital delillerin adli kopyalarını alabilmektedir. Disk kopyası alma modu klonlama, DD, E01 olabilmektedir. Ayrıca ek dönüştürücüler kullanılarak SAS ve SCSI tipi disklerin de cihaza bağlanması mümkün olmaktadır. Donanım aynı anda birden fazla kopya alabilmesi özelliğiyle öne çıkmaktadır. Diğer cihazlarda da olan aynı anda bir delilin iki kopyasını alabilmesini yanında iki farklı delilin aynı anda kopyasını da alabilmektedir. Veri aktarım hızı dakikada 7 GB'a kadar çıkmaktadır. Cihaz delillerin hash değerlerini MD5 algoritması ile hesaplayabilmektedir. Ayrıca diğer cihazlardan farklı olarak büyük bir diskin imajını bölerek daha küçük iki diske yayma özelliğine de sahiptir. Bu özellik kullanıcılara kullanım kolaylığı sağlamakta ve disk ihtiyacını hafifletmektedir. Forensic Dossier donanımının en önemli özelliği ise imaj alma işlemi yapmaksızın veya imaj alma işlemi devam ederken dahi delil üzerinde anahtar kelimesi arama yapabilmesidir. Cihazın ayrıca diskleri wipe etme ve ethernet portu üzerinden delil diskini paylaşma ve HPA/DCO gizli alanlarına erişme özelliği vardır. Şekil 2.7.'de Forensic Dossier adli kopyalama donanımı gösterilmiştir.



Şekil 2.7. Forensic Dossier adli kopyalama donanımı

2.4.4 Veri Kurtarma Donanımları

Veri kurtarma donanımları adli delil inceleme aşamalarından inceleme safhasında kullanılan cihazlardır. Elde edilen medya ortamındaki verilerin normal olarak okunamadığı durumlarda veri kurtarma donanımları kullanılmaktadır. Verilerin okunamamasının birçok nedeni olabilmektedir. Örneğin bir harddisk düşünülürse, harddiskin fiziksel olarak zarar görmüş olması, PCB kartının zarar görmüş olması, üretici tarafından yüklenen firmware yazılımının bozulması, diskin çizilmesi gibi durumlarda diskin donanım ve yazılımlar tarafından tespit edilmesi zordur. Böyle durumlarda diskin, veri kurtarma donanımları kullanılarak tekrar okunabilir ve üzerinde işlem yapılabilir duruma getirilmesi gerekmektedir. Fiziksel hasar veya parça bozukluklarında bir donör diske ihtiyaç duyulur. Hasarlı diskin hasarlı olan parçaları tamir edilemiyorsa veri ihtiva eden parçaları hariç diğer parçaları donör diskin parçaları ile değiştirilerek diskteki verilere erişimi sağlanır. Firmware, dosya sistemi bozuklukları ve disk çizikleri gibi problemlerin çözümü için ise geliştirilmiş özel donanım ve kitler mevcuttur. Bu donanımlar, disklerin firmware, servis alanı bilgileri, okuma kafası durumları, disk plaka durumları gibi birçok bilgiye erişebilmekte ve bu bilgileri değiştirebilmektedir böylelikle okunabilen diskin içindeki verileri dışarı aktarabilmektedir. Günümüzde bu amaçla geliştirilmiş çok fazla donanım mevcut olmamakla birlikte ACE Laboratory ve Dolphin Data firmalarının ürettiği PC 3000 ve DFL donanımları adli bilişim alanında yaygın olarak kullanılmaktadır. Şekil 2.8.'de Dolphin ve PC 3000 cihazları gösterilmiştir.



Şekil 2.8. Çeşitli üreticiler tarafından üretilmiş veri kurtarma donanımları

2.4.5 Dijital Delil İnceleme Yazılımları

Dijital delil inceleme yazılımları adli delil inceleme aşamalarından inceleme ve çözümleme adımında kullanılmaktadır. Elde etme ve inceleme aşamasında adli delil niteliği taşıyan medya cihazından elde edilebilecek maksimum derecede veri elde edilmek amaçlanmaktadır. Çıkarılan veriler çözümleme aşamasında bilgiye dönüştürülmektedir. Yani incelenen dava ile ilişkili olan veriler belirlenir ve raporlanmaya hazır hale getirilir. Özellikle verilerin dijital ortamdaki silinmiş bile olsa kazınması, indexlenmesi, filtrelenmesi, sonuçlar arasında arama yapma gibi işlemleri gerçekleştirebilmek için adli delil inceleme yazılımlarından yararlanılmaktadır. Bu yazılımlar genellikle bir imaj dosyası üzerinde çalıştırılmakta veya ilk olarak diskin imajının alınması için kullanılmaktadır. Günümüzde disk boyutlarının çok büyük olması, dosya tiplerinin çok fazla olması gibi nedenlerden dolayı adli delil inceleme yazılımlarının kullanılması adli delil inceleme sürecinde mecburi hale gelmiştir. Açık kaynaklı adli bilişim yazılımları bulunabileceği gibi birçok ticari yazılım açık kaynaklı yazılımların üzerine tercih edilmektedir. Ticari yazılımların daha fazla tercih edilmesinin çeşitli nedenleri vardır. Bunlardan bazıları; hızlı bir şekilde teknik destek alınabilmesi, sürekli güncellemeler ile teknolojiye gelişmelere ayak uydurabilme, eğitim ve sertifika programlarının olması, daha çok ülkemizde yaygın olarak kullanılan Microsoft Windows işletim sistemi için üretilmeleri olarak sıralanabilir. Bunun yanında açık kaynaklı yazılımların da tercihe edilmesini sağlayan nedenler bulunmaktadır. Bunlar; ticari yazılımların çok yüksek maliyete sahip olması, satın alınan lisansların sürekli yenilenmesi gerekmesi, sertifika ve eğitim programlarının çok maliyetli olması, program üzerinde değişiklikler yapılamaması olarak sıralanabilir. Bu nedenlerden dolayı, daha uygun bir bütçe ile adli inceleme gerçekleştirilmek istendiğinde açık kaynaklı yazılımlar tercih edilmektedir. Piyasada birçok adli inceleme yazılımı yer almaktadır. Bu yazılımlar yaklaşık olarak aynı kabiliyetlere sahip olmakla birlikte, birbirleri üzerinde işlem hızı, ek modülleri, arayüzü gibi çeşitli üstünlükle sahip olabilmektedir. Hangi yazılımın seçileceği incelemecinin kullanım alışkanlığı ve bütçesine göre değişiklik göstermektedir. Günümüzde adli inceleme yazılımlarından en çok tercih edilen yazılımlar arasında; EnCase Forensic, Forensic Tool Kit, Magnet AXIOM, Forensic Explorer, Oxygen Forensics, X-Ways Forensics, Paraben E3, Autopsy, SANS Investigative Forensic Toolkit (SIFT) sayılabilir. Bu yazılımlar; veri indexleme ve veri tipine göre sınıflandırma, veri arama, veri kurtarma, gömülü dosyaları bulma, dosya imza analizi yapma, veri kazıma, sanal bilgisayarları inceleme, canlı analiz

yapma, hash doğrulama, hash filtreleme, mobil cihaz rootlama, mobil cihaz inceleme, bulguların raporlanması gibi birçok özelliğe sahiptir. Ancak tüm yazılımların ortak özelliği inceleme aşamasında kullanılmasıdır. Adli delil inceleme yazılımlarının önemli özelliklerinden bazıları Tablo 2.1’de verilmiştir. Tablodan görüldüğü gibi verilere ulaşmak ve içeriğine bakmaksızın filtrelenmesi için kullanılan birçok özellik yazılımların içerisinde mevcuttur. Ancak, verilerin içerdiği verilere göre sınıflandırma yapabilen herhangi bir özellik bu yazılım araçlarında mevcut değildir. Bazı yazılımlarda bulunan renk analizi özelliği kısmen içeriğe göre resim verileri içerisinde arama yapsa da oldukça ilkel bir yöntemdir. Bu özellik görüntü dosyalarında yoğun olarak bulunan renklere göre görüntüleri ayırtmaktadır. Örneğin, ten rengine benzer rengin yoğun olarak bulunduğu resimlerin çıkarılması sağlanabilirken muz gibi sarı ağırlıklı nesnelerin ağırlıkta olduğu resimler, bej rengi duvarlar gibi nesneler de saptanmaktadır. Ayrıca bu işlem inceleme süresini çok fazla uzatmaktadır. Bu durum, dijital verilerin çok fazla olduğu günümüzde adli delil incelemelerinin daha uzun ve hataya açık olmasına sebep olmaktadır.

Tablo 2.1. Yaygın olarak kullanılan adli delil inceleme yazılımlarının özellikleri

Yazılım	İmaj Alma	Veri Kazıma	İndexleme	Filtreleme	Hash Kontrolü	Renk Analizi	Verilerin İçeriğe Göre Otomatik Anlamlandırılması
Encase	+	+	+	+	+	+	-
FTK	+	+	+	+	+	+	-
Forensic Explorer	+	+	+	+	+	+	-
Paraben	+	+	+	+	+	-	-
Autopsy	+	+	+	+	+	-	-
Oxygen Forensic	+	+	+	+	+	-	-
Celebrite	+	+	+	+	+	+	-
Magnet Axiom	+	+	+	+	+	+	-

Adli inceleme yazılımlarının adli inceleme sürecindeki görevi, dijital medya üzerinden delil niteliği taşıma olasılığı olan tüm verileri çıkarmaktır. Çıkarılan veriler incelenen dava ile ilgili veya ilgisiz olabilmektedir. Elde edilen veriler çözümleme aşamasında adli bilişim uzmanları ve kolluk görevlileri tarafından değerlendirilir ve incelenen dava ile ilişkilendirilen

veriler çıkartılır. Bu noktada çözümleme aşamasının iyileştirilmesi için yapılacak çalışmalara ihtiyaç duyulduğu görülmektedir.

2.5 Adli Bilişim İncelemelerinin Zorlukları ve Önerilen Çözüm

21. yy'ın başlarından bu yana teknolojiadaki hızlı gelişime bağlı olarak dijital cihazlar kişilerin günlük hayatında daha fazla yer kaplamaya başlamış ve günümüzde günlük hayatın değişmez bir parçası haline gelmiştir. Bu gelişim, çok daha karmaşık yapıya sahip dijital cihazların, çok daha fazla veri depolanabilen ortamların ve depolanabilecek veri çeşidinin artmasını da sağlamıştır. Aynı zamanda bir soruşturmada dijital delillerin bulunduğu durumlar artış göstermiştir. Bu nedenlerden ötürü, adli bilişim alanında mevcut durumu değerlendirmek önemlidir. Bulut tabanlı hizmetler, nesnelerin interneti kullanımı, anti adli bilişim teknikleri, dağıtılmış ve yüksek kapasiteli saklama ve ilgili cihazların hacmi, bu dijital kanıtların elde edilmesi, depolanması ve analizi için yeni ve sorunlar ortaya çıkarmaktadır. Ortaya çıkan sorunların sonucunda adli delil inceleme süresinin uzaması kaçınılmaz olmuştur [21].

Raghavan adli bilişim alanındaki zorlukları beş başlık altında toplamıştır [22]. Bunlar:

-Karmaşıklık problemi: Artan veri hacmi ve heterojenlik dolayısı ile elde edilen verilerin sadeleştirilmesi gerekmektedir. Bu da adli delil incelemesini yavaşlatmaktadır.

-Çeşitlilik problemi: Gittikçe artan işletim sistemi, dosya tipi gibi veri ve veri tipinin doğal bir sonucu olarak çok sayıda farklı veri tipinin incelemesinin yapılması zorluklara sebep olmaktadır. Veri inceleme yöntemlerinde bir standardın oluşturulamaması bu problemin çözülmesini zorlaştırmaktadır.

- Tutarlılık problemi: Bu tez çalışmasında çözüm aranan problemdir. Mevcut adli delil inceleme yazılımlarının sadece dijital ortam üzerinden veri parçalarını bulmak amacıyla kullanılabilmesi, soruşturma ile ilgili başka bir yardım sağlamamasından kaynaklanmaktadır. Yazılımlar tarafından bulunan delil niteliği taşıyan verilerin çözümleme aşamasında gözden kaçırılması veya kasten manipüle edilmesi sonucu bu problem ile karşılaşılmaktadır.

-Hacim sorunu: Veri depolama teknolojilerinin yüksek kapasiteli disk ve bulut depolama alanları sayesinde çok büyümesi ve bilgi depolayan aygıtların sayı ve çeşitlilik olarak artması beraberinde hacim sorununu getirmektedir. Bütün bu büyük miktarlardaki verinin analizi için otomasyonun mevcut sistemlerde yeterli olmaması da bu sorunu

tetiklemektedir. Bu tezde geliştirilecek uygulama ile hacim probleminin çözümüne de, çok büyük katkı sunulacaktır.

-Saat dilimi problemi: Çok farklı kaynaklardan elde edilen saat dilimi, zaman damgası dönüşümleri, yaz-kış saati uygulamaları ve tarih yazım formatları delillerin zaman anlamında incelenmesini zorlaştırmaktadır.

Diğer birçok çalışmada araştırmacılar daha spesifik problemlerden bahsetmiştir. Ancak bunlar genel olarak yukarıdaki gibi sınıflandırılabilir [1,23-26].

Yukarıda bahsedilen zorlukların yanında hukuki zorluklar da adli delil incelemelerinde problemlere neden olmaktadır. Örneğin, adli delil incelenirken bazı durumlarda daha fazla veriye ulaşabilmek amacıyla delil üzerine ek yazılımların eklenmesi gerekmektedir. Bazen bu yöntem aranan veriye ulaşmanın tek yolu olmaktadır. Ancak, dijital delillerin ilk alındığı andan itibaren hiç bir şekilde içeriğinin değişmemesi hukuken belirlenmiş bir kuraldır. Aksi takdirde, dijital delilin bütünlüğü bozulmuş sayılmaktadır. Çünkü dijital delilin elde edilme anı ile inceleme safhası arasında sayısal özet (hash) değeri değişmiştir. Bütünlüğü bozulan dijital deliller ise suça ilişkin deliller olarak kullanılamamaktadır. Bu gibi hukuki prosedürler de dijital delil inceleme sürecini zorlaştırmakta ve yavaşlatmaktadır.

Bu tez çalışması adli delil incelemelerinde yaşanan iki temel problemin çözümü için çalışmaları içermektedir. Dijital delil inceleme süreçlerinin otonom hale getirilmesi ve dijital delil inceleme araçlarının sadece inceleme aşaması yerine çözümleme aşamasında da incelemeciye kolaylık sağlaması amaçlanarak uygulamalar geliştirilecektir. Bu uygulamalar, ilk olarak resim verileri üzerinde gerçekleştirilecektir. Bunun nedeni mevcut dijital delil inceleme yazılımlarının ofis dokümanları üzerinde arama yapabilse bile görsel verilerin tamamen insan algılarına dayalı olarak incelenmesidir. Dolayısı ile görsel verilerin incelenmesi aşamasında hem inceleme süresi uzamakta hem de insan kaynaklı hatalar ortaya çıkabilmektedir. Yapılan uygulamalarda bu durumun önüne geçilmesi ve dijital delil inceleme süreçlerinin hızlandırılması amaçlanmaktadır. Bu amaca yönelik olarak son yıllarda popüler olan derin öğrenme algoritmaları kullanılacaktır. Bu yöntemin seçilmesini nedeni, günümüzde tanıma ve saptama gibi uygulamalarda derin öğrenme yöntemlerinin insan algıları seviyesinde hatta daha fazla doğruluk oranına sahip sonuçlar vermekte olmasıdır. Bu doğrultuda yapay sinir ağı tarafından bir suçun araştırılması aşamasında görsel veriler arasında en çok aranan nesnelerden olan silah nesnesinin hızlı bir şekilde saptanması amaçlanmıştır. Bunu yaparken derin öğrenme yöntemleri kullanılacak ve içerisinde silah olan veriler hızlı bir şekilde tespit edilerek kullanıcıya sunulacaktır. Bunun için nesne tespiti

algoritmaları incelenecek ve adli delil incelemelerinde hız ve doğruluk oranlarına dikkat edilerek inceleme sürecinde kullanılabilecek olan algoritmalar oluşturulacak olan veri setlerine uygulanarak uygulamalar gerçekleştirilecektir. Gerçekleştirilen uygulamaların, günümüzde kullanılan adli delil inceleme yazılımlarına eklenti olarak eklenilmesi düşünülmektedir. Bu işlem dijital delilin incelenme hızını oldukça artıracaktır. Örneğin, Roussev ve ark. FBI’ın adli bilişim laboratuvarlarındaki davalarda incelenen ortalama veri miktarının 2011 yılında 559 GB olduğunu belirtmiştir [27]. 500 GB boyutunda bir harddiskin içinde ortalama 65.000.000 resim dosyası bulunabilmektedir (22 MP, sıkıştırılmış %100 kaliteli JPEG dosyası). Bu veriler içinden yalnızca 10.000 tanesinin aranan delili içerdiği düşünüldüğünde inceleme süresi insan algılarının yorulması göz ardı edilerek kabaca %99.9 oranında azalmaktadır. Dijital inceleme sürecini bu denli hızlandıran bir araç günümüzde bulunmamaktadır.

3. MAKİNE ÖĞRENMESİ

Makine öğrenmesi, bir makinenin insanlar gibi daha önce görmüş olduğu örnekleri kullanarak yeni karşılaşılan durumlarda yeni sonuçlar üretebilmesini sağlamayı hedeflemektedir. Bu amaç doğrultusunda bir makineyi asıl kontrol eden yapı olan yazılımlar aracılığı ile çeşitli algoritmalar geliştirilmiş ve öğrenen bir yapı oluşturulmaya çalışılmıştır. Bu isteğin asıl nedeni karmaşık ve uzun işlemlerin makineler tarafından otomatik olarak yapılmasının istenmesidir. Çoğu durumda bir problemin çözümü matematiksel bir denklemin sonucu olabilmektedir. Bu gibi durumlarda yazılımlar aracılığı ile problemin çözümü çok basit olmaktadır. Ancak cevap aranan problemin matematiksel olarak modellenemediği durumlarda yazılımlar ile problemin çözümü zorlaşmaktadır. Böyle durumlarda makine öğrenmesi ve optimizasyon algoritmalarından yararlanılmaktadır. Makine öğrenmesi günümüzde, reklam, e-ticaret, biyometrik kimliklendirme, veri madenciliği, veri güvenliğinin sağlanması, otonom taşıt sistemleri gibi daha birçok alanda kullanılmaktadır. Makine öğrenmesinin bir diğer kullanım alanı da bilgisayar görmesidir. Bilgisayar görmesi, bir bilgisayar sisteminin, kamera veya kendisine gönderilen görüntüler aracılığı ile dış dünyadaki olayları yorumlaması ve ya öğrenmesini amaçlayan makine öğrenmesi koludur. Öğrenme algoritmaları bilgisayar görmesi alanında yaygın olarak kullanılmaktadır. Bu bölümde, bilgisayar görmesi ile ilgili çalışmalardan önce makine öğrenmesinin temelleri ortaya konmuştur.

Makine öğrenmesi, formüle dökülmesi zor olan problemlerin modellenenebilmesi amacıyla ortaya çıkmıştır. Klasik bilgisayar programları, bir görevi gerçekleştirmek için programlanmış ve insan gözetimi ile çalıştırılan yazılımlardır. Ancak, makine öğrenmesinde insan tarafından yapılan bazı kısımların yerini öğrenme algoritmalarının çıktıları almaktadır [28]. Hesaplama kapasitesi ve verilerin kullanılabilirliği arttıkça, makine öğrenmesi yıllar içinde neredeyse her alanda kullanılabilir derecede pratik hale gelmiştir. Ve günümüzde farklı kullanım alanlarında kullanılmaya devam etmektedir. Makine öğrenmesinin gelişmesi ve kullanım alanlarının genişlemesi ile bu alanda çeşitli yöntemler ve alt dallar oluşmuştur.

3.1 Genelleştirme

Öğrenme algoritmasına girdi olarak verilen, eğitim verilerinin, olası tüm girdileri içinde barındırması imkânsızdır. Bu nedenle, öğrenme algoritması, daha önce görmemiş olduğu verileri doğru şekilde işlemesi ve tanınması için bir genelleme yapmaya ihtiyaç duyar. Çok basit bir model kullanılırsa, genellenmek istenen verinin önemli noktaları istemeden göz ardı edilebilir. Bunun yanında çok karmaşık bir genelleme yapılırsa, aslında verinin kendine ait olmayan, gürültü ve önemsiz ayrıntılar da genellemeye dâhil edilir ve bu gereksiz ayrıntılara sahip olmayan veriler için olumsuz bir genelleme olmuş olur. Bu iki durumda ya öğrenme gerçekleşemez ya da ezberleme meydana gelerek sadece eğitim veri setinde bulunan veriler öğrenilir.

Genellikle ezberleme (overfitting-aşırı eğitim), eğitim setinin az olduğu ve çok karmaşık bir yöntemin kullanıldığı durumlarda gerçekleşir. Bu durumda eğitimde kullanılan veriler öğrenilmiş olur ancak bu verilerin ortak noktaları öğrenilmemiş olmaktadır. Bunun tersi olarak basit bir yöntem ve çok verinin kullanıldığı durumda da yeterli öğrenme gerçekleşmediğinden doğru bir genelleme yapılamaz.

Bu gibi durumlarda algoritmanın başarısını test etmek amacıyla hataların nitelikleri ve sayısından yararlanılabilir. Hataların matematiksel olarak modellenmesi için ortalama hata kareleri toplamı gibi bir kayıp fonksiyonu kullanılmaktadır. Eğitim süresince amaç bu değeri en aza indirmektir. Çünkü eğitimin ne kadar doğru sonuç verdiği kayıp fonksiyonunun sonucunun sıfıra ne kadar yakın olduğu ile ölçülmektedir. Ancak, bu değer sıfır olduğunda aşırı eğitim gerçekleşmiş olacağından sıfıra oldukça yakın ama eşit olmayan bir sonuç eğitimin başarılı bir şekilde sonuçlandığını gösterir.

3.2 Özellikler

Makine öğrenmesi yöntemlerinde, eğitim için kullanılan verilerin hemen hemen her zaman bir ön işleme aşamasından geçirilmesi gerekmektedir. Genellikle eğitim setindeki verilerin tam boyutta kullanılması mantıksız veya imkânsız olmaktadır. Bu nedenle, işlem karmaşıklığından kurtulmak veya daha başarılı bir öğrenme sağlamak amacıyla verilerin daha basit bir değişken uzayına ötelenmesi gerekmektedir. Verinin daha basit bir değişken uzaya ötelenmesine özellik çıkarımı denmektedir. Özellik çıkarımını gerçekleştirmesi için çeşitli yöntemler önerilmiştir. Bu yöntemlerden çıktı olarak alınan veriler öğrenme

algoritmalarına girdi olarak verilir. Böylece öğrenilmesi istenen veri özellikleri yardımıyla öğrenilmiş olmaktadır.

Geçmişte, özellik çıkarıcılar kişiler tarafından belli özellikler hedef alınarak oluşturulmuştur. Bu yaklaşımdaki problem, hangi özelliklerin daha önemli olduğunun her zaman önceden bilinmemesi idi. Ancak, günümüzde makine öğrenmesinde trend, eğitim veri seti için en doğru ve eksiksiz özellik çıkarma biçiminin de makine öğrenmesi ile gerçekleştirilmesi yönündedir [28].

3.3 Makine Öğrenmesi Tipleri

Makine öğrenmesinin tipik bir kullanımı denetimli öğrenmedir [29]. Bu öğrenme tipinde yazılıma, öğrenmesi istenen açıklamaları yapılmış veya etiketlenmiş veriler gönderilir. Daha sonra, yazılım benzer bir durum ile karşılaştığında öğrenme gerçekleşmiş ise aynı etiket veya açıklamanın çıktı olarak alınması beklenmektedir. Örneğin, nesne tespiti uygulamalarında tespit edilmesi istenen nesnenin resimlerde işaretlenmiş olduğu veriler algoritmaya gönderilir. Algoritma bu verileri yorumlayarak daha önce hiç görmemiş olduğu bir resimde de tespit edilmesi gereken nesneyi tespit edebilmektedir. Bu şekilde eğitimin sonucunun testten önce algoritmaya öğretildiği/verildiği eğitim tipine denetimli öğrenme adı verilmektedir.

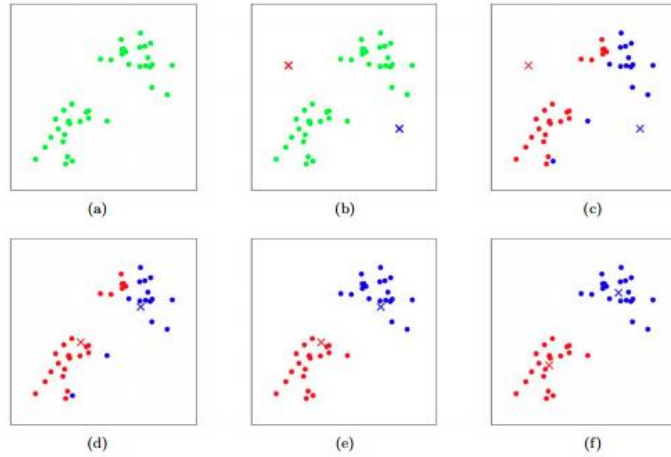
Sınıflandırma ve regresyon da çok önemli makine öğrenmesi konularıdır [29]. Sınıflamada algoritma, eğitim verilerine dayalı olarak yeni bir verinin doğru sınıfını tahmin etmeye çalışır. Regresyonda, ayrı sınıflar yerine, algoritma sürekli bir çıktı tahmin etmeye çalışır. Örneğin, iki sınıflı bir resim sınıflandırma uygulamasında algoritma yeni gördüğü bir veriyi birinci veya ikinci sınıf olarak etiketlerken, regresyon uygulamasında algoritma her iki sınıf için de bir sayısal değer üretecektir. Bu sayısal değer göz önünde bulundurularak algoritma gerekli karar ve işlemleri gerçekleştirecektir. Yani algoritmanın ürettiği çıktının kategorik olması istendiği durumlar sınıflandırma işlemleri olurken, nümerik değerlerin beklendiği uygulamalar regresyon uygulamalarıdır denebilir.

Denimsiz öğrenmede ise verilere ait sınıf veya açıklamalar algoritmaya verilmemektedir. Kümeleme olarak da isimlendirilen bu yöntemde amaç etiketleri bilinen verilerin sınıflandırılması değil, karmaşık olarak verilen verinin kendi içinde benzer özellikleri taşıyan verilere göre kümelenmesidir [29]. Özellikle derin öğrenme yöntemlerinin gelişmesi ve yaygınlaşmasıyla birlikte, denetimli öğrenme algoritmalarında, verilerin önemli

özelliklerinin çıkarılması için denetimsiz öğrenmenin kullanılması yaygın bir uygulama haline gelmiştir [30].

3.3.1 K-Means Algoritması

K-means algoritması denetimsiz bir öğrenme algoritmasıdır. Bu yöntem il olarak MacQueen [31] tarafından kullanılmıştır. Bu tip yöntemler kümeleme yöntemleri olarak da ifade edilmektedir. K-means algoritmasında amaç hangi gruba ait olduğu bilinmeyen bir veri setini benzerliklerinden yararlanarak k tane ayrı kümeye ayırmaktır. Hangi elemanın hangi kümeye dâhil olacağına belirlenmesi için öncelikle rastgele küme merkezleri seçilir. Daha sonra her küme merkezine en yakın elemanlar o kümeye dâhil edilir. İterasyonun son adımı olarak küme merkezleri, kendi kümesine dâhil olan elemanların ortasına/ortalamasına kaydırılır. Bu işlem küme merkezleri iterasyonlar arasında yer değiştirmemeye başlayana kadar devam edilir. Böylelikle kümeleme işlemi sona ermiş olur. Daha sonra, veri setine yeni bir eleman eklendiğinde hangi küme merkezine en yakın ise o kümeye dâhil edilir. Algoritmanın çalışma adımları Şekil 3.1.'de gösterilmiştir.



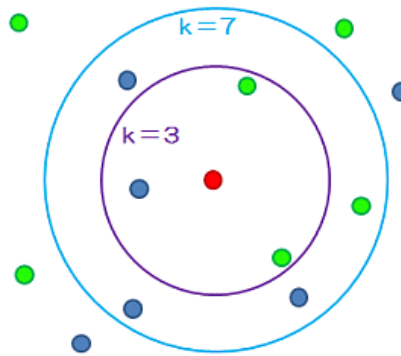
Şekil 3.1. K-means algoritması işlem adımları

K-means algoritması en popüler kümeleme algoritmasıdır. Ancak bu algoritmanın da bazı zayıf yönleri bulunmaktadır. İşlem karmaşıklığı olduğundan dolayı, kullanılan uzaklık algoritmasına göre bir yerel optimum noktasında algoritma durabilir. Bu algoritma yalnızca ortalaması alınabilen özellikler için uygulanabilir. Ayrıca, veri setinin kaç bölüme ayrılacağı kullanıcı tarafından belirlenmek zorundadır; optimum ayırma miktarını algoritma belirlememektedir. K-means algoritması uç değerler ile karşılaştığında kırılgan bir yapıya

sahiptir. Aslında bir kümeye dâhil olması gereken veri küme merkezinden çok uzakta olduğunda küme merkezini kendine doğru çekmekte ve yanlış kümeleme yapılmasını sağlayabilmektedir. Bu zayıflıklarının yanında k-means günümüzde hala en yaygın olarak kullanılan kümeleme algoritmasıdır.

3.3.2 K-NN Algoritması

K-Nearest Neighbors algoritması, 1967 yılında Cover ve Hart tarafından önerilmiştir [32]. K-NN günümüzde en sık kullanılan denetimli sınıflandırma algoritmalarındandır. Bunun nedeni uygulamasının basit ve anlaşılır olmasıdır. Algoritmada ilk olarak etiketlenmiş eğitim setinin girdi olarak verilir. Daha sonra sınıflandırılması istenen test verilerinin, eğitim setleri ile olan öklit uzaklıkları hesaplanır. Başka uzaklık ölçüm yöntemleri de mevcuttur, ancak öklit uzaklığı anlaşılabilirliği, verimliliği ve kolay uygulanması bakımından tercih edilmektedir [33]. Tüm eğitim seti ile test elemanı arasındaki uzaklıkların hesaplanmasından sonra, kullanıcı tanımlı olan k tane test verisine en yakın eğitim verisi tespit edilir. Bu verilerden etiketi en fazla tekrar eden sınıfa test elemanı dâhil edilir ve sınıflandırma etiketi daha önce görülmemiş test verisine atanmış olur. Farklı k değeri için algoritmanın sınıflandırma Şekil 3.2.'de gösterilmiştir.



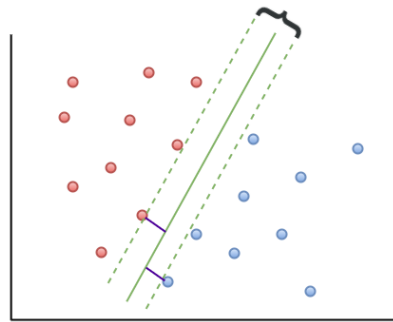
Şekil 3.2. K-NN algoritması

K-NN algoritması uygulanması çok güçlü ve açık olan, parametrik olmayan tembel bir öğrenme algoritmasıdır. Parametrik olmayan algoritmalar, veri dağılımı hakkında herhangi bir ön yargıda bulunmazlar. Tembel bir öğrenme algoritma olmasının anlamı, herhangi bir genelleme yapmak için eğitim veri noktalarını kullanmamasıdır. Başka bir deyişle, açık bir eğitim aşaması yoktur veya çok azdır. Bu, eğitim aşamasının oldukça hızlı olduğu anlamına

geliyor. Genelleme yapılmaması, KNN'nin tüm eğitim verilerini muhafaza ettiği anlamına gelir. Daha doğrusu, test aşamasında tüm eğitim verilerine ihtiyaç duyulmaktadır. Bu durum bir dezavantaj olarak kabul edilebilir. K-NN'nin en büyük dezavantajlarından biri, büyük ölçekli ve yüksek boyutlu veri setleri için verimin yetersiz olmasıdır. Çünkü en yakın k tane eğitim seti verisinin bulunması aşamasında tüm eğitim setinin, sınıflandırılacak olan elemana uzaklığı hesaplanmaktadır ve dolayısı ile algoritmanın ihtiyaç duyduğu işlem yükü artmaktadır. Ayrıca, k parametresinin belirlenmesi bir diğer problemdir. Optimum sınıflandırma performansını veren k parametresi optimizasyon kullanılarak veya uygulama yoluyla belirlenmesi gerekmektedir.

3.3.3 Destek Vektör Makineleri

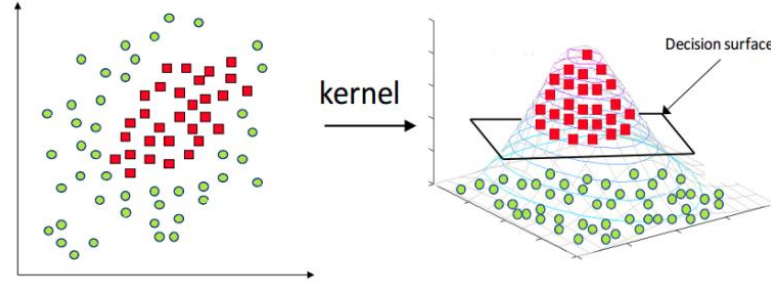
Destek vektör makineleri (DVM), denetimli veri sınıflandırma algoritmalarıdır. Vladimir Vapnik ve arkadaşları tarafından ortaya konulmuş ve geliştirilmiştir [34,35]. DVM kendisine verilen etiketli verileri kullanarak yeni gelen test verilerini doğru sınıflandırmak için kullanılacak bir düzlem oluşturur. Bu düzlem özellik uzayını optimum şekilde ikiye bölmeyi amaçlamaktadır. Daha sonra test verilerinin düzlemin ne tarafında kaldığına bakılarak sınıflandırma işlemi gerçekleştirilir. DVM modelinde amaç iki farklı sınıfa ait birbirine en yakın örneklerin arasından paralel olarak çizilebilecek en uzak iki doğrunun çizilmesidir. Birbirine en yakın bu örneklere destek vektörleri denmektedir. Destek vektörleri üzerinden geçen bu iki doğrunun tam ortasından ise ayırıcı düzlem geçmektedir.



Şekil 3.3. DVM sınıflandırıcı

DVM, normalde iki sınıfı ayırmak için tasarlanmış olsa da ikiden fazla sınıf için sınıflandırma işlemi de gerçekleştirilebilmektedir. İkiden fazla sınıflar için, bire karşı tümü yöntemi kullanılarak, sınıfları ikili ikili ayıran birden fazla düzlem elde edilir. Veri setinin

mevcut durumda ayrılamaz olması durumunda da çekirdek(kernel) denilen çarpanlar ile veri seti ayrılabilen bir boyuta taşınır ve düzlem ile ayrılır [34].

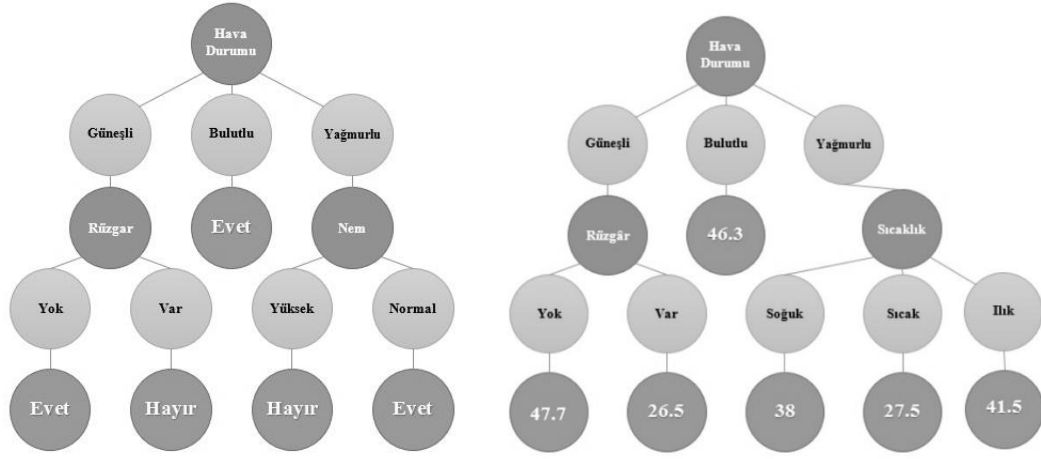


Şekil 3.4. DVM kernel uygulanma işlemi

DVM, çok boyutlu uzayda hatta boyut sayısının örnek sayısından fazla olduğu durumlarda dahi başarılı sonuçlar vermesi, karar için sadece destek vektörlerini kullanarak hafızayı efektif kullanması, çeşitli çekirdek fonksiyonları kullanarak farklı durumlardaki veri setlerini sınıflandırabilmesi gibi nedenlerden dolayı günümüzde yaygın bir şekilde kullanılmaktadır. Bunların yanında, kullanılan veri seti için kernel seçimini zor olması, büyük veri tabanlarında işlem yükünün artması, karar analizinde bir kararın neden verildiğinin analizinin zor olması gibi durumlar DVM'nin zayıf yönleri olarak sayılabilir [36]. DVM'nin elemanları ayırma mantığı Şekil 3.3'de, kernel uygulanması ise Şekil 3.4'te gösterilmiştir.

3.3.4 Karar Ağaçları

Karar ağaçları, sınıflandırma ve regresyon işlemleri için yaygın olarak kullanılan makine öğrenmesi modelleridir [37]. Esasen, bir "if-else" soru hiyerarşisini öğrenirler ve defaten tekrarlanan bu soruların sonunda bir karara varılır. Ağaç sorgusu sonunda çıktı olarak belirli etiket değerleri elde ediyorsa buna sınıflandırma ağacı denir; bu ağaç yapılarında yapraklar sınıf etiketlerini temsil eder ve dallar bu sınıf etiketlerine götüren özelliklerin birleşimlerini temsil eder. Hedef değişkenin sürekli değerler (genellikle reel sayıları) alabileceği karar ağaçlarına, regresyon ağaçları denir. Karar analizinde, karar ağacı, kararları ve karar vermeyi görsel ve açık bir şekilde temsil etmek için kullanılabilir. Şekil 3.5'te karar ağaçlarının sonuç üretme adımları gösterilmiştir.



Şekil 3.5. Sınıflandırma ağacı ve regresyon ağacı

Karar ağaçları eğitilirken, eğitim kümesi yakın belli özelliklere göre alt kümelere bölünür ve bu işlem her alt küme için özyinelemeli olarak tekrarlanır. Alt sınıflara bölme işleminin herhangi bir işlem gerçekleştiremediği anda eğitim tamamlanmış olur [38].

Karar ağaçlarının uygulanması ve anlaşılmasının kolay olması, verilerin ön işlemeye çok az ihtiyaç duyması, hem sınıflandırma hem de regresyon için kullanılması, her adımının görüntülenebilir olması, artan veri miktarına karşılık hızlı bir şekilde sınıflandırma yapabilmesi karar ağaçlarının avantajları arasındadır. Ancak, karar ağaçlarının yapısı gereği aşırı eğitime yatkındır. Bu nedenle eğitimden önce veya sonra gereksiz değişkenlerin çıkarılması işlemi uygulanması gerekmektedir. Karar ağaçlarının en önemli problemi aşırı eğitime eğilimli olmasıdır.

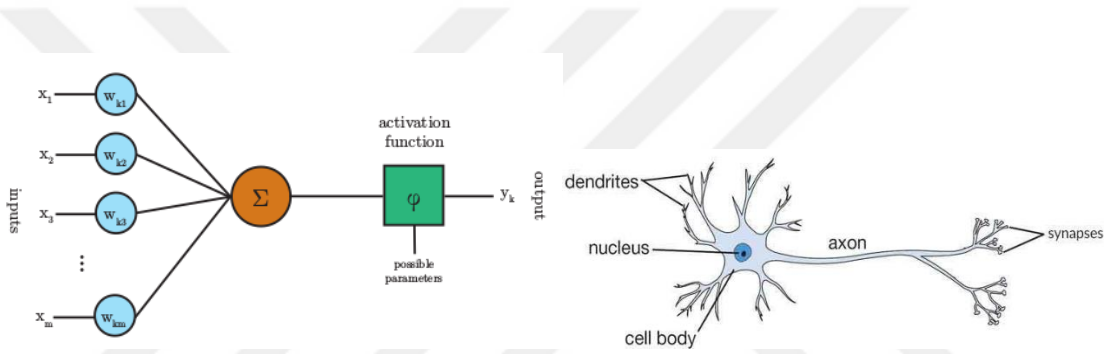
3.3.5 Yapay Sinir Ağları

Yapay sinir ağları, çok bilinen ve kullanılan bir makine öğrenmesi modelidir. İnsan beyninin sinirsel davranışlarını taklit etmek amacıyla oluşturulduklarından ‘yapay’ sinir ağları olarak adlandırılmışlardır. Yapay sinir ağlarının özelleştirilmiş bir hali olan konvolusyonel sinir ağları (CNN), bu tezin uygulanmasında kullanılan ana yöntemdir. Ancak CNN konusuna giriş yapmadan önce, bu bölümde sinir ağlarının çalışma mantığı ve tarihi incelenecektir.

3.3.5.1 Yapay Sinir Ağlarının Temelleri

Yapay sinir ağlarına öncü olan çalışmalar olarak, 1943'de Warren McCulloch ve Walter Pitts tarafından sunulan eşik mantık birimi ve 1957'de Frank Rosenblatt tarafından tasarlanan Perceptron [39] kabul edilebilir.

Biyolojiden esinlenmiş olsa da, yapay nöronlar ile biyolojik nöronlar veya sinir bilimi arasındaki bağlantıyı aşırı vurgulamak yanıltıcı olabilir. İnsan beyni birbirine paralel olarak çalışan yaklaşık 100 milyar nöron içerir [40]. Yapay sinir ağları tam olarak paralel çalışmayan bilgisayarlarda uygulanan matematiksel fonksiyonlardır. Sinir ağlarına yönelik araştırmalar çoğunlukla biyoloji yerine mühendislik ve matematik alanlarındaki gelişmelere dayanmaktadır [28].



Şekil 3.6. Yapay ve biyolojik nöron yapısı

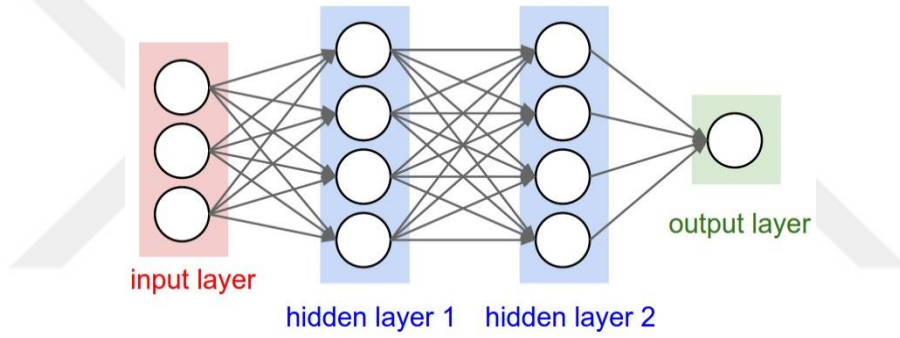
McCulloch-Pitts modeline dayanan yapay bir nöron Şekil 3.6'da gösterilmiştir [29]. Şekilde, yapay nöron m tane giriş parametresi almaktadır (x_j). Nöron (k), alınan her bir giriş parametresi için bir ağırlık parametresine sahiptir (w_j). Bu sistemde, girişler ve ağırlıklar doğrusal olarak birleştirilir ve toplam işlemine tabi tutulur. Bu toplam, daha sonra, nöronun çıkışını (y_k) üreten aktivasyon fonksiyonuna (ϕ) gönderilir. Bu işlem Denklem 3.1'deki gibi formülize edilebilir.

$$y_k = \phi(s_k) = \phi\left(\sum_{j=0}^m w_{kj}x_j\right) \quad (3.1)$$

Nöronun, her bir giriş için istenen çıktının üretilmesi amacıyla, ağırlıklar dikkatle seçilerek eğim gerçekleştirilir. Birden fazla nöron kullanılarak oluşturulan yapılara sinir ağı denmektedir.

3.3.5.2 Çok Katmanlı Ağlar

Yapay sinir ağı yapay nöronların birleşiminden oluşan ağa denir. Genellikle, sinir ağlarında nöronlar katmanlar halinde gruplanır. Bu yüzden çok katmanlı ağlar diye adlandırılmışlardır. Çok katmanlı ağlarda da her bir nöron çıktı üreterek bir sonraki katmana göndermektedir. Eğer, her nöron ürettiği çıktı değerini, bir sonraki katmandaki tüm nöronlara gönderiyor ise bu gibi sinir ağlarına tam bağlı (fully connected) katman ya da ağ denmektedir. Bu yapıya sahip yapay sinir ağlarında ilk katmanlar orijinal veriyi işlerken diğer katmanlar, kendinden önceki katmanın ürettiği çıktı verilerini işler. Bu durumda her bir nöron kendinden önceki katmanda bulunan nöron sayısı kadar ağırlık parametresine sahip olmaktadır. Çünkü işleme sokacağı girdi sayısı, kendinden önceki katmandaki nöron sayısına eşittir [29]. Şekil 3.7’de çok katmanlı tam bağlı yapay sinir ağı örneği gösterilmiştir.



Şekil 3.7. Çok katmanlı tam bağlı yapay sinir ağı örneği

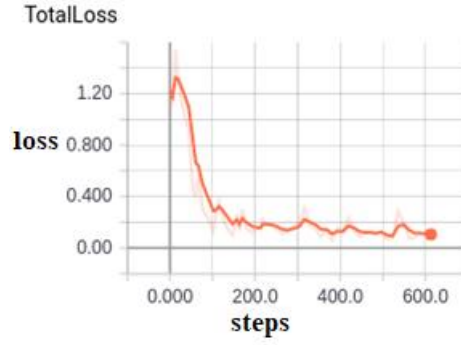
Çok katmanlı bir yapay sinir ağı genellikle üç tip katman içerir. Bunlar: bir girdi katmanı, bir veya daha fazla gizli katman ve bir çıkış katmanı olarak sıralanabilir [29]. Çok katmanlı sinir ağlarının ilk katmanı olan giriş katmanı genellikle verileri değiştirmeden sadece yapay sinir ağına alır alır. Hesaplamanın çoğu gizli katmanlarda gerçekleşir. Gizli katmanlar hesaplamanın yapıldığı ve giriş katmanı ile çıkış katmanı arasında kalan tüm katmanlara verilen isimdir. Çıkış katmanı, gizli katmanlarda gerçekleştirilen işlemlerin sonucunu çıktıya dönüştürür. Bu sonuç, sınıf etiketi gibi sinir ağından beklenen en son veridir. En az bir gizli katmana sahip olan çok katmanlı ileriye besleme ağı evrensel bir hesaplayıcı olarak işlev görebilir, yani neredeyse herhangi bir fonksiyonu hesaplamak için kullanılabilir [41]. Bu tez çalışmasında yoğun olarak resim verileri üzerinde başarı oranı yüksek olan konvolüsyonel sinir ağları ve tam bağlı sinir ağlarından yararlanılacaktır. Konvolüsyonel ağlar, parametre paylaşımını özelliği kullanır ve tam bağlı ağlara kıyasla

sınırlı bağlantılara sahiptir. Bunlar dışında kalan sinir ağı yapıları tez çalışmasının kapsamının dışındadır.

3.3.5.3 Geri Yayılım

Bir yapay sinir ağının öğrenmesi demek, yapay sinir ağında bulunan tüm nöronların ağırlıklarının, çıktıları bilinen girdi değerleri kullanılarak, istenilen çıktıları verebilecek şekilde ayarlanması anlamına gelmektedir. Çok katmanlı bir yapay sinir ağının ağırlıklarının analitik olarak hesaplanması oldukça zordur [28]. Geri yayılım algoritması, ağırlıkları iteratif olarak hesaplamak için basit ve etkili bir çözüm sağlar [42]. Ağırlıkların hesaplanması için çeşitli optimizasyon yöntemlerinden faydalanılır. Klasik geri yayılım algoritmasında, optimizasyon yöntemi olarak meyilli azalım (Gradient descent, GD) kullanılır. GD algoritması oldukça zaman alıcı olabilir ve global minimum hata değerini bulması garanti değildir, ancak doğru yapılandırma ile iyi sonuçlar alınmaktadır. Bu yapılandırma, makine öğrenmesinde hiper parametre düzenlemesi olarak bilinmektedir.

Algoritmanın ilk adımında, bir girdi vektöründeki değerler yapay sinir ağı boyunca, çıkış katmanına kadar nöronlar üzerinde tanımlı olan toplam ve aktivasyon fonksiyonu işlemlerinden geçirilir. Buna ileri yayılım denir. Bu işleme başlamadan önce, yapay sinir ağında bulunan nöronların ağırlıklarına, küçük rastgele değerler ilk değerler olarak atanmış olmalıdır. Yapay sinir ağının alınan çıktısı, ortalama karesel hata gibi bir kayıp fonksiyonu kullanılarak istenen çıktı değeri ile karşılaştırılır. Daha sonra, kayıp fonksiyonunun eğimi (gradyanı) hesaplanır. Elde edilen bu değer hata değeri olarak da adlandırılır. Daha sonra elde edilen hata değerleri, gizli katman nöronlarının hata değerlerini hesaplamak için ağ üzerinden geri iletilir. Gizli nöron kayıp fonksiyonu gradyanları türevlerin zincir kuralı kullanılarak çözülebilir. Son olarak, nöron ağırlıkları ağırlıkların gradyanının hesaplanması ve bir katsayı ile çarpılan gradyanların ağırlıklardan çıkarılmasıyla güncellenir. Gradyanlar ile çarpılan bu değere öğrenme hızı (learning rate) adı verilir. Öğrenme hızı sabit veya değişken olarak ayarlanabilir. Ağırlıklar bu şekilde güncellendikten sonra algoritma farklı girdiler için tekrar aynı işlem adımlarını tekrarlayarak devam eder. Bu eğitim aşaması kayıp fonksiyonunun grafiğinde global minimum noktası bulunduğu veya istenen değerler elde edildiğinde sona erdirilir. Örneğin; Şekil 3.8’ de gösterilen kayıp fonksiyonunda yaklaşık 600’üncü adımda en düşük değer elde edilmiş ve eğitim sonlandırılmıştır.

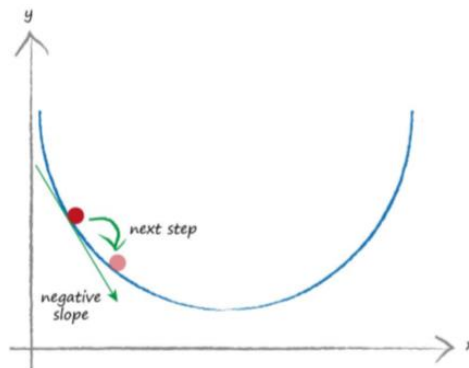


Şekil 3.8. Örnek kayıp fonksiyonu

İlk değerleri atanmış eğitime hazır bir ileri beslemeli ağıın ağırlıklarının güncellenmesini ele alalım. Ağıın eğitiminde kullanılacak eğitim setinin $[(x_1, t_1), (x_2, t_2), \dots, (x_p, t_p)]$ gibi p adet veri olduğu düşünöldüğünde her x_i değerine karşılık ağıın çıktıısı o_i değerinin t_i 'ye eşit olması beklenmektedir. Bu şart sağlanmadığında bir hata değeri hesaplanmaktadır. Bu hata değeri (E) Denklem 3.2'de verilmiştir.

$$E = \frac{1}{2} \sum_{i=1}^p \|o_i - t_i\|^2 \quad (3.2)$$

Algoritmanın amacı elde edilen bu hata değerinin olabilecek en düşük değeri almasını sağlamaktır. Hata değerin Şekil 3.9'da x parametresine bağılı y grafiğı ile ifade edildiğı düşünöldüğünde, y grafiğinin bir noktadaki eğimi negatif olduğunda x parametresinin değeri artırılırsa y değeri minimuma yaklaşmaktadır. Tam tersi eğimin pozitif olduğü durumlarda y değerin minimum olması için x parametresinin değeri düşürölmelidir.



Şekil 3.9. Hata grafiğinin eğimi yardımı ile parametrenin güncellenmesi [43].

Şekil 3.9’da hata değerini etkileyen tek parametre x parametresidir. Ancak, sinir ağlarında öğrenilmesi gereken birden fazla parametre vardır. Bu parametrelerin artırılması ya da azaltılması gerektiği, hata fonksiyonunun güncellenecek parametreye göre türevinin alınması ile anlaşılmaktadır. Bu durumda w ağırlık değerlerinin güncellenmesi $\frac{\partial E}{\partial w}$ türevinin hesaplanmasına bağlıdır. Çünkü hesaplanan, değer w parametresinin değişimi ile E değerinin ne kadar ve hangi yönde değiştiğini göstermektedir. Elde edilen değer ağırlığın ne kadar güncellenmesi gerektiğini göstermektedir. Bu değer öğrenme hızı katsayısı ile çarpılarak ağırlık değerinden çıkarılır. Sonuçta bir sonraki adımda kullanılacak olan ağırlık değeri elde edilir [43]. Parametreler Denklem 3.3’teki gibi güncellenmektedir.

$$yeni\ w = eski\ w - \alpha \frac{\partial E}{\partial w} \quad (3.3)$$

Yukarıda, her girdi değerinden sonra ağırlıkların güncellendiği artırımı öğrenme yöntemi açıklanmıştır [28]. Ancak, bu yöntem kullanıldığında kayıp fonksiyonunun gradyan değeri her yeni gelen girdi için yön değiştirebileceğinden direkt olarak global minimuma yaklaşmamaktadır. Bu da performans ve doğruluk açısından algoritmayı başarısız kılmaktadır. Güncellemeleri hesaplamanın diğer bir yolu tam veri seti için ağırlık güncellemelerini hesapladığımız toplu öğrenme (full batch learning) yöntemidir [28]. Bu yöntemde tüm veri seti için ağırlık güncellemeleri tek seferde hesaplanır. Ancak toplu öğrenme yöntemi yeterince esnek olmadığı ve işlem yükü olarak çok fazla kaynağa ihtiyaç duyduğundan dolayı kullanışlı olmamaktadır [28]. Bu iki yöntemin ortası olarak, her bir güncelleme için eğitim kümesinin yalnızca bir bölümünü kullandığımız mini toplu öğrenme (mini-batch learning) yöntemi daha kullanışlı bir performans sunmaktadır [44].

3.3.5.4 Aktivasyon Fonksiyonları

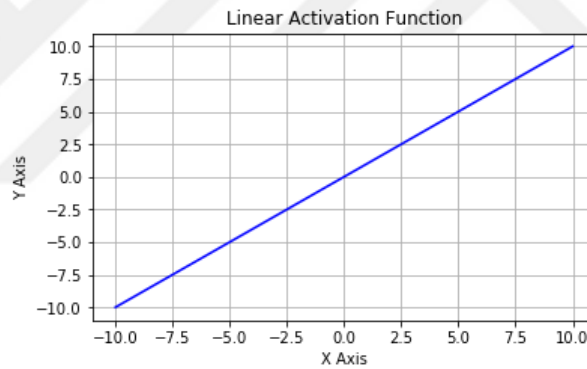
Aktivasyon fonksiyonu her nöronun en son katmanındaki çıktısını belirleyen fonksiyondur. Etkili bir ağ oluşturmak için bu fonksiyonun doğru seçmesi önemlidir.

Biyolojik olarak nöronlar kendilerine gelen sinyaller belli bir eşik değerin üzerinde ise sinyali iletirler, ancak yeteri kadar büyük bir sinyal değilse bu sinyali iletmezler ve biyolojik olarak bir tepki oluşmamış olur. Aktivasyon fonksiyonu da, bir nöron üzerinde işlenen verinin çıkışa aktarılıp aktarılmayacağını belirleyen fonksiyondur. Ayrıca aktivasyon fonksiyonunun bir diğer görevi gönderilecek olan sinyali belli aralıktaki değerlere

sıkıştırmaktır. Bir nöronun çıkışı çok büyük değerler alabilir. Bu çıktı, değişiklik yapılmaksızın bir sonraki tabaka nöronuna beslendiğinde, daha da büyük sayılara dönüştürülebilir ve böylece hesaplama işlemi hesaplanamaz hale getirilebilir. Bundan dolayı nöronların çıkış değerleri aktivasyon fonksiyonu tarafından kontrol edilmektedir. Birçok aktivasyon fonksiyonu yapay sinir ağlarında kullanılmaktadır. Ancak hangi fonksiyonun en doğru yöntem olduğu sorusunun bir cevabı bulunmamaktadır. Her fonksiyonun kendine ait avantaj ve dezavantajları bulunmaktadır.

Lineer Aktivasyon Fonksiyonu

Basit bir doğrusal fonksiyondur. Hesaplanan değer $f(x) = ax$ gibi bir fonksiyondan geçerek çıkış birimlerine gönderilir. Genellikle herhangi bir değişiklik yapılmadan geçiren nöronlarda kullanılır. Lineer aktivasyon fonksiyonları, verilerin lineer olarak ayrılabilir olduğu durumlarda kullanılmaktadır. Fonksiyonun grafiği Şekil 3.10'da verilmiştir.



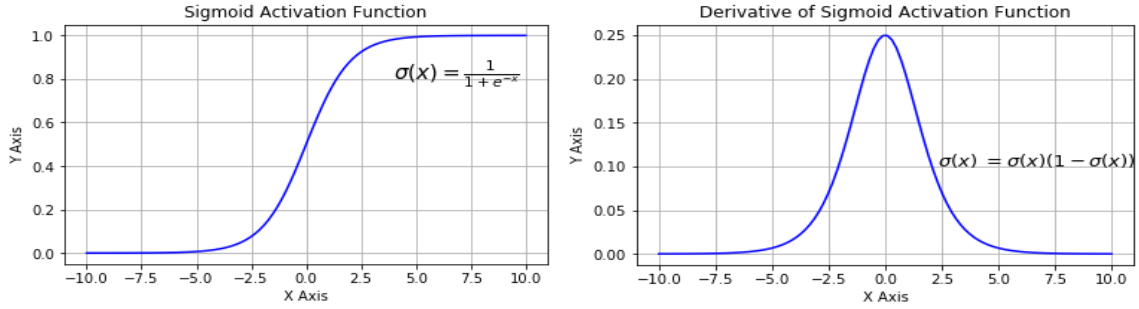
Şekil 3.10. Lineer aktivasyon fonksiyonu

Sigmoid Aktivasyon Fonksiyonu

Sigmoid fonksiyonu logistik aktivasyon fonksiyonu olarak da bilinmektedir. Fonksiyon girdi değeri olarak gerçek sayıları alır ve çıktı olarak 0 ile 1 arasında değerler üretir. 0 ve 1 arasında değerler üretmesi, amacın yüzde olasılık hesaplaması olduğu uygulamalarda en son çıkış katmanında kullanılmasını sağlamaktadır. Büyük negatif sayıları 0'a ve büyük pozitif sayıları 1'e dönüştürür. Matematiksel olarak Denklem 3.4'teki gibi ifade edilir.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (3.4)$$

Fonksiyonun ve türevinin grafiği Şekil 3.11’de verilmiştir.



Şekil 3.11. Sigmoid aktivasyon fonksiyonu ve türev grafiği

Sigmoid aktivasyon fonksiyonu 3 ana dezavantaja sahiptir. Bunlardan ilki kaybolan gradyanlar problemidir. Sigmoid fonksiyonunun 0 ve 1 değerlerine yakın yerlerde düze yakın veya düz olduğu görülmektedir. Bu durumda sigmoid fonksiyonunun gradyanı (türevi-eğimi) 0 ve 1 noktalarında ve yakınında sıfır veya sıfıra çok yakın değerlerdir. Sigmoid aktivasyonu olan ağ üzerinden geri yayılımda, çıktısı 0 veya 1'e yakın olan nöronlardaki gradyanlar neredeyse sıfırdır. Bu tür nöronlara doymuş nöron adı verilir. Güncelleme işlemi gradyanlara bağlı olduğundan dolayı, bu nöronlardaki ağırlıklar güncellenmez. Ayrıca, bu nöronlara bağlı nöronların ağırlığı da yavaş yavaş güncellenir. Büyük yapıli bir yapay sinir ağında doymuş nöronların fazla bulunması geri yayılımın yapılamamasına ve dolayısıyla öğrenmenin gerçekleşmemesine yol açacaktır.

İkinci dezavantaj ise çıktı verilerin sıfır merkezli olmamasıdır. Sigmoid fonksiyonu 0 ve 1 arasında değerler ürettiğinden dolayı çıktı değerleri sıfır etrafında düzenli bir dağılıma sahip değildir. Sıfır merkezli girdiler makine öğrenmesinde ideal durum olarak düşünülmektedir ve eğitim işlemine başlanmadan önce ön işleme ile veriler sıfır merkezli bir uzaya aktarılır. Ancak bu fonksiyon sıfır merkezli çıktılar üretmemektedir.

Son dezavantaj ise, işlem yükünün üstel işlem nedeniyle maliyetli olmasıdır. Bu durum, işlemlerin diğer fonksiyonlara göre daha yavaş gerçekleştirilmesi anlamına gelmektedir.

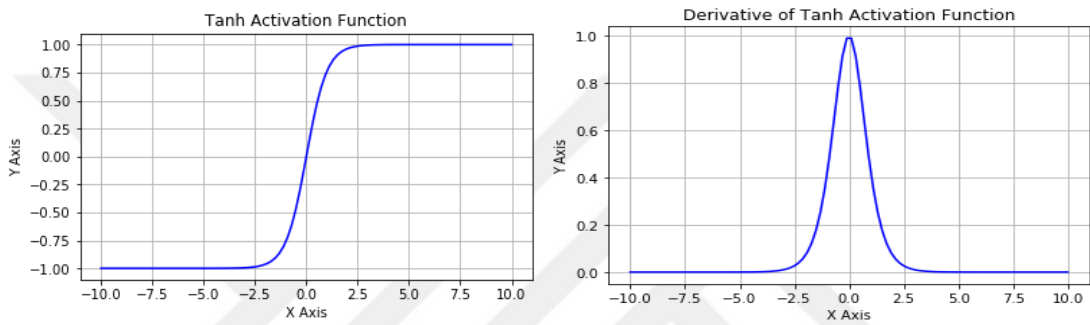
Tanh Aktivasyon Fonksiyonu

Hiperbolik tanjant aktivasyon fonksiyonu olarak da bilinir. Sigmoid'e benzer şekilde, tanh da girdi değeri olarak gerçek sayılar alır, ancak çıktı değerleri sigmoidden farklı olarak -1 ile 1 arasında bir aralıkta üretir. -1 ile 1 arasında çıktı değerleri üretmesi sayesinde sıfır merkezli bir fonksiyon halini almıştır. Bu yüzden uygulamada sigmoid yerine tercih

edilebilen bir fonksiyondur. Tanh fonksiyonu iki sigmoid fonksiyonunun birleşimi olarak düşünülebilir. Tanh fonksiyonunda büyük negatif girdiler -1, büyük pozitif sayılar 1 ve sıfır kendi değeri olarak nöronun çıkışına yönlendirilmektedir. Matematiksel olarak Denklem 3.5'teki gibi ifade edilir.

$$\tanh(x) = \left(\frac{2}{1+e^{-2x}} \right) - 1 \quad (3.5)$$

Fonksiyonun ve türevinin grafiği Şekil 3.12'de verilmiştir.



Şekil 3.12. Tanh fonksiyonu ve türev grafiği

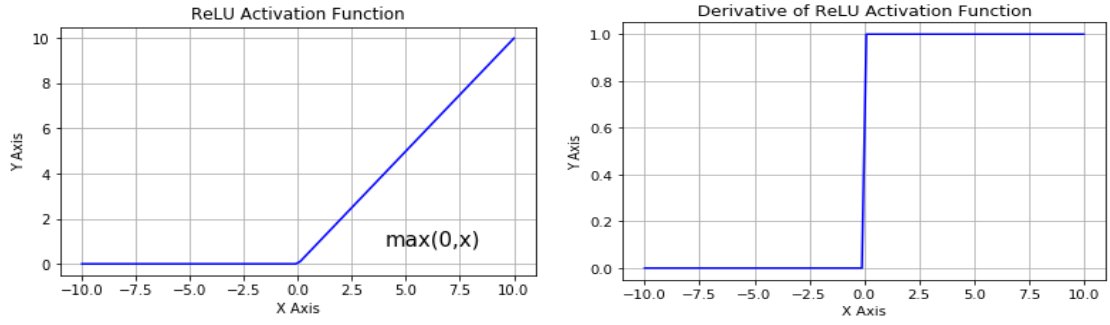
Tanh fonksiyonu, sigmoidin taşıdığı sıfır merkezli olmama dezavantajından kurtulmuş olsa da kaybolan gradyanlar problemi ve exponansiyel işleminin karmaşıklığı bu yöntemde de mevcuttur.

Rectified Linear Unit (RELU) Aktivasyon Fonksiyonu

RELU işlem karmaşıklığı çok az olan hızlı bir aktivasyon fonksiyonudur. Oldukça hızlı sonuçlar üretmesi tercih edilmesini sağlamaktadır. Derin öğrenme uygulamalarının hemen hemen hepsinde tercih edilen aktivasyon fonksiyonu RELU'dur. Matematiksel olarak Denklem 3.6'daki gibi ifade edilir.

$$f(x) = \max(0, x) \quad (3.6)$$

Yani fonksiyona gönderilene değer pozitif ise aynı değeri çıktı olarak verirken, negatif girdileri sıfır olarak değiştirmektedir. İşlem karmaşıklığı sadece bir karşılaştırmadan ibarettir.



Şekil 3.13. RELU fonksiyonu ve türev grafiği

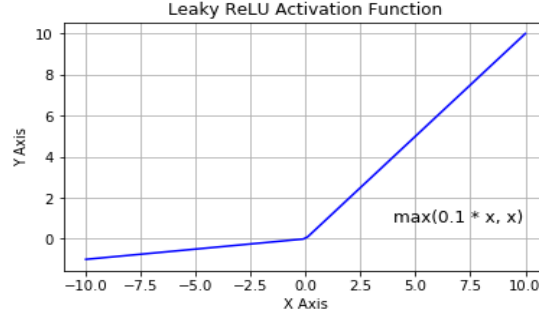
Relu aktivasyon fonksiyonunun grafikleri Şekil 3.13'deki gibidir. İlk bakışta pozitif tarafta lineer bir doğru olduğundan lineer aktivasyon fonksiyonu ile aynı problemleri taşıdığı düşünülebilir ancak Relu non-lineer bir fonksiyondur. Bunun yanında çıktı değerlerinin pozitif tarafında herhangi bir sınırlama yoktur. Relu'nun çıktı aralığı $[0, \infty)$ şeklinde ifade edilir. Buradan Relu'nun nöron aktivasyonunu bir anda yüksek değerlere çıkarabileceği görülmektedir. Ayrıca işlem kolaylığı ile global minimuma tanh gibi karmaşık yapıli yöntemlerden daha hızlı yakınsama yapılmasını sağlamaktadır. Bunun yanında negatif eksenindeki girdiler sıfır olarak döndürüldüğünden ileri yayılım sırasında negatif değer alan nöronlar tamamen pasif kalmakta ve sinir ağı daha sade bir hale dönüşmektedir. Sigmoid ve tanh fonksiyonlarında her nöron bir çıktı ürettiğinden sinir ağı daha karmaşık olarak kalmakta idi. Bunlara ek olarak, Relu fonksiyonunda pozitif ekseninde nöronlar asla doyuma ulaşmazlar ve geri yayılım olumsuz yönde etkilenmez. Negatif yönde ise geri yayılım sırasında türev sıfır olduğundan dolayı ağırlıklar güncellenmez ve nöronlar aktif olmadığından girdi ve hata değerlerine tepki vermezler. Buna ölü nöron problemi denir. Bu problemin önüne geçilmesi için negatif ekseninde sıfır değeri üretmeyen çeşitli RELU varyasyonları önerilmiştir. Bundaki temel amaç türevin sıfır olmaması ve nöronların aktif kalarak eğitim sürecinde yenilenerek öğrenmeye katkı yapmasını sağlamaktır.

Leaky ve Parametrik RELU

Ölü nöron probleminin önüne geçmek için önerilen yöntemlerdir. Klasik RELU'da negatif ekseninde sıfır olan türevi çok küçük sayılarda sıfır olmayan değerlere taşımayı amaçlarlar. Leaky RELU Denklem 3.7'de matematiksel olarak gösterilmiştir.

$$f(x) = \max(0.1x, x) \quad (3.7)$$

Parametrik RELU’da ise çarpan olarak 0.01 sabit sayısı yerine kullanıcı tanımlı olarak uygulanan α değeri kullanılmaktadır. Fonksiyonun grafiği Şekil 3.14’te verilmiştir.



Şekil 3.14. Leaky ve Parametrik RELU grafiği

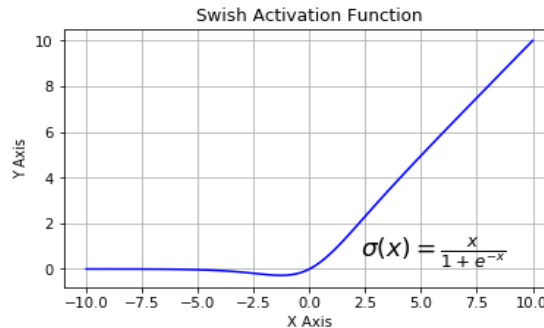
Leaky ve parametrik RELU ile ölü nöron problemi çözülmekle birlikte her zaman daha iyi sonuç vermeleri garanti değildir. Ancak, her biri hesaplama açısından verimli olma, çok daha hızlı yakınsama ve pozitif bölgede doymama gibi avantajlara sahiptir.

Swish

Swish, kendinden kapılı aktivasyon fonksiyonu olarak da bilinir ve yakın bir zamanda Google araştırmacıları tarafından önerilmiştir [45]. Matematiksel olarak Denklem 3.8’deki gibi ifade edilir.

$$\sigma(x) = \frac{x}{1+e^{-x}} \quad (3.8)$$

Bu yöntemin RELU’dan daha iyi sonuç verdiği gösterilmiştir [45]. Şekil 3.15’te bu formüle ait grafik verilmiştir.



Şekil 3.15. Swish aktivasyon fonksiyonu grafiği

Yukarıdaki şekilden, x-ekseninin negatif bölgesinde kuyruk şeklinin ReLU aktivasyon fonksiyonundan farklı olduğunu gözlemlenebilir. Bu fark, Swish aktivasyon fonksiyonunun

ıktısının, giriş değeri arttığı zaman bile düşebilmesini sağlamaktadır. Çoğu aktivasyon fonksiyonu monotondur, yani giriş arttıkça çıkış değeri asla düşmez. Swish ise monoton olmayan bir yapıya sahiptir. Bu nedenle daha dinamik ve değişken bir yapıya sahiptir [45].



4. DERİN ÖĞRENME

Derin öğrenme, geçmişte çok eski zamanlara dayanan yapay sinir ağları kullanarak oluşturulan bir makine öğrenmesi yöntemidir. Çok katmanlı yapay sinir ağları 1980'lerden beri tanınıyor olsa da, birden fazla gizli katmana sahip yapay sinir ağlarını düzgün bir şekilde eğitilmesi algoritmaların eksiklikleri, işlem gücünün yetmemesi gibi nedenlerden dolayı mümkün olmamıştır. Günümüzde bu tip yapay sinir ağlarının eğitilebilir hale gelmesiyle birlikte, daha çok katmanlı yapılara sahip olan modern yapay sinir ağları genellikle derin sinir ağları (DNN) olarak tanımlanmaktadır [46]. Derin sinir ağları kullanılarak makine öğrenmesinin gerçekleştirilmesine derin öğrenme adı verilmektedir. Son on yılda derin sinir ağlarının kullanımında, daha güçlü bilgisayarların ve daha geniş veri kümelerinin ortaya çıkması nedeniyle bir artış yaşanmıştır. 2000'lerin başında, sinir ağlarının grafik işlemci birimleri kullanılarak verimli bir şekilde eğitilebileceği keşfedilmiştir. Grafik işlemciler yapay sinir ağlarının eğitimi için CPU'lardan çok daha kullanışlıdır [47]. Günümüzde, araştırmacılar tipik olarak NVIDIA Tesla K40 gibi ileri teknoloji grafik kartlarını büyük çaplı yapay sinir ağlarını, büyük veri setleri üzerinde eğitmek için kullanmaktadırlar [48]. Teknolojik gelişmelerin yanında yapay sinir ağlarının eğitiminde hız kazanılmasını sağlayan, daha önceden yaygın olarak kullanılan sigmoid ve tanh aktivasyon fonksiyonlarının yerine RELU ve geliştirilmiş versiyonları, swish gibi daha hızlı ve verimli sonuç veren aktivasyon fonksiyonlarının kullanılmaya başlanması; hata fonksiyonlarının geliştirilmesi gibi gelişmeler de büyük çaplı yapay sinir ağlarının geliştirilmesini kolaylaştırmıştır. Bu gelişmelerin tamamı, derin öğrenmenin gerçekleştirilebilmesini ve yaygınlaşmasını sağlamıştır. Derin öğrenme terimi, birden fazla katmana sahip olan, her katmanda daha spesifik özellikler öğrenilerek çok fazla özelliği girdi setinden çıkarma yeteneğine sahip karmaşık yapay sinir ağları kullanılarak makine öğrenmesi işleminin gerçekleştirilmesine için kullanılmaktadır. Derin öğrenme sayesinde, insanlar tarafından belirlenen makine öğrenmesi çözümlerine olan ihtiyaç azalmıştır [28]. Örneğin, bir resim tanıma işlemi sırasında klasik yöntemler kullanılması durumunda, makine öğrenmesine geçilmeden önce manuel olarak ayarlanan bir özellik saptama safhası gerçekleştirilmesi gerekir. Ancak, aynı işlemin derin öğrenme ile gerçekleştirilmesi için sadece bir yapay sinir ağı yeterli olmaktadır. Sinir ağına alt katmanları, veri setinin temel özelliklerini tanımayı

öğrenir ve daha sonra ağın üst katmanlarına bu verileri göndererek daha karmaşık özelliklerin çıkarılması sağlanır. Bu derin öğrenmenin en güçlü noktasıdır denilebilir [49].

Günümüzde derin sinir ağları (DNN), görüntü sınıflandırma [2], görüntüler üzerinde bölütleme [3] ve görüntü sıkıştırma [50] gibi pek çok alanda etkileyici sonuçlar vermiştir. Hatta bu yapay sinir ağlarının performansı, insan algılarının üzerine çıkmaya başlamıştır. Büyük ölçekli ImageNet veri setinde insan ilk-5 sınıflandırma hata oranı % 5.1 olarak bildirilmiştir [51], oysa son teknoloji bir sinir ağına [2] ilk-5 sınıflandırma hata oranı % 3.57 olmaktadır. Bu veriler ışığında derin öğrenmenin resim değerlendirme gereken uygulamalarda kullanılabileceği anlaşılmaktadır. Bu tez çalışmasında bir dijital delilden, adli delil inceleme yazılımları kullanılarak elde edilen görüntü dosyalarının, derin öğrenme kullanılarak incelenemediğinin gösterilmesi ve özel bir kullanım amacı ile bir dijital delilin incelenme sonucunun gözlemlenmesi amaçlanmıştır. Derin öğrenme yöntemleri ve tez çalışmasında kullanılan yöntemler bu bölümde açıklanmıştır.

4.1 Bilgisayar Görmesi

Bilgisayar görmesi, dijital görüntülerin veya videonun içeriklerinden anlamlı bilginin çıkarılması ile ilgilidir. Bilgisayar görmesi, piksel seviyesinde görsel bilginin manipüle edilmesini gerektiren basit görüntü işleme yöntemlerinden farklıdır. Bilgisayar görmesi uygulamaları arasında görüntü sınıflandırması, görsel algılama, 2D görüntülerden 3D sahne oluşturulması, görüntü çıkarımı, artırılmış gerçeklik, makine görmesi ve trafik otomasyonu gibi uygulamalar bulunmaktadır [52].

Günümüzde, makine öğrenmesi birçok bilgisayar görmesi uygulamasının gerçekleştirilebilmesi için gerekli bir bileşendir [53]. Bu uygulamalarda bilgisayar görmesi ve makine öğrenmesi birbirini tamamlamakta ve kullanışlı çıktılar vermektedir. Ancak, başarılı sonuçların alınabilmesi için görsel imgelerde bulunan fazla miktarda bilgiyle baş edebilecek algoritmalar gerekmektedir. Ayrıca, bazı uygulamalar için bu büyük boyutlu verinin gerçek zamanlı olarak işlenmesi gerekmektedir [54]. Bilgisayar görmesi uygulamalarının baş etmesi gereken bu problemler için günümüzde derin öğrenme tabanlı birçok algoritma geliştirilmektedir [4,55-61]. Bu tez çalışmasında bilgisayar görmesi uygulamaları içinden nesne tespiti uygulaması kullanılmıştır.

4.1.1 Nesne Tespiti

Nesne tespiti, bilgisayar görmesinin klasik problemlerinden biridir ve sıklıkla zor bir işlem olarak tanımlanmaktadır. Deformasyon, ışık ve bakış açısı gibi parametrelerin değişmesine karşın sabit bir çözüm oluşturmayı amaçlaması gibi birçok açıdan, diğer bilgisayar görmesi işlerine benzemektedir. Nesne tespitini farklı bir problem haline getiren şey ise hem bir görüntünün içindeki nesnenin sınıflandırılmasını hem de resim üzerinde hangi bölgede olduğunu tespitini (lokalizasyon) gerektirmesidir [48]. Yani nesne tespiti uygulamaları sınıflandırma ve lokalizasyon uygulamalarını kapsamakla birlikte birden fazla nesne için de çalışan karmaşık bir bilgisayar görmesi uygulamasıdır. Şekil 4.1’de sınıflandırma, lokalizasyon ve tespit uygulamalarının çıktıları gösterilmiştir.



Şekil 4.1. Sınıflandırma, lokalizasyon ve nesne tespiti

Bir nesneyi algılamak için, nesnenin nerede olabileceği ve görüntünün nasıl bölümlendirileceği konusunda fikir sahibi olunması gerekmektedir. Bu bilgiler bazı uygulamalarda rastgele tahmin edilirken bazen de resim bölütlenerek nesnelerin bulunabilecekleri alanlar buna dayanarak tahmin edilmeye çalışılmaktadır. Bu tahminler resimde aranan nesnenin boyut ve yerini belirtmektedir. Genellikle tahminler bir dörtgenin köşe koordinatları şeklinde üretilmektedir. Şekil 4.1’de gösterilen kırmızı, yeşil ve mavi dikdörtgenler buna örnek olarak gösterilebilir. Oluşturulan bu tahminlere sınırlayıcı kutular (bounding box) denmektedir. Sınırlayıcı kutular, rastgele seçilen bir çokgenin eğitim sırasında problemlere yol açabilmesi ve öğrenme sırasında gerçekleşen adımların dörtgen filtreler ile gerçekleştirildiğinden dörtgen bir şeklin daha kolay yönetilmesi nedenlerinden dolayı genellikle dörtgen şeklinde kullanılmaktadır. Tahmini sınırlandırıcı kutular oluşturulduktan sonra resim üzerinde sınırlandırıcı kutu tarafından çevrelenen alanlar bir

sınıflandırma algoritması ile sınıflandırılır [62]. Aranan nesne olduğu sınıflandırma algoritması tarafından doğrulanan kutular daha sonra nesnenin sınırlarına göre optimize edilebilir [52].

Derin öğrenme yöntemleri günümüzdeki gibi popülerlik kazanmadan önce nesne tespiti uygulamaları için ölçekten bağımsız özellik dönüşümü (SIFT), yönlendirilmiş gradyanların histogramı (HOG) gibi çeşitli özellik çıkarımı algoritmaları yaygın olarak kullanılmıştır [63,64]. Ancak 2010'lu yıllarda nesne tespiti için konvolüsyonel yapay sinir ağlarının kullanılması yaygınlaşmaya başlamıştır ve günümüzde en başarılı yöntemler konvolüsyonel yapay sinir ağları kullanılarak oluşturulmuştur [48,55,56,62].

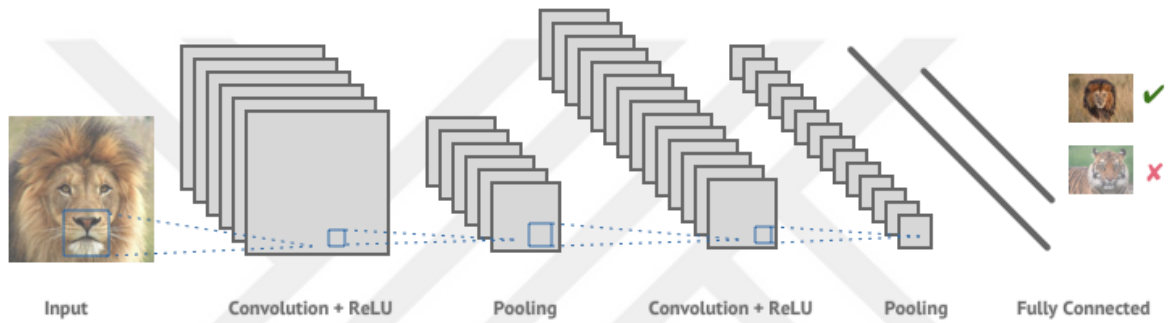
CNN'lerin geniş çapta benimsenmesinden önce sınırlayıcı kutular üretmek için sunulan yaygın iki yöntem kullanılmıştır. İlk yöntemde, nesnelerin yerlerinin tespiti için alan olasılıkları içeren büyük çaplı tahmin kümeleri rastgele üretilmiştir. Genellikle bu yöntemde kayan pencere yöntemleri kullanılmıştır. Ancak bu yöntemlerin çoğu hız ve maliyet bakımından zayıf oldukları için benimsenmemiştir [55]. İkinci yöntemde ise nesnelerin yerleri resim özelliklerine göre tahmin edilmiştir. Bu yöntemde daha az ve daha fazla oranda doğru bölge içeren tahminler üretilmiştir [65]. Konvolüsyonel sinir ağlarının yaygınlaşması ve bu alan tahmin yöntemi ile birleştirilmesi ile nesne tespitinde başarılı sonuçlar elde edilmiştir. Günümüzde standart haline gelen nesne tespiti uygulamaları konvolüsyonel sinir ağları ve selective search [65] gibi resmin özelliklerine göre özellik çıkaran algoritmaları bir arada kullanmaktadır.

4.2 Konvolüsyonel Sinir Ağları (CNN)

Son yıllarda yapay sinir ağları, çeşitli alanlarda alınan başarılı sonuçlar sayesinde yaygın olarak kullanılmaya başlanmıştır. Daha spesifik olarak, derin yapay sinir ağları birçok örüntü tanıma ve bilgisayar görmesi uygulamalarında klasik yöntemleri geride bırakarak en başarılı sonuçları ortaya koymuştur. Bu sistemlerin başarısı, manuel olarak belirlenen özelliklere ihtiyaç duymadan güçlü nesne özellikleri oluşturma kabiliyetlerinden kaynaklanmaktadır. Derin sinir ağları hem düşük seviye görüntü özelliklerini hem de üst düzey soyut kavramları ve yeterli eğitim verisi sağlanarak verilen soyut giriş-çıkış ilişkilerini öğrenebilir. Konvolüsyonel sinir ağları derin sinir ağlarının bir alt ailesidir ve günümüzde yaygın olarak bilgisayar görmesi problemlerinde kullanılmaktadır. CNN'ler 1970'lerin

sonunda ortaya çıkmış ve 1980'lerin sonu ve 1990'larda yaygın olarak sınıflandırma ve tespit uygulamalarında kullanılmıştır [66,67].

Konvolüsyonel sinir ağları genellikle sonunda aktivasyon fonksiyonu bulunan konvolüsyon, havuzlama, tam bağlı sinir katmanları ve bir karar verici fonksiyona sahip çıkış katmanlarından oluşmaktadır. Aktivasyon fonksiyonu olarak genellikle RELU kullanılmakla beraber, kesin bir kural bulunmamaktadır. Ancak, yapay sinir ağının son katmanında bulunan aktivasyon fonksiyonu problemin tipine göre değişkenlik göstermektedir. İki sınıflı sınıflandırma için sigmoid, çok sınıflı sınıflandırma için softmax, regresyon problemleri için ise lineer aktivasyon fonksiyonu kullanılmaktadır.



Şekil 4.2. Konvolüsyonel sinir ağlarının genel yapısı

Şekil 4.2.'de görüldüğü gibi konvolüsyonel sinir ağlarında girdi olarak verilen resim konvolüsyon ve aktivasyon işlemlerinden geçmekte, daha sonra en önemli özelliklerin çıkarılması için havuzlama işlemine sokulmakta ve bu işlemler tekrarlanmaktadır. En son olarak tam bağlı katmanlar ve karar verici aktivasyon fonksiyonu ile birlikte algoritma sonuç değerini döndürmektedir. Konvolüsyonel sinir ağlarında konvolüsyon ve pooling adımları özellik çıkarmak için kullanılırken tam bağlı katmanlar çıkarılan özelliklere göre sınıflandırma yapmak için kullanılmaktadır.

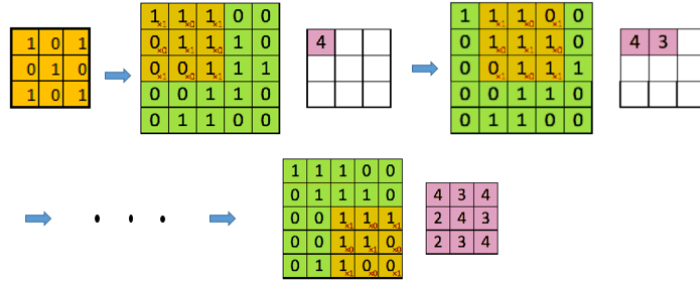
4.2.1 Konvolüsyonel Sinir Ağlarının Yapısı

Konvolüsyonel sinir ağları, klasik sinir ağlarından farklı olarak yapısında, giriş katmanından gelen veriden özellik çıkarmak amacıyla kullanılan, konvolüsyon, havuzlama, seyreltme (dropout), gibi katmanlar bulundurmaktadır. Bu katmanların sonunda klasik sinir ağlarında bulunan tam bağlı katmanlara sahiptir. Bu bölümde bu katmanlar ve çalışma prensipleri anlatılmıştır.

4.2.1.1 Konvolüsyon Katmanı

CNN yapısının oluşturulmasında bir biyoloji terimi olan reseptif alanlardan esinlenilmiştir [68]. Reseptif alanlar hayvan görsel korteksin bir özelliğidir. Kenarlar gibi belirli uyaranlara duyarlı detektörler gibi davranırlar. Ve bu detektörler tüm görme alanında bulunurlar. Bu biyolojik fonksiyon, konvolüsyon işlemi kullanılarak bilgisayarlarda uygulanabilir. Görüntü işlemede çeşitli uyaranlar filtreler yardımıyla konvolüsyon kullanılarak elde edilebilir [69]. Elde edilen uyaranlara özellik adı verilir. Kullanılan filtre ise konvolüsyon filtresi olarak adlandırılabilir. Yani bilgisayar üzerinde konvolüsyon işlemi resim üzerinde kaydırılan bir filtre ile gerçekleşmektedir. Filtrenin elemanları her adımda üzerine gelen resim piksel değerleri ile çarpılmakta ve tüm filtreden gelen çarpın değerleri toplanarak sonuç matrisine kaydedilmektedir. Bu işlem tüm resim kaydırma miktarına göre gezilerek tamamlanmaktadır. Konvolüsyon adımının sonunda bir sonuç matrisi elde edilmektedir bu matrise aktivasyon matrisi veya özellik matrisi adı verilmektedir. Bir konvolüsyon katmanında birden fazla konvolüsyon filtresi bulunabilir [68]. Filtreler içinde sayılar bulunduran matrislerdir ve girdiden özellik çıkarmak için kullanılırlar. Bu durumda konvolüsyon filtreleri klasik sinir ağlarındaki nöron katsayılarına benzetilebilir.

Görüntünün tüm bölümleri için aynı filtreler kullanıldığından, serbest parametrelerin sayısı tam bağlı bir sinir ağı katmanı ile karşılaştırıldığında önemli ölçüde azaltılır. Çünkü tam bağlı sinir ağı katmanında her bir nöron için farklı parametreler kullanılır [66]. Konvolüsyon katmanındaki nöronlar genellikle aynı parametreleri paylaşırlar ve her adımda resmin lokal bir alanı üzerinde işlem yaparlar. Konvolüsyonun parametre paylaşımı özelliği resimde aranan bir nesnenin konumdan bağımsız olarak tanınabilmesini sağlar. Konvolüsyon işlemlerinin arka arkaya koyulması nedeniyle bu tür sinir ağlarına konvolüsyonel sinir ağları adı verilmektedir. Ancak bu katmanların arasında genellikle farklı katmanlar eklenmekte ve önemli özellikler üzerinde işlemlerin devam etmesi sağlanmaktadır. Ayrıca geri yayılım algoritması yardımı ile konvolüsyon filtresi güncellenmektedir. Teorik olarak, girdiye daha yakın katmanlar, görüntünün köşe ve kenar noktaları gibi düşük seviyeli özelliklerini öğrenmekte, çıktıya daha yakın katmanlar ise, bu özellikleri birleştirerek daha anlamlı şekilleri tanımak için sonuçlar üretmektedir. Bu tezde de konvolüsyonel ağların nesneleri tanıyabilmesi özelliğinden yararlanmak amaçlanmıştır.



Şekil 4.3. Konvolüsyon işlemi

Şekil 4.3’de görüldüğü gibi konvolüsyon adımında her bir adımda nokta çarpımlar hesaplanarak sonuç matrisi elde edilir. Derin öğrenmedeki mantık çarpım yapılacak olan bu filtrelerin değerlerinin makine öğrenmesi ile güncellenmesine dayanmaktadır. Bunun nedeni her öğrenme uygulaması için manuel olarak tanımlanan bir filtrenin çıkarılmasının zor olması ve hatta bazı durumlarda imkânsız olmasıdır. Bu nedenle konvolüsyon işleminde, filtrede kullanılan değerlerin güncellenmesi derin öğrenme yöntemlerinin başarısını artırmaktadır. Ancak, konvolüsyon işleminde dikkat edilmesi gereken noktalardan birisi her konvolüsyon işleminden sonra çıktı verisinin boyutunun, girdi verisinden daha küçük olacağıdır. Bu durumda birkaç kez üst üste konvolüsyon işlemi sonrasında elde edilen özellik kümesi çok küçülecek ve özellik çıkarımının gerçekleşmemesi problemi ortaya çıkacaktır. Bu nedenle konvolüsyon işleminden önce, girdi ile çıktı matrisinin boyutlarının istenilen değerde olması için bir doldurma (padding) işlemi gerçekleştirilmelidir. Padding bir resmin kenarlarının çerçeve şeklinde yeni bitler ile doldurulması demektir. Filtre boyutu ve kaydırma miktarına bağlı olarak konvolüsyon işleminin sonucunda arzu edilen boyutta çıktının alınması için farklı boyutlarda uygulanabilmektedir. Örneğin; Şekil 4.3’de 5x5 boyutunda bir görüntü üzerinde, filtre boyutu 3x3 ve kaydırma miktarı 1 olacak şekilde konvolüsyon işlemi gerçekleştirilmiştir. Sonuç olarak 3x3 boyutunda bir çıktı elde edilmiştir. Eğer kaydırma miktarı 2 olsa idi 2x2 boyutunda bir çıktı elde edilmiş olurdu. Özellik matrisinin boyutu, resim boyutu $n \times n$, kaydırma miktarı s , filtre boyutu $k \times k$, padding boyutu p olarak kabul edildiğinde;

$$\left[\frac{n+2p-k}{s} + 1 \right] \times \left[\frac{n+2p-k}{s} + 1 \right] \quad (4.1)$$

Denklem 4.1 ile hesaplanmaktadır. Makine öğrenmesinde genellikle anlamlı verilerin sadece giriş katmanında verilen girdiye ait olması istendiğinden padding işlemi yapılırken sıfır değeri kullanılmaktadır.

5	6	7	8						
0	6	7	8						
5	6	15	8						
5	6	7	8						

Padding →

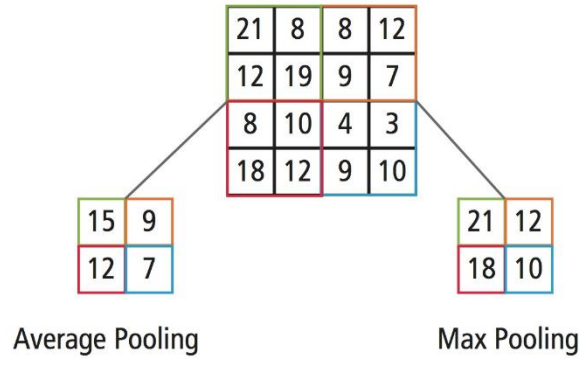
0	0	0	0	0	0
0	5	6	7	8	0
0	0	6	7	8	0
0	5	6	15	8	0
0	5	6	7	8	0
0	0	0	0	0	0

Şekil 4.4. $p = 1$ için padding sonucu

$s=1$ ve $k=2$ için Şekil 4.4'deki matrisini padding işlemi yapılmadan önce konvolüsyon işlemine sokulması sonucu 3×3 boyutunda bir özellik matrisi elde edilirken, aynı matrisin $p=1$ ile padding işlemine tabi tutulması sonucunda konvolüsyon işlemi gerçekleştirildiğinde formülden de çıkarılacağı üzere özellik matrisi 4×4 boyutunda bir özellik matrisi elde edilmektedir. Yani giriş katmanındaki veri ile eşit boyutta çıkış elde edilmiş olmaktadır.

4.2.1.2 Havuzlama (Pooling) Katmanı

Havuzlama katmanı, özellik haritasının sinir ağı üzerinde ilerlerken boyutunun küçültüldüğü ve herhangi bir öğrenme işleminin gerçekleşmediği katmanlardır. Genellikle, sinir ağının çıkış katmanına daha yakın olan katmanlarında, aranan özelliğin resmin hangi bölgesinde olduğu bilgisi, gereksinim duyulmayan bir bilgidir. Sadece önemli olan özelliklerin işlenmesi öğrenme açısından daha iyi sonuçlar doğurmaktadır. Ayrıca sinir ağının daha kolay yönetilebilmesi ve öğrenme aşamasının hızlandırılması için çok önemli olmayan verilerin temizlenmesi gerekmektedir. Bu iki işlem için de özellik vektörünün küçültülmesi gerekmektedir. Bunun için konvolüsyonel sinir ağlarında pooling katmanından yararlanılmaktadır. Konvolüsyon katmanına benzer olarak pooling katmanında da $k \times k$ boyutunda bir filtre ve bir s kaydırma miktarı bulunmaktadır. En yaygın olarak iki tür pooling katmanı kullanılmaktadır. Bunlar; maksimum havuzlama (max-pooling) ve ortalama havuzlama (average-pooling) yöntemleridir. Max-pooling yönteminde her adımda filtrenin kapsadığı alanda bulunan maksimum değer çıktı olarak verilirken, average-pooling yönteminde kapsanan alandaki değerlerin ortalaması çıktı olarak verilmektedir. Aynı özellik matrisi için iki farklı yöntemin çıktıları Şekil 4.5'te gösterilmiştir.



Şekil 4.5. Maksimum ve ortalama havuzlama

Özellik matrisini boyutunun ne kadar küçüleceğini filtre boyutunun yanında kaydırma miktarı da etkilemektedir. Genellikle kaydırma miktarı filtrenin boyutu ile aynı seçilmektedir. Pooling katmanı önemli özelliklerin konumdan bağımsız olarak öğrenilmesini ve sinir ağıının daha hızlı çalışmasını sağlamaktadır.

4.2.1.3 Aktivasyon ve Tam Bağlı Sinir Ağı Katmanları

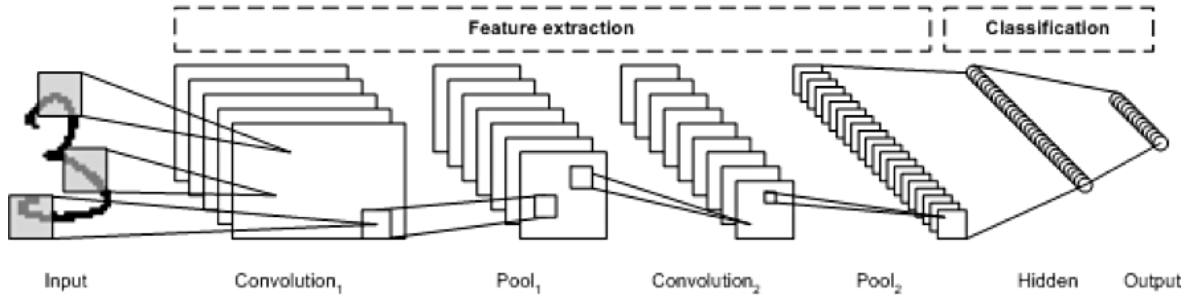
Konvolüsyonel sinir ağlarında, konvolüsyon adımından sonra genellikle elde edilen çıkış değerleri aktivasyon fonksiyonundan geçirilir. Ancak bazı çalışmalarda bu adım farklı bir katman olarak kabul edilmektedir. Bu katmanda bölüm 3.3.5.4’de anlatılan aktivasyon fonksiyonları kullanılmaktadır.

Konvolüsyonel sinir ağlarının son katmanları genellikle tam bağlı sinir ağı katmanlarıdır. Ancak böyle bir şart yoktur. Tam bağlı katmanın kullanılmasının amacı konvolüsyon katmanında saptanamayan önemli özelliklerin tam bağlı katman tarafından öğrenilebilme olasılığıdır. Tam bağlı katmanların verimli ve doğru olarak kullanılabilmesi için küçük giriş verileri kullanılmalıdır. Giriş verilerin boyutunun azaltılması için de bir önceki katmanda havuzlama işlemi gerçekleştirilmektedir.

4.2.2 Konvolüsyonel Sinir Ağlarının Gelişimi

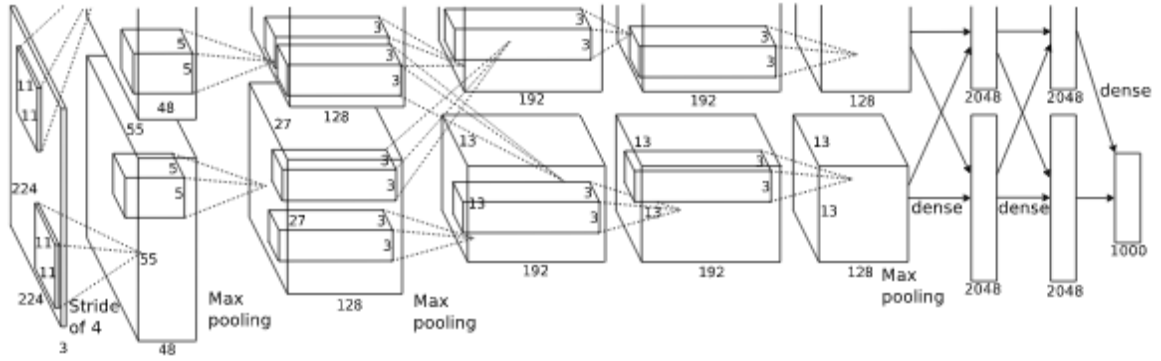
Bu bölümde, CNN gelişiminde kilometre taşı olarak kabul edilen ve daha sonra geliştirilen birçok sinir ağına ilham vermiş olan CNN mimarilerinden bahsedilmiştir. Bu CNN’lerin performansı her yıl düzenlenen ImageNet Büyük Ölçekli Görüntü Tanıma Yarışması (ILSVRC) [51] temel alınarak değerlendirilmiştir.

LeNet [66] CNN'in ilk başarılı uygulaması olarak bilinmektedir. Bu yapay sinir ağı el yazması sayıların tanınması amacıyla kullanılmıştır. Günümüzdeki CNN'lere oranla yavaş olsa da ilk başarılı sonuçların alınması ile kendini takip edecek olan çalışmalara yol gösterici olmuştur. 2 konvolüsyon, 2 pooling ve bir tam bağlı sinir katmanından oluşmaktadır. MNIST [70] test veri seti üzerinde %1.7 hata oranına sahiptir. *LeNet*'in yapısı Şekil 4.6'da gösterilmiştir.



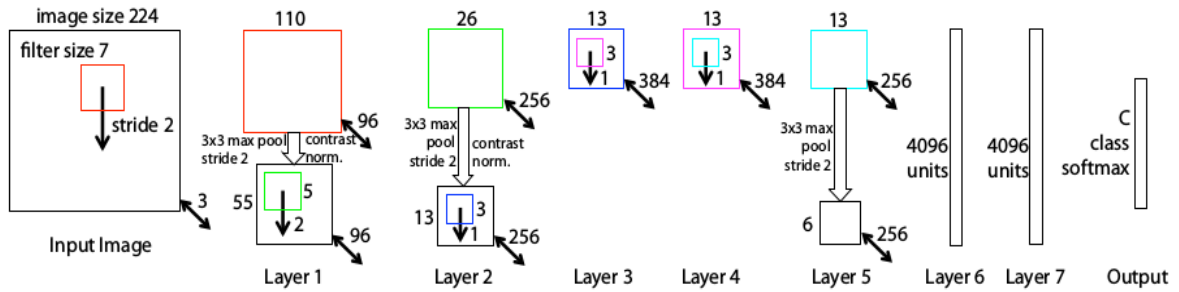
Şekil 4.6. *LeNet* yapısı [66].

AlexNet [57], A. Krizhevsky ve ark. tarafından geliştirilmiştir ve büyük ölçekli görüntü tanıma üzerine başarıyla uygulanan ilk derin CNN olarak tanınmaktadır. 2012 yılında ILSVRC'de resim sınıflandırma alanında %16,4 hata oranı ile birinci olmuştur. *AlexNet*, büyük oranda *LeNet*'e benzemekle birlikte daha fazla konvolüsyon katmanına sahiptir. Aşırı eğitim ve kaybolan gradyan problemlerine karşı daha iyi çözümler kullanmıştır. Bunlardan ilki aktivasyon fonksiyonu olarak, sigmoid yerine ReLU kullanmasıdır. İkinci olarak aktivasyon fonksiyonuna girmeden önce girdilerin normalize edilerek ReLU katmanından geçirilmesidir. Bu sayede kaybolan gradyan probleminden kaçınılmak amaçlanmıştır. Üçüncü olarak, üst üste binen havuzlama katmanları kullanmıştır. Bu tip havuzlamada kaydırma miktarı filtre boyutuna eşit değildir. Kaydırma miktarı filtre boyutundan daha küçüktür, yani işleme alınan bitler bir sonraki adımda tekrar işleme alınmaktadır. [57]'de bu yöntemin aşırı eğitimden kaçınmakta yararlı olduğu açıklanmaktadır. Dördüncü önlem olarak aşırı eğitimden kaçınmak için eğitimin her adımında bazı nöronların işleminden kaldırılmıştır. Bu yöntem ile sürekli aynı nöronların çalışması engellenmiş olmaktadır [2,71]. Genel olarak *AlexNet* 5 konvolüsyon ve 3 tam bağlı katman ile 60 milyon parametreye sahiptir. Bunların en büyük kısmı kullanılan 3 tam bağlı katmandadır. Günümüzde kullanılan CNN yapıları ile karşılaştırıldığında bu sayı, hız açısından verimsiz olabilecek bir değer olarak görülmektedir. *AlexNet*'in yapısı Şekil 4.7'de gösterilmiştir.



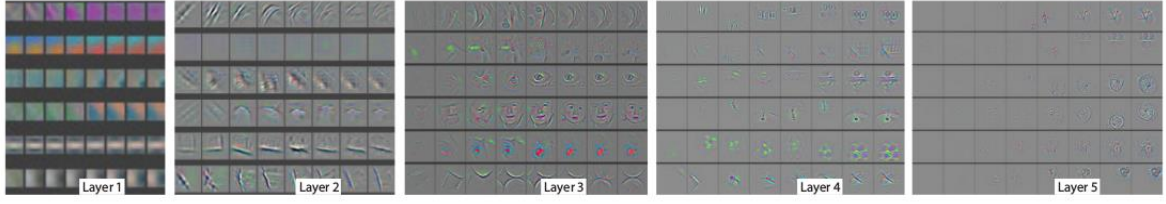
Şekil 4.7. AlexNet yapısı [57].

ZFNet [72], 2013 yılında ILSVRC'yi kazanmıştır ve Zeiler and Fergus tarafından sunulmuştur. Bu ağ da AlexNet'e çok benzemektedir. Ancak, eğitimin başlangıcında seçilmesi gereken parametrelerin düzenlenmesini daha ayrıntılı olarak ele alınmış ve çıkış katmanlarına yakın katmanlarda daha küçük konvolüsyon filtreleri kullanılmıştır. ZFNet AlexNet den daha fazla (75 milyon) parametreye sahiptir. ZFNet'in yapısı Şekil 4.8'de gösterilmiştir.



Şekil 4.8. ZFNet yapısı [72].

Bu yayının ana katkısı ters konvolüsyon ve ters havuzlama işlemleri kullanarak gizli katmanların görselleştirilmesini sağlamasıdır. CNN yapılarında bu tip işlemlerin kullanılması oldukça yeni ve nadirdir. Bu iki katmanın kullanımı ile derin özellik haritalarını görüntüye aktarabilmektedir. Böylelikle gizli katmanlarda vurgulanan özelliklerin görselleştirilmesi sağlanmaktadır. Günümüzde bu yöntem yeni bir araştırma konusu haline gelmiştir [73-76]. ZFNet sinir ağında eğitim boyunca rasgele seçilen özellik haritaları Şekil 4.9'da gösterilmiştir.



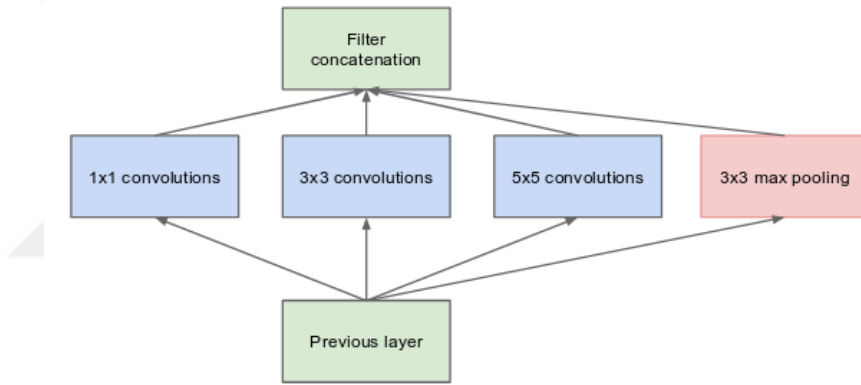
Şekil 4.9. ZFNet sinir ağında eğitim boyunca rasgele seçilen özellik haritalarının görselleştirilmesi [72].

VGGNet [58], sınıflandırmanın yanında, köşe tespiti ve mekân tanıma gibi birçok uygulamada kullanılan popüler bir sinir ağı yapısıdır [77-79]. 2014 yılında gerçekleştirilen ILSVRC’de GoogleNet’in arkasında ikinci en düşük hata oranına sahip olmuştur (%7.3) [58]. Beş adet konvolüsyon kümesi adımına sahiptir. Her birimdeki tüm özellik haritaları aynı boyut ve sayıya sahiptir. Birimler maksimum havuzlama katmanlarıyla birbirlerine bağlanır. VGGNet’in farklı katmanlı sürümleri mevcuttur. En iyi sonuçların elde edildiği 16 katmanlı olan versiyonunda 140 milyon parametre bulunmaktadır. VGG, temiz ve düzenli mimarisi ve eğitilmiş modellerinin bulunması sayesinde dâhili katmanlarda değişiklik yapabilecek araştırmalara olanak sağlamaktadır. Dolayısıyla, semantik bölütleme ve kenar algılama gibi üst düzey bilgisayar görme görevleri için yaygın kullanılan mimaridir. VGGNet’in farklı konfigürasyonları Şekil 4.10’da gösterilmiştir.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

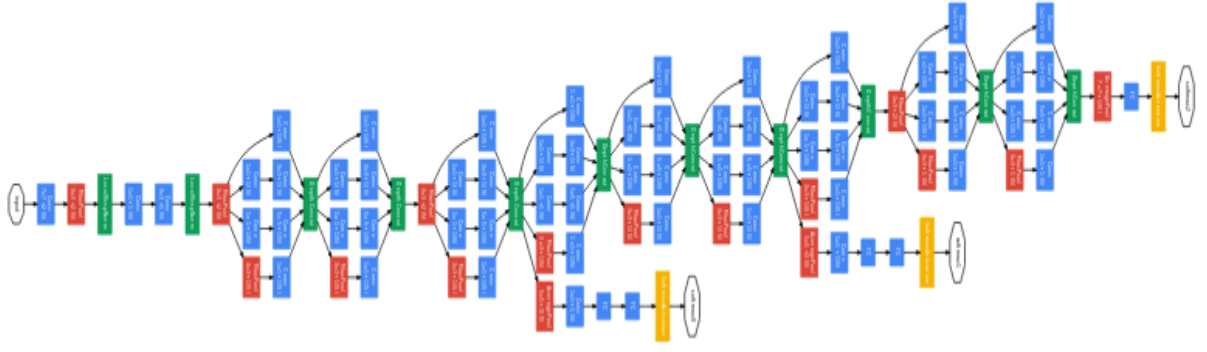
Şekil 4.10. VGGNet yapısı [58].

GoogLeNet [59], Szegedy ve ark. tarafından 2014 yılında tasarlanmış ve 2015 yılında sunulmuştur. 2014 yılında VGGNet'i %6.7'lik hata oranı ile geride bırakarak ILSVRC'de birinci olmuştur. İlk derin sinir ağı olarak ILSVRC'yi kazanan AlexNet'e göre hata oranını %10 azaltmıştır. Yapısı art arda küçük konvolüsyonlara dayanması sebebiyle daha derin bir yapıya sahip olmasına rağmen 12 kat daha az parametreye (4 milyon) sahiptir. Bu ağın asıl katkısı, her katmandaki özelliklerden daha iyi yararlanmayı sağlamak amacıyla inception modüllerini kullanmasıdır. Basit bir inception modülü, bir katmana gelen girdiyi alır ve paralel olarak birden çok farklı ve bağımsız katmanlara gönderir. Örneğin aynı girdi hem 3x3 konvolüsyon, hem 5x5 konvolüsyon, hem de maksimum havuzlama işlemlerine sokulur. Daha sonra elde edilen özellik haritaları birleştirilerek toplam çıktı elde edilir. Girdinin paralel olarak geçtiği bu işlemlerin bulunduğu katmana inception modülü adı verilir. Şekil 4.11'de örnek inception modülü görülmektedir.



Şekil 4.11. Basit inception modül yapısı [59].

GoogLeNet, son konvolüsyon katmanının özellik haritalarını bir vektöre dönüştürmek için ortalama havuzlama kullanır ve tahmini gerçekleştirmek için yalnızca bir tam bağlı sinir katmanı kullanılır. Ortalama havuzlama katmanı, konvolüsyon katmandan gelen parametrelerin çoğundan tasarruf etmesi nedeniyle, son konvolüsyon katmanından sonra tam bağlı sinir katmanına iyi bir alternatif olarak kabul edilmiştir.

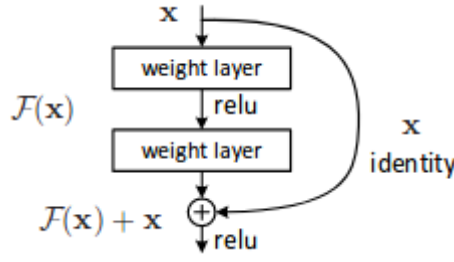


Şekil 4.12. GoogleNet yapısı [59].

Şekil 4.12’de GoogleNet’in yapısı gösterilmiştir. Dikkatli olarak incelendiğinde GoogleNet’in inception modüllerin birleşiminden meydana geldiği anlaşılmaktadır. Resimde yeşil ile gösterilen adımlar parametre birleşim noktaları, kırmızı ile gösterilen adımlar havuzlama işlemleri, mavi ile gösterilen adımlar konvolüsyon adımları ve sarı ile gösterilen adımlar ise sınıf tahmini yapılan adımları ifade etmektedir.

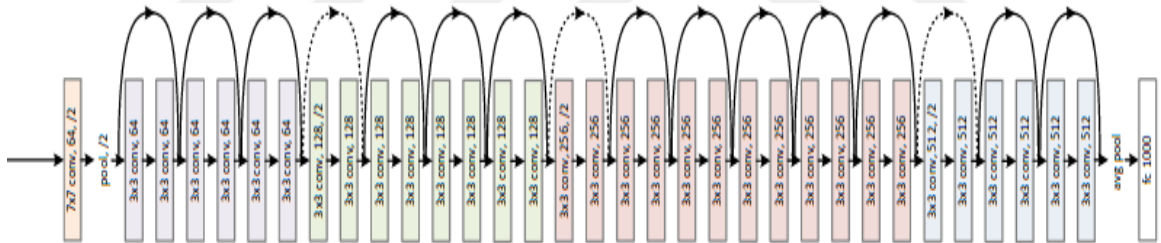
Resnet [2], son yıllarda geliştirilen en başarılı CNN’lerden biridir. Bunun yanında, çok derin CNN’lerin eğitilmesine olanak sağlayacak yeni bir yaklaşım ortaya koymuştur. 1000 katman sınırını aşan ilk CNN’dir. 2015 yılınca ILSVRC yarışmasında sınıflandırma, bölütleme ve tespit gibi çeşitli alanlarda birinci olmuştur. ResNet’in sınıflandırma hata oranı sadece %3.6 olarak test edilmiştir. Bu modelin başarısı ortalama bir insanın %5 civarında hataya sahip olduğu düşünüldüğünde daha net anlaşılmaktadır [80]. ResNet kendinden sonra gelen birçok modele öncülük etmiş ve başta sınıflandırma olmak üzere birçok farklı uygulamada kullanılmıştır [60,61,81].

Genel olarak daha derin sinir ağlarının daha başarılı olması beklenir. Ancak durum her zaman böyle olmamaktadır. Model büyüdükçe eğitim kümesi küçük kalabilmekte ve aşırı sayıda parametre olduğundan parametrelerin optimizasyonu gerçekleştirilememektedir buna bağlı olarak eğitimin gerçekleşmemesi gibi durumlar ortaya çıkarak başarısız sonuçlar üretilebilmektedir. ResNet bu problemin önüne geçmek amacıyla ‘residual blok’ olarak adlandırılan yapıyı kullanmıştır. Bu şekilde derin sinir ağlarında öğrenmenin daha kolay bir şekilde gerçekleştirileceği gösterilmiştir. Şekil 4.13’de resudial blok yapısı verilmiştir.



Şekil 4.13. Residual blok yapısı [2].

Bu yapıda konvolüsyon katmanlarının girişleri kendinden sonraki konvolüsyon katmanının aktivasyon fonksiyonundan önce çıkışına eklenmektedir. Bu sayede geri yayılım esnasında işlem kolaylığı sağlanmakta ve kaybolan gradyan probleminden kurtulmuş olmaktadır. ResNet, VGGNet'e benzer olarak düzenli bir yapıya sahiptir. 3x3 çıkış olarak giriş boyutunda veri üreten konvolüsyon katmanlarına ve bu katmanlar arasında pooling katmanlarına sahiptir. ResNet'in, CNN'lerin gelişimine ne büyük katkısı, çok daha derin sinir ağlarının verimli bir şekilde nasıl eğitilebileceğini göstermesidir. 34 katmanlı ResNet'in yapısı Şekil 4.14'te gösterilmiştir.



Şekil 4.14. 34 Katmanlı ResNet yapısı [2].

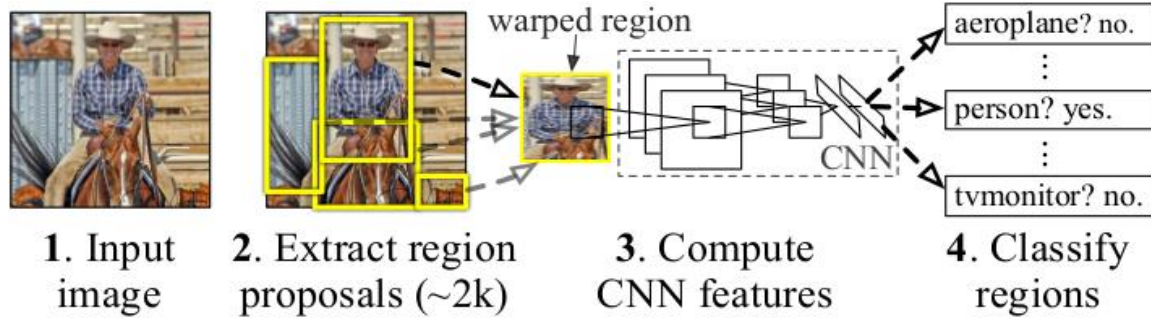
4.2.3 Nesne Tespit Uygulamalarında Konvolüsyonel Sinir Ağları

Sınıflandırma probleminden ayrı olarak, resimler üzerinde nesne tespiti yapılması daha karmaşık bir problemidir. Nesne tespiti, tahmin sonucu olarak gerçek sayıların üretildiği bir regresyon problemidir. Tespit işleminde, bir görüntü içindeki nesneler üç farklı değer kullanılarak etiketlenmektedir. Bunlar; en olası sınıf, sınırlayıcı kutu ve algoritmanın regresyon sonucu üretmiş olduğu kesinlik oranıdır. CNN'lerin kullanımı ile birlikte sınıflandırma hata oranı 2010 ve 2015 tarihleri arasında dokuz kattan daha fazla azaltılmıştır. CNN'lerin sınıflandırma konusundaki bu tatmin edici performansından esinlenerek, çok sayıda araştırmacı derin sinir ağı yapılarını nesne tespiti uygulamalarına uyarlama yöntemleri önermiştir.

Nesne tespiti uygulamalarının önemli bir adımı nesne bulunabilecek alanların çıkarımıdır. Nesne bulunma olasılığı olan alanların bulunmasına yönelik geliştirilmiş algoritmalara bölge teklifi algoritmaları denmektedir. Bu işlem farklı büyüklüklerde ve farklı kaydırma miktarıyla resim üzerinde çerçeveler kaydırarak yapılabileceği gibi daha hızlı ve başarıyı yüksek sonuçlar elde etmek için çeşitli algoritmalara dayalı bölge teklifleri de üretilebilmektedir. Günümüzde, objectness [82], selective search [65], category independent object proposals [83] gibi algoritmalar sunulmuştur. Bu algoritmalar çeşitli CNN'ler ile birleştirilerek elde edilen bölge tekliflerinde aranan nesneler saptanmaya çalışılmıştır. Bu bölümde son yıllarda nesne tespiti işlemi için önerilen CNN yapıları incelenmiştir.

4.2.3.1 Regions with CNN features (R-CNN)

R-CNN, R. Girshick ve ark tarafından 2013 yılında önerilmiştir [62]. Kendinden önceki nesne tespiti algoritmalarına göre %30'luk bir geliştirme ile başarılı sonuçlar elde edildiği öne sürülmüştür. Bunun için kendinden önceki on yıldır SIFT [63] ve HOG [64] gibi algoritmalara dayalı uygulamalardan farklı olarak özellik çıkarımı için CNN'lerden yararlanmış ve bunu bölge teklifi algoritmaları ile birleştirmiştir.



Şekil 4.15. R-CNN çalışma prensibi [62].

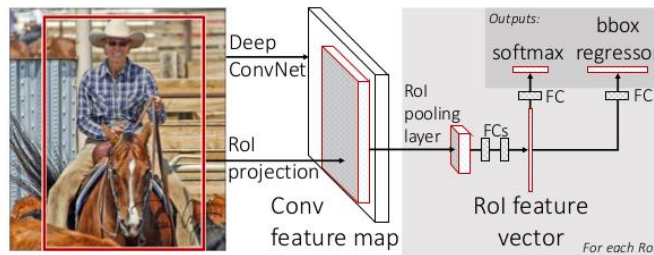
R-CNN algoritmasının işleyişi Şekil 4.15'te görülmektedir. İlk olarak alınan girdi verisi üzerinde ilgi alanları (Regions of Interest - RoI) oluşturulmuştur. İlgi alanları, kategoriden bağımsız olarak üretilen ve içerisinde nesne ihtiva etme olasılığı yüksek olan sınırlayıcı kutulardır. R-CNN algoritmasında ilgi alanlarının üretimi için Selective Search algoritması kullanılmıştır. Ancak yazarlar herhangi bir bölge teklifi algoritmasının da kullanılabileceğini belirtmiştir. İkinci adım olarak, ilgi alanı olarak belirlenen her bir veri kırılıp AlexNet CNN yapısına girdi olarak gönderilmektedir. Sonuç olarak özellikler, beş

adet konvolüsyon katmanı ve iki adet tam bağlı katman kullanılarak elde edilir. Algoritmada girdi olarak kullanılan resimlerin boyutu 227×227 ve renkli resimlerdir. Daha sonra elde edilen özellikler son sınıflandırmayı sağlayan destek vektör makinesine (DVM) gönderilmiştir ve girdi olarak verilen alanda aranan nesnenin bulunup bulunmadığı test edilmiştir. Ağın eğitimi aşama aşama gerçekleştirilmiştir. İlk olarak özelliklerin çıkarılması için kullanılacak olan CNN büyük bir veri seti üzerinde eğitilmiştir. Daha sonra DVM, CNN'den gelen özellik haritalarına göre doğru sınıflandırma yapacak şekilde eğitilmiştir. En son olarak da bölge teklifi algoritması uygulanmıştır.

R-CNN nesne tespiti için CNN tabanlı önerilen ilk yöntem olduğundan dolayı yol gösterici bir nitelik taşımaktadır. Ancak, ilk olmanın dezavantajlarını da barındırmaktadır. Daha sonra R-CNN yine aynı yazarlar tarafından geliştirilmiştir. R-CNN geliştirildiğinde R-CNN'in üç ana problemi vurgulanmıştır. Bu problemlerden ilki, eğitimin ayrı aşamalardan oluşması, ikincisi eğitimin maliyetli olmasıdır. Her bölge teklifi için özellikler çıkarılarak kaydedilmekte ve bu işlem hem DVM hem de bölge teklifi algoritması için uygulanmaktadır. Bu durum eğitimin günlerce sürmesi ve çok fazla depolama alanına ihtiyaç duyulmasına yol açmaktadır. Üçüncüsü ve en önemlisi ise grafik işlemcisinde çalışılsa dahi eğitimden sonra tespit işlemi için yaklaşık bir dakika gibi bir süre gerekmesidir. Bunun nedeni her bölge teklifi için aynı nesneyi gösterse veya üst üste binse dahi aynı işlemlerin yine de tüm bölge teklifleri için uygulanmasıdır. Bu üç problem R-CNN'in önemli eksiklikleri olarak görülmektedir. Daha sonraki yıllarda R-CNN geliştirilerek daha verimli sonuçlar alınmıştır.

4.2.3.2 Fast R-CNN

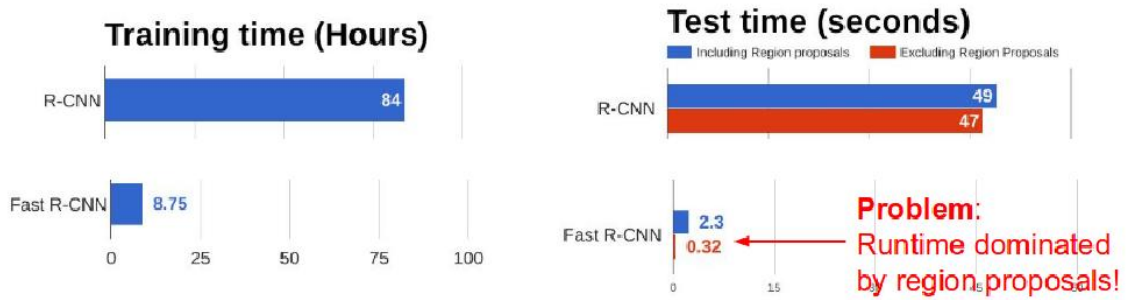
Fast R-CNN [48] Girshick ve ark. tarafından 2015 yılında R-CNN'in geliştirilmiş hali olarak sunulmuştur. Geliştirmenin ana fikri CNN katmanındaki hesaplamaların her bölge teklifi için ayrı ayrı değil tüm resim için bir defa yapılmasıdır. Fast R-CNN'in çalışma mantığı Şekil 4.16'da gösterilmiştir.



Şekil 4.16. Fast R-CNN çalışma prensibi [48].

Fast R-CNN nesne tespiti yapılacak resmi girdi olarak alır; konvolüsyon ve maksimum havuzlama katmanlarından geçirilir. Aynı zamanda R-CNN’de olduğu gibi bölge teklifleri CNN’den ayrı olarak oluşturulur. Daha sonra, elde edilen özellikler ve bölge teklifleri, ilgi alanı havuzlama katmanına gönderilir. Bu katmanda her bir bölge önerisi $n \times m$ boyutlu alanlara bölünür ve her bölüme kendi içinde maksimum havuzlama işlemi uygulanır. Bu sayede bölge teklifleri farklı boyutta olsa dahi sonuçta elde edilen özellikler her bölge teklifi için $n \times m$ boyutunda olmaktadır. Böylece her bölge teklifi için ayrı özellikler elde edilmiş olur. Elde edilen özellikler, sonunda nesne sınıflarının olasılıklarının hesaplandığı softmax katmanı ve sınırlayıcı kutuların koordinatlarının hesaplandığı regresyon katmanı olmak üzere iki katman olan tam bağlı katmana gönderilir ve çıktılar elde edilir.

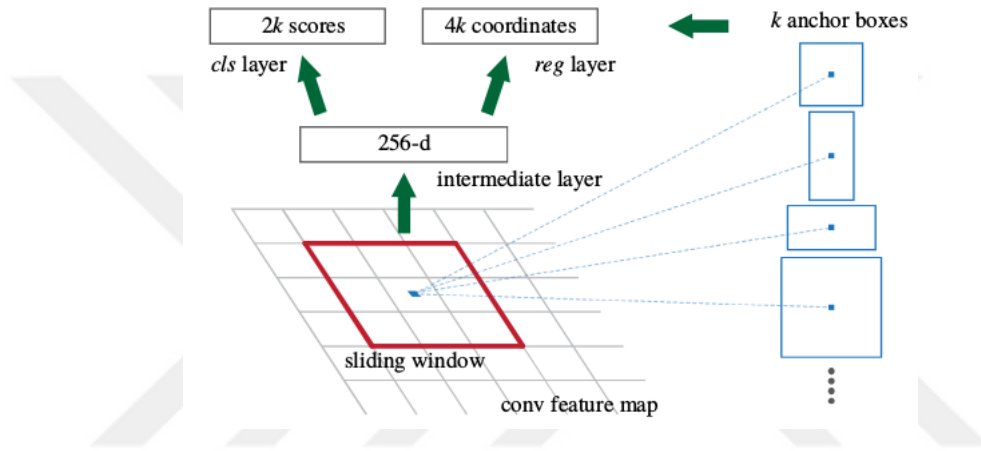
Fast R-CNN’in öne çıkan özelliği her bölge teklifi için ayrı ayrı CNN hesaplarının yapılmamasıdır. Bu sayede yazarlara göre, Fast R-CNN, eğitim süresinde 9 kat azalma ile R-CNN’den daha etkili bir şekilde eğitilmektedir. Eğitim aşamasında R-CNN’de olduğu gibi daha önceden eğitilmiş bir CNN kullanılmıştır. Ek olarak, her nesneyi sınıflandırmak için farklı bir DVM eğitimi yerine, sınıf olasılıklarını doğrudan üreten tek bir softmax katmanı kullanılmıştır. Bu durumda, bir sinir ağı ve birçok DVM’nin yerine, eğitilecek sadece bir sinir ağı kalmakta ve bu eğitim hızını önemli derecede kısaltmaktadır. Yazarlara göre Fast R-CNN, R-CNN’e göre VGGNet gibi derin ağları 9 kat daha hızlı eğitirken, test süresini 213 kat azaltmaktadır. Ancak Fast R-CNN’in hız bakımından dar boğaza uğradığı nokta bölge tekliflerinin üretilmesidir [4]. Fast R-CNN Selective Search algoritması ile bölge teklifleri oluşturmaktadır ve bu işlem yaklaşık 2 saniye sürmektedir [48]. Bu değer R-CNN test süresinin %86’sıdır. Bu durum daha sonra Faster R-CNN algoritması ile çözülmüştür. Şekil 4.17’de çalışma süresi bakımından R-CNN ve Fast R-CNN modellerinin karşılaştırılması gösterilmiştir.



Şekil 4.17. R-CNN ve Fast R-CNN çalışma süreleri [48,62].

4.2.3.3 Faster R-CNN

Faster R-CNN, Ren ve ark. tarafından Fast R-CNN algoritmasının en büyük zaman harcadığı kısım olan bölge teklifleri üretme adımı geliştirilerek sunulmuştur [4]. Algoritmanın temel mantığı konvolüsyon katmanlarını hem nesne tespiti hem de bölge tekliflerinin üretilmesi için ortak kullanımına dayanmaktadır. Bölge tekliflerinin üretildiği konvolüsyon katmanına Region Proposal Network (RPN) adı verilmiştir. RPN yapısı Şekil 4.18’de verilmiştir. Bölge tekliflerinde kullanılan harici fonksiyonun çıkarılması dışında Fast R-CNN yapısı kullanılmıştır.



Şekil 4.18. RPN yapısı [4].

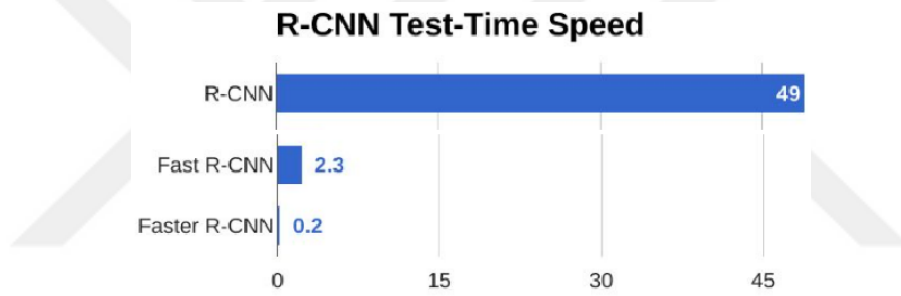
RPN, girdi olarak bir resmin özellik özellik çıkarıcı ağdan elde edilen özellik haritasını alır ve çıktı olarak her birinin nesne olma ihtimali ile nesne bulunması beklenen dikdörtgen sınırlayıcı kutular üretir. Bunu yaparken RPN ve CNN yapısının belli katmanları ortak olarak kullanması sağlanmıştır. Testler yapılırken 5 paylaşımlı kullanılabilen katmana sahip ZFNet ve 13 paylaşımlı kullanılabilen katmana sahip VGGNet kullanılmıştır.

Bölge tekliflerinin oluşturulması için en son paylaşımlı kullanılan konvolüsyon katmanından çıktı olarak elde edilen özellik haritası üzerinde bir pencere kaydırılmaktadır. Pencere boyutu 3x3 olarak seçilmiştir. Pencerenin kaydırıldığı her noktada k (bu çalışma için 9) adet, pencereyi merkez alan sınırlayıcı kutu (anchor) üretilmektedir. Daha sonra bu kutuların nesne barındırma olasılıkları hesaplanması ve boyut/pozisyon değerlerinin optimize edilmesi için sonraki sınıflandırma ve sınırlayıcı kutu regresyonu yapılan katmanlara gönderilmektedir.

Faster R-CNN’in uygulamasında ilk olarak hem bölge teklifi hem de nesne tespiti için kullanılan ağlar eğitilmiştir. Daha sonra bu ağlar birleştirilerek toplam çalışacak olan ağ elde

edilmiştir. Eğitilen bu ağ girdi olarak bir resim almakta ve paylaşılan konvolüsyon katmanları boyunca resmin özellikleri elde edilmektedir. Özellikler daha sonra RPN'e girdi olarak verilir ve bölge teklifleri üretilir. Üretilen bölge teklifleri ve özellikler beraber Fast R-CNN'de olduğu gibi ilgi alanı havuzlama, sınıflandırma ve sınırlayıcı kutu optimizasyon katmanlarına gönderilerek çıktılar elde edilir.

Paylaşımlı konvolüsyon katmanlarının kullanılmasıyla bölge teklifleri neredeyse masrafsız bir yolla üretilmiş olmaktadır [4]. Ayrıca baştan sona tek bir ağ üzerinde hesaplama yaparak sonuçlar elde edilmektedir. Bu, daha önce CPU'da üretilen bölge tekliflerinin artık GPU üzerinde üretilebilmesi dolayısı ile çok daha hızlı işlem yapılabilmesi anlamına gelmektedir. Bu sayede Fast R-CNN'de karşılaşılan dar boğaz problemi de ortadan kalkmış olmaktadır. Şekil 4.19'da R-CNN modellerinin hız bakımından karşılaştırılması gösterilmiştir.



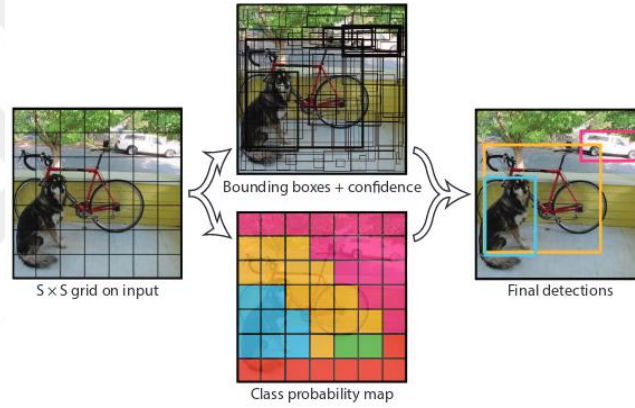
Şekil 4.19. R-CNN çalışma süreleri [4,48,62].

4.2.3.4 You Only Look Once (YOLO)

YOLO, hem nesnenin sınıflandırılması hem de yerlerinin tespit edilmesi için tek bir CNN ağı kullanan bir nesne tespit yöntemidir [56]. Daha önceki başlıklarda anlatılan algoritmalarda nesne tespit işlemi, sınıflandırma yöntemi ile çözülmekte idi. Ancak, YOLO nesne tespit işlemini bir regresyon problemi olarak ele almaktadır. Bu regresyon problemi içerisinde resim pikselleri, sınırlayıcı kutular ve sınıf olasılıkları hesaplanmaktadır. Resim algoritmaya, bölge teklifleri şeklinde değil bir bütün olarak gönderildiğinden dolayı YOLO oldukça hızlı bir yöntem olarak göze çarpmaktadır. Ancak hızı önemli ölçüde artırırken, doğruluk oranında bir kayıp yaşanmaktadır.

YOLO algoritmasında girdi olarak verilen resim, ilk olarak $S \times S$ boyutunda bir matris şeklinde bölünür. Eğer bir nesnenin merkezi bir matris hücresine girerse o matris hücresi o nesneyi algılamaktan sorumlu olur. Her matris hücresi, B sınırlayıcı kutu ve bu kutulara ait

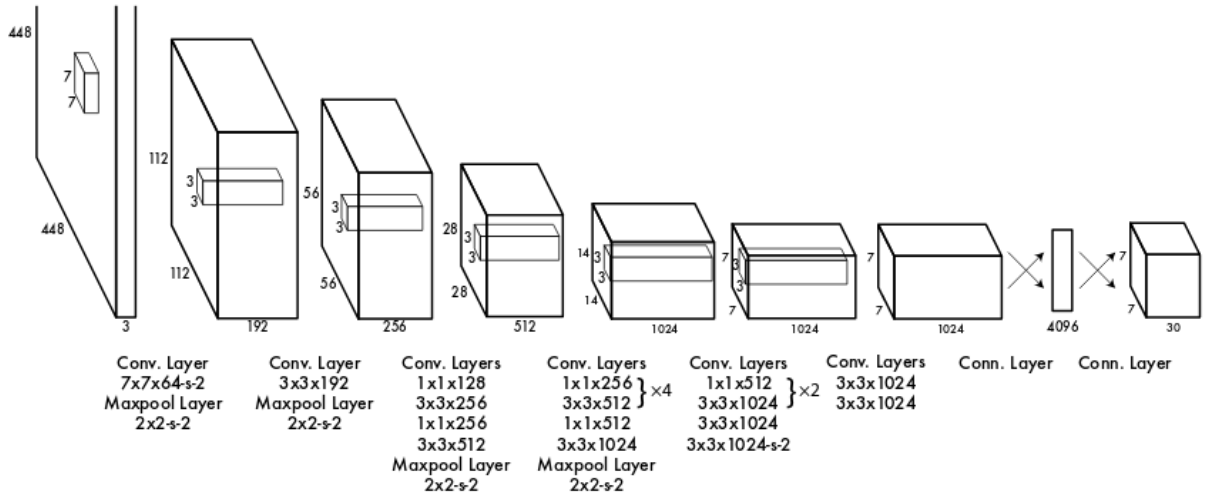
olasılık puanı tahmin etmektedir. Bu olasılık puanı, modelin kutunun bir nesne içerdiğinden ne kadar emin olduğuna ve kutunun tahminini ne kadar doğru yaptığı gösterir. Olasılık puanları, regresyon ile elde edilen sınıf olasılığı ve tahmin edilen sınırlayıcı kutunun gerçekte olması gereken sınırlayıcı kutuya oranının çarpılması ile hesaplanmaktadır. Eğer, matris hücresinde hiç bir nesne yoksa bu değer sıfır olmalıdır. Üretilen her sınırlayıcı kutu beş tane değer tahmin etmektedir. Bunlar, matris hücresine göre hesaplanan sınırlayıcı kutunun merkezi (x,y), tüm resme göre hesaplanan sınırlayıcı kutunun en ve boyu (w,h) ve sınırlayıcı kutunun gerçek sınırlayıcı kutuya oranı ile elde edilen olasılık puanıdır. Ayrıca her matris hücresi de kendisinin hangi sınıfa ait bir resim parçasını içerdiğini belirtmek üzere C tane sınıf tahmini oluşturmaktadır. YOLO algoritmasının kabaca çalışma prensibi Şekil 4.20’de gösterilmiştir.



Şekil 4.20. YOLO çalışma prensibi [56].

YOLO, 20 sınıftan nesneler içeren Pascal VOC veri setinde değerlendirilmiştir. Bu durumda sınıf sayısı $C=20$ olmaktadır. Ayrıca yazarlar $S=7$, $B=2$ olarak eğitimi gerçekleştirmiştir. Dolayısı ile toplamda sonucu tahmin etmek için $S \times S \times (B \times 5 + C) = 7 \times 7 \times 30$ boyutunda bir vektör elde etmişlerdir.

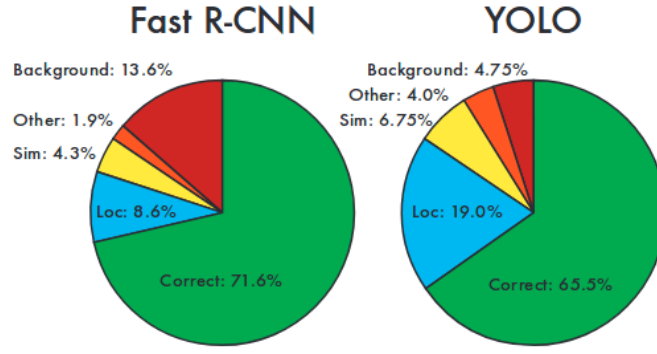
YOLO ağ mimarisi, sınıflandırma için kullanılan LeNet ağından esinlenilmiştir. YOLO 24 konvolüsyon katmanına onları takip eden 2 tam bağlı katmandan oluşmaktadır. LeNet’te kullanılan inception modülleri yerine, boyut azaltma için 1x1 konvolüsyon katmanları ve sonrasında 3x3 konvolüsyon katmanları kullanılmıştır. Ağın başlangıçtaki konvolüsyon katmanları görüntünün özelliklerini çıkarırken tam bağlı katmanlar çıkış olasılıklarını ve koordinatlarını hesaplamaktadır. YOLO’nun ağ mimarisi Şekil 4.21’de gösterilmiştir.



Şekil 4.21. YOLO ağ mimarisi [56].

Özelliklerin çıkarıldığı konvolüsyon katmanlarının eğitimi için 1000 sınıflı ImageNet yarışması veri seti kullanılmıştır. Eğitim için Resim 44’de görülen 20 konvolüsyon katmanı, sonrasında buna bağlı bir ortalama havuzlama katmanı ve tam bağlı katman kullanılmıştır. Bu ağ yaklaşık olarak bir hafta boyunca eğitilerek ImageNet 2012 doğrulama veri seti üzerinde %88 gibi bir sınıflandırma doğruluk oranı elde edilmiştir. Daha sonra bu ağa nesne tespiti işlemini gerçekleştirmesi için dört konvolüsyon ve iki tam bağlı katman eklenmiştir. Oluşan ağın son katmanında hem sınırlayıcı kutu hem de sınıflandırma sonuçları elde edilmiştir.

Yazarlar aynı zamanda YOLO ve diğer nesne tespiti algoritmalarını karşılaştırmıştır. En iyi sonuçların elde edildiği Fast R-CNN algoritmasının arka plan hataları, sanat resimleri sınıflandırma gibi problemlerinde daha iyi sonuçlar alırken doğruluk oranı konusunda geride kaldığını belirtmişlerdir. Ayrıca YOLO’nun, her matris hücresinin sadece bir nesne algılayacak şekilde tasarlanması nedeniyle, birbiri arkasında kalan veya küçük nesnelerin algılanmasında zayıf olduğu belirtilmiştir. Şekil 4.22’de YOLO ve Fast R-CNN modellerinin karşılaştıkları hatalar kategoriler bazında gösterilmiştir.

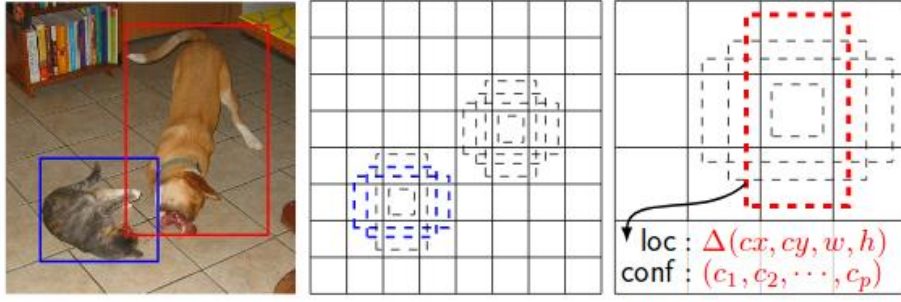


Şekil 4.22. Fast R-CNN ve YOLO algoritmalarının arka plandan, benzer sınıftan, lokalizasyondan ve diğer nedenlerden kaynaklanan hatalarının ve doğruluk oranlarının karşılaştırılması

4.2.3.5 Single Shot Multibox Detector (SSD)

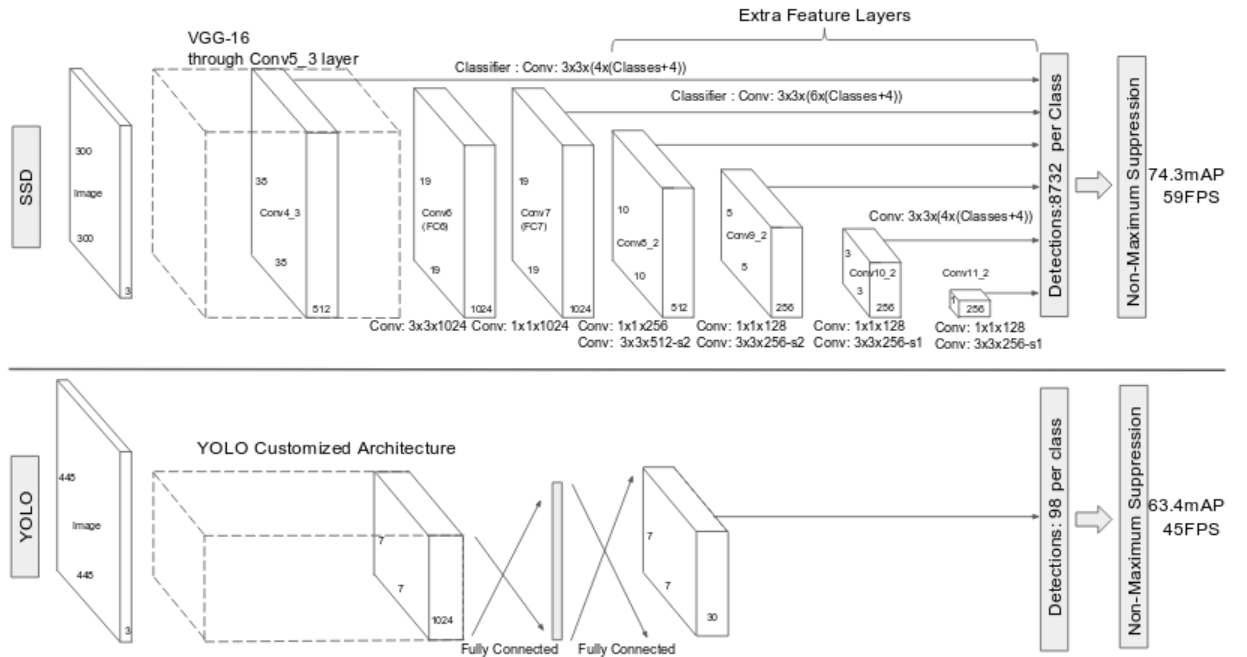
SSD de, YOLO gibi nesne tespiti işlemini regresyon problemi olarak ele alan bir algoritmadır [55]. SSD bölge teklifleri oluşturmamakta; dolayısı ile resmin belli parçaları için ayrı ayrı işlemler gerçekleştirmemektedir. Konvolüsyonel sinir ağını tek bir sefer işleyerek sonuçları elde etmektedir. Bu sayede çok hızlı şekilde sonuç üretilmektedir.

SSD eğitim için resimle birlikte, nesnelerin gerçek yerlerini belirten sınırlayıcı kutulara gereksinim duymaktadır. YOLO’da olduğu gibi SSD’de de girdi olarak alınan resim matris hücrelerine bölünür. Daha sonra dönüşümlü bir biçimde, farklı ölçek oranlarına (ör. 8x8 ve 4x4) sahip birkaç özellik haritasında farklı en boy özelliklerine sahip varsayılan sınırlayıcı kutular kullanılarak, bu kutuların nesne içerip içermediği kontrol edilir. Bu sayede boyutu farklı nesnelerin algılanması kolaylaştırılmaktadır. Her varsayılan sınırlayıcı kutu için hem kutunun olması gereken boyutu hem de her sınıf için olasılık puanları hesaplanır. Eğitim sırasında ilk olarak, varsayılan sınırlayıcı kutular ile olması gereken sınırlayıcı kutular eşleştirilir. Örneğin kedi ve köpek resmi olan bir imgede, kedi için bir, köpek için ise iki sınırlayıcı kutuda eşleşme sağlanmış olsun. Eşleşen sınırlayıcı kutular pozitif tahminler, geri kalan kutular ise negatif olarak sınıflandırılır. Daha sonra modelin hata oranı lokalizasyon hatası ve olasılık hatalarının ağırlıklı toplamı olarak hesaplanır ve elde edilen hataya göre eğitim devam eder. Şekil 4.23’te varsayılan sınırlayıcı kutular ile gerçek sınırlayıcı kutuların bulunması gösterilmiştir.



Şekil 4.23. Varsayılan sınırlayıcı kutuların gerçek sınırlayıcı kutuları tahmin etmesi [55].

SSD ağında özellik çıkarımı için daha önceden eğitilmiş olan VGG ağından yararlanılmıştır. Ayrıca VGG’de konvolüsyon katmanlarından sonra gelen iki tam bağlı katman konvolüsyon katmanına dönüştürülmüştür ve beşinci katmanda bulunan havuzlama adımında kaydırma miktarı 2’den 1’e düşürülmüş ve filtresi 2x2 boyutundan 3x3 boyutuna değiştirilmiştir. Bu iki katmandan sonra gelen son tam bağlı katmanda ise seyreltme (dropout) katmanları devre dışı bırakılmıştır. Ağın yapısı Şekil 4.24’te verilmiştir.



Şekil 4.24. SSD ve YOLO mimarilerinin karşılaştırılması [55].

Yapılan testler sonucunda çok fazla veri ile birlikte SSD’nin iyi sonuç veren algoritmalar ile yarışacak seviyede sonuçlar üretebileceği gösterilmiştir. Ayrıca kendisi gibi gerçek zamanlı nesne tespiti yapabilen YOLO’ya göre hem girdi olarak daha küçük resimlerde çalışabildiği hem de daha yüksek başarımlar elde edebildiği gösterilmiştir. Ancak

SSD de çok küçük nesnelerin algılanmasında zorluklar yaşamaktadır. Bu problemin aşılabilmesi için eğitim verilerinin çok fazla çeşitlendirilmesi çözüm olarak sunulmuştur. Şekil 4.25'te SSD ve R-CNN modellerinin doğruluk açısından karşılaştırılması gösterilmiştir.

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast [6]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster [2]	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster [2]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster [2]	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	81.6	86.6	88.3	82.4	76.0	66.3	88.6	88.9	89.1	65.1	88.4	73.6	86.5	88.9	85.3	84.6	59.1	85.0	80.4	87.4	81.2

Şekil 4.25. SSD, Fast R-CNN ve Faster R-CNN doğruluk oranlarının karşılaştırılması [55].

5. DERİN ÖĞRENME YÖNTEMLERİ KULLANARAK ADLİ BİLİŞİM İNCELEMELERİNDE DELİL ÇIKARIMININ GERÇEKLEŞTİRİLMESİ

Bu tez çalışmasında, adli delil incelemelerinin analiz aşamasında görüntü dosyalarının incelenmesinde karşılaşılan zaman kaybı ve yanlışlık yapma ihtimalinin fazla olması gibi problemlerinin önüne geçilmesi amacıyla, dijital delilden elde edilen görüntü dosyalarından inceleyici için anlamlı olabilecek verilerin otomatik olarak saptanmasını sağlayan bir sistem önerilmiştir. Gerçekleştirilen uygulama ses ve video dosyalarına da genişletilebilir bir uygulamadır ve daha sonraki çalışmalarda bu alanda çalışmalara devam edilmesi düşünülmektedir. Görüntü dosyalarından anlamlı verinin çıkarılması, adli bilişim açısından suç teşkil eden görüntülerin algılanması anlamına gelmektedir. Bunun için adli delil inceleme yazılımlarından elde edilen görüntü dosyalarının tek tek incelenmesi yerine içinde suçla ilişkili olabilecek nesnelerin tespit edilmesini sağlayacak bir sistemin geliştirilmesi amaçlanmıştır. Sistemin gerçekleştirilmesi için son yıllarda nesne sınıflandırma ve tanıma alanında yüksek başarılar elde etmiş olan derin öğrenme yöntemlerinden [4,48,55-59,62] yararlanılmıştır. Otomatik delil algılama sisteminin temelinde konvolüsyonel sinir ağları ile gerçekleştirilen nesne tespit uygulamaları yatmaktadır. Konvolüsyonel sinir ağı suç ile ilişkin nesneleri tespit etmek amacıyla eğitilerek adli delil inceleme yazılımlarının çıkardığı sonuçların bu ağ ile taranması amaçlanmaktadır. Sonuç olarak yapay sinir ağı tarafından tespit edilen görüntü verileri adli incelemeciye sunularak şüpheli veya suçla direkt olarak ilişkili verilerin hızlı bir şekilde ayırt edilmesi sağlanacaktır. Derin öğrenme yöntemlerinin kullanılması için derin öğrenme uygulamaları geliştirmek amacıyla geliştirilen çeşitli frameworkler kullanılmaktadır. Birçok farklı framework bulunmakla beraber bu çalışmada, yaygın olarak kullanılan, uygulama ve kaynak bakımından zengin, nesne tespiti uygulamaları için ayrı bir API geliştirmiş olan ve çeşitli eğitilmiş modelleri kullanıcılara sunan tensorflow frameworkü [84] kullanılmıştır. Tensorflow, kullanıcılara model zoo [85] adında eğitilmiş modeller sunduğu bir ortama sahiptir. Bu platformda çeşitli veri setleri üzerinde farklı modeller eğitilerek hız ve doğruluk oranları ile birlikte kullanıma sunulmuştur. Tez çalışmasında burada yer alan modeller kullanılmıştır. Adli delillerden elde edilecek delil verileri bu tez çalışması için insan içeren ve silah içeren dosyalardır. COCO [86], Kitti [87] ve Open Images [88] veri setleri üzerinde eğitilmiş modeller arasında insan etiketi bulunurken, silah resimlerini saptayan bir model bulunmamaktadır. Bu nedenle, insan

içeren resimler önceden standart veri setleri ile eğitilmiş modellerin çıktıları değiştirilerek elde edilirken, silah resimlerinin saptanması amacıyla eğitilmiş model tekrardan eğitilerek silah resimlerini saptaması sağlanmıştır. Gerçekleştirilen uygulama adımları sırasıyla; veri setinin oluşturulması, verilerin etiketlenmesi, tensorflow framework kurulumu, eğitilmiş modellerin eğitim için hazırlanması ve sonuçların değerlendirilmesi şeklindedir.

5.1 Transfer Learning

Derin öğrenme uygulamalarında günümüzde başarılı modelleri sıfırdan başka bir eğitim seti ile eğitmek yüksek işlem kapasitesi ve depolama alanına ihtiyaç duyması, çok güçlü grafik işlemcilerle dahi eğitimin günlerce sürmesi ve yeterli miktarda verinin toplanmasının çok zor olmasından dolayı kolaylıkla gerçekleştirilememektedir. Bu nedenle araştırmacılar çalışmalarında transfer learning adı verilen yöntemi kullanmaktadırlar. Transfer learning sıfırdan eğitime göre çok daha hızlı ve iyi sonuçlar alınabilmesine olanak sağlamaktadır [89].

Transfer learning, bir sinir ağının eğitimine başlarken daha önceden farklı özellikler için eğitilen ağırlıkları kullanarak modelin ilk değerlerini atanmasını öneren bir yöntemdir. Bu yöntemin iyi sonuçlar vermesinin nedeni, binlerce kategori için başarılı bir genelleme yapabilen ağırlıkların yeni eğitimecek olan sınıf için de başarılı bir genelleme yapabilmesidir. Üç ana yöntem ile transfer learning yöntemi gerçekleştirilebilmektedir.

Bunlardan ilki eğitilmiş olan konvolüsyonel sinir ağının ağırlıklarının değiştirilmeyip sadece ağın sonundaki sınıflandırıcı katmanın yeniden eğitildiği durumdur. Bu yöntemde eğitilmiş konvolüsyonel sinir ağı, yeni eğitim kümesi için özellik çıkarıcı görevi görmektedir. Modelin sınıflandırıcı katmanı çıkarılarak yeni bir sınıflandırıcı katman eklenmektedir. Çıkarılan özellikler kullanılarak da modelin yeni sınıflandırıcısı eğitilmektedir. Örneğin, özellik çıkarıcı olarak AlexNet konvolüsyonel sinir ağı kullanıldığında, kendi sınıflandırıcısından önce her resim için 4096-D boyutunda bir vektör oluşturmaktadır. Transfer learning yönteminde hazır ağırlıklar ile çıkarılan bu özellikler yeni eklenen sınıflandırıcıyı eğitmek için kullanılır.

İkinci yöntemde eğitilmiş modelin son katmanının (sınıflandırıcı katman) değiştirilmesinin yanında, yeni veri seti kullanılarak bazı katmanlardaki ağırlıklar da geri yayılım algoritması ile güncellenmektedir. Tüm katmanlardaki ağırlıkların güncellenmesi de mümkündür ancak küçük bir veri seti ile karmaşık bir sinir ağı eğitilmiş olduğundan aşırı

eğitim problemi ortaya çıkabilmektedir. Bu durumdan kaçınmak amacıyla genellikle ilk katmanlar sabit bırakılır ve üst seviye bazı katmanlar güncellenir. Çünkü, konvolüsyonel sinir ağlarının ilk katmanlarında kenarlar, renk özellikleri gibi genel amaçlı özellikler çıkarılırken ilerleyen katmanlarda, veri seti için spesifik özellikler çıkarılmaktadır.

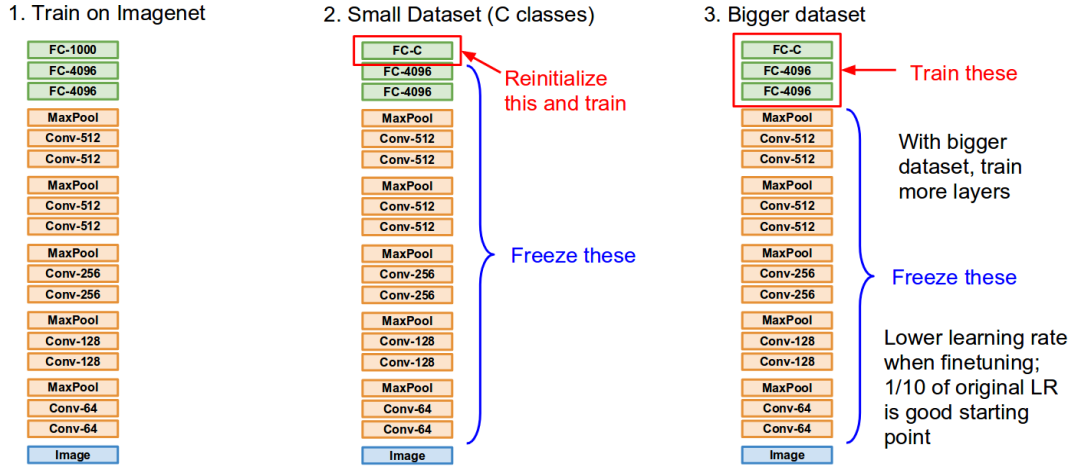
Üçüncü yöntem ise eğitilmiş olan modellerin direkt olarak kullanılmasıdır. Araştırmacılar çeşitli platformlarda eğitilmiş modellerini ve bu modellerin ağırlıklarını paylaşmaktadır. Yeniden eğitimin mümkün olmadığı veya ihtiyaca karşılık alınabildiği durumlarda bu ağırlıklar hem özellik çıkarımı hem de sınıflandırma için kullanılabilir.

Bu yöntemlerden hangisinin kullanılacağı veri setinin durumu ve sahip olunan işlem gücüne göre değişmektedir.

Tablo 5.1. Veri seti durumuna göre tercih edilebilecek transfer learning yöntemleri

Veri seti \ Benzerlik	Benzer Veri Seti	Farklı Veri Seti
Küçük Veri Seti	Son katmandaki sınıflandırıcı değiştirilir	Farklı katmanlarda sınıflandırıcı eğitilir
Büyük Veri Seti	Daha az katman eğitilir (finetune)	Daha fazla katman eğitilir (finetune)

Tablo 5.1’de veri setinin büyüklüğüne bağlı olarak hangi yöntemin tercih edilebileceği gösterilmiştir. Buradan görüldüğü gibi küçük bir veri seti kullanılarak yapılacak uygulamalarda sadece çıkış katmanı değiştirilerek bu katmanın eğitilmesi daha iyi bir seçim olarak kabul edilmektedir. Bunun nedeni az veri ile büyük ağlar eğitildiğinde aşırı eğitim problemi ile karşılaşılmasıdır. Büyük veri seti için ise son katmanla birlikte daha fazla katmanın eğitilmesi önerilmektedir. Böylece doğruluk oranı daha yüksek eğitim gerçekleştirilmektedir. Şekil 5.1’de veri seti büyüklüğüne göre izlenebilecek transfer learning yöntemleri gösterilmiştir.



Şekil 5.1. Veri Seti büyüklüğüne göre izlenebilecek transfer learning yöntemi [90].

5.2 Veri Setinin Oluşturulması ve Verilerin Etiketlenmesi

Adli delil incelemelerinde, sıklıkla karşılaşılan problemlerden birisi görsel verilerin incelenmesidir. Görsel veriler, yasadışı maddelerin veya araçların ticareti, cinayet aracı, terör örgütüne ait resim, video veya amblemler gibi delillerin arandığı durumlarda incelenmektedir. Bu tez çalışmasında cinayet veya yasadışı silah ticareti senaryoları üzerinde çalışmak üzere silah(tabanca) veri seti oluşturulmuştur. Silah tiplerinden tabanca sınıfı seçilmiştir daha sonraki çalışmalarda sınıfların çoğaltılması amaçlanmaktadır.

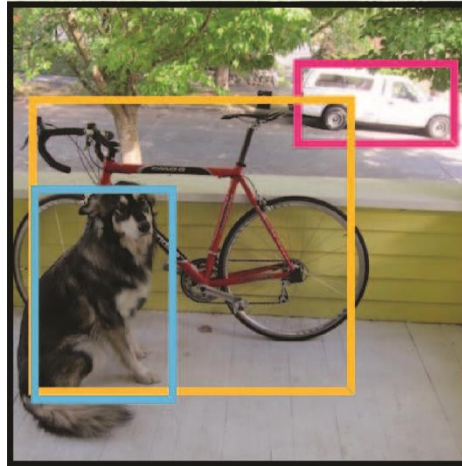
Derin öğrenme algoritmaları eğitimin sıfırdan gerçekleştirildiği durumlarda çok fazla veriye ihtiyaç duymaktadır. Bu miktarda verinin tek kişi tarafından toplanması ve etiketlenmesi çoğu zaman çok fazla zaman almakta hatta imkânsız olmaktadır. Bu nedenle gerçekleştirilen çalışmada transfer learning yönteminden yararlanılmıştır. Şekil 5.1’de görüldüğü üzere küçük veri setleri üzerinde başarılı bir şekilde eğitim yapılması transfer learning kullanılarak mümkün olmaktadır.

Çalışmada kullanılmak üzere google arama motoru, imagenet veri seti ve çeşitli stok resim sitelerinden resimler toplanmakla birlikte Olmos ve ark. [91] tarafından gerçekleştirilen çalışmada kullanılan veri seti de toplanan resimlere eklenmiştir. Toplam veri seti çeşitli büyüklüklerde 6660 adet resimden oluşmuştur. [91]’de Olmos ve arkadaşları 3000 resim kullanarak başarılı sonuçlar elde etmiştir. Tez çalışmasında daha geniş bir veri seti kullanılarak başarının artırılması amaçlanmıştır.



Şekil 5.2. Veri setinde bulunan örnek resimler

Şekil 5.2’de veri setinde bulunan resimlerden bazıları gösterilmiştir. Resimlerin olabildiğince çeşitli açılardan tabanca resimleri içermesine dikkat edilmiştir. Ancak, nesne tespiti uygulamalarında, sınıflandırma uygulamalarında olduğu gibi resimlerin toplanması yeterli olmamaktadır. Nesnelerin yerlerinin belirtildiği sınırlayıcı kutuların eğitim için hazırlanması gerekmektedir. Bu sayede yapay sinir ağına nesnelerin resim içerisinde tam olarak nerede bulunduğunu tespit etmeyi öğretmek mümkün olmaktadır. Şekil 5.3’te nesnelerin sınırlayıcı kutular ile etiketlenmesi gösterilmiştir.



Şekil 5.3. Farklı nesneleri gösteren sınırlayıcı kutular [56].

Tez çalışmasında tüm resimler için sınırlayıcı kutuların oluşturulması için labelImg [92] açık kaynaklı programından yararlanılmıştır. labelImg programı kullanımı oldukça basit ve kullanışlı bir resim etiketleme programıdır. Sınırlayıcı kutuları bir xml dosyası olarak

temsil ederek kaydetmektedir. Oluşan xml dosyasının formatı Pascal VOC veri seti standart formatıdır ve imagenet gibi birçok bilgisayar görmesi ortamında standart olarak kullanılmaktadır. Şekil 5.4'te tabanca veri setinin oluşturulmasında labelling programı ile oluşturulan sınırlayıcı kutular gösterilmiştir.



Şekil 5.4. Labeling programı ile oluşturulan sınırlayıcı kutular

Şekilde görülen her bir kutuya karşılık gelen koordinat bilgileri, her resim için oluşturulan xml dosyalarında kaydedilmektedir. Bu dosyalarda koordinat bilgilerinden ayrı olarak resmin boyutu, derinliği, resimde işaretlenen nesnelerin etiketleri gibi veriler de kaydedilmektedir. Şekil 5.5'te örnek bir xml dosyasının kaydettiği veriler gösterilmiştir.

```
1 <annotation>
2   <folder>Definitiva</folder>
3   <filename>armas1400.jpg</filename>
4   <path>C:\Users\Rob\Desktop\Definitiva\armas (1400).jpg
5   <source>
6     <database>Unknown</database>
7   </source>
8   <size>
9     <width>500</width>
10    <height>351</height>
11    <depth>3</depth>
12  </size>
13  <segmented>0</segmented>
14  <object>
15    <name>handgun</name>
16    <pose>Unspecified</pose>
17    <truncated>0</truncated>
18    <difficult>0</difficult>
19    <bndbox>
20      <xmin>21</xmin>
21      <ymin>24</ymin>
22      <xmax>414</xmax>
23      <ymax>282</ymax>
24    </bndbox>
25  </object>
26 </annotation>
```

Şekil 5.5. Şekil 5.4'te gösterilen ilk resim için oluşturulan xml dosyası

Bu dosyalar tüm resim dosyaları için tek tek oluşturulmuştur. Son olarak resim dosyaları %20 test veri seti %80 eğitim veri seti olarak ayrılarak veri setinin oluşturulması tamamlanmıştır. Sonuçta 1332 adet resim test için, 5328 adet resim eğitim için kullanılmıştır.

5.3 Tensorflow Ortamının Kurulması

TensorFlow, yüksek performanslı sayısal hesaplama için açık kaynaklı bir yazılım kütüphanesidir. Esnek mimarisi, çeşitli platformlarda (CPU'lar, GPU'lar, TPU'lar) ve masaüstü bilgisayarlardan sunucu kümelerine ve mobil cihazlara kadar hesaplamanın kolayca yapılmasını sağlar. Orijinal olarak Google'ın AI organizasyonu içindeki Google Beyin ekibinden araştırmacılar ve mühendisler tarafından geliştirilen bu ortam, makine öğrenimi ve derin öğrenmeye güçlü bir destek vermektedir. Ayrıca sahip olduğu esnek hesaplama çekirdeği birçok bilimsel alanda kullanılmaktadır. Tez çalışmasında derin öğrenme alanındaki çalışmalar için de tensorflow kullanılmıştır.

Bu tez çalışmasında resim verileri üzerinde işlemlerin yapılması amaçlanmıştır. Bu nedenle işlemlerin grafik işlemcisi ile daha verimli ve hızlı bir şekilde gerçekleştirilmesini sağlamak için tensorflow'un GPU işlem desteğinden yararlanılmıştır. Bunun için NVIDIA Cuda ve cuDNN [93] yazılımlarının kurulumu gerçekleştirilmiştir. Böylelikle tensorflow yazılımının grafik işlemcisini görmesi ve onun üzerinden işlemleri yürütebilmesi sağlanmıştır.

İlk olarak grafik işlemcisi sürücüsü güncellenip Cuda yazılımının son sürümü indirilerek kurulmuştur. Daha sonra, grafik işlemcisi üzerinde konvolüsyon, pooling, normalizasyon, aktivasyon gibi işlemlerin verimli bir şekilde yapılabilmesini sağlayan Cuda kütüphanesi cuDNN kurulmuştur. Eğitim aşamasında kullanılacak olan adımların tümü bu kütüphane ile grafik işlemcisine aktarılmaktadır. Son olarak gpu destekli tensorflow dağıtımı kurularak deneylerin gerçekleştirileceği yazılım ortamının kurulumu tamamlanmıştır. Kurumlar sonunda Nvidia Quadro M4000 grafik sürücüsü 184.130 sürümü, Cuda 8 ve cuDNN 6 yazılımları, tensorflow 1.3.0 versiyonu çalışır durumda elde edilmiştir. Şekil 5.6'da grafik sürücüsü kütüphanelerinin ve donanımının düzgün ayarlandığı görülmektedir.

```
mstf@mstf-HP-Z840-Workstation:~$ nvidia-smi
Mon Jun 11 11:41:30 2018

+-----+
| NVIDIA-SMI 384.130                Driver Version: 384.130 |
+-----+-----+
| GPU  Name            Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
|  0  Quadro M4000     Off       | 00000000:04:00:0 | On         |           N/A       |
| 48%   45C   P0      42W / 120W | 474MiB / 8112MiB |      0%    Default   |
+-----+-----+

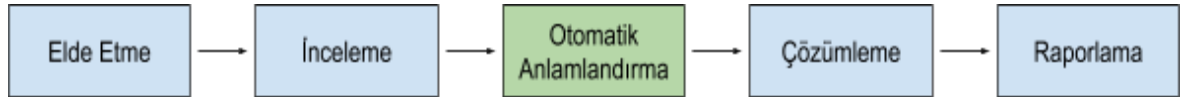
Processes:
+-----+
| GPU    PID    Type    Process name                      GPU Memory |
|-----|
|  0     1116    G      /usr/lib/xorg/Xorg                    253MiB |
|  0     1869    G      compiz                               116MiB |
|  0     2496    G      /usr/lib/firefox/firefox              3MiB |
|  0     4065    G      ...-token=9AE009E2A3B149180D6B224AF5364753 97MiB |
+-----+

mstf@mstf-HP-Z840-Workstation:~$ nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2016 NVIDIA Corporation
Built on Tue_Jan_10_13:22:03_CST_2017
Cuda compilation tools, release 8.0, V8.0.61
```

Şekil 5.6. Kurulan yazılımların doğrulanması

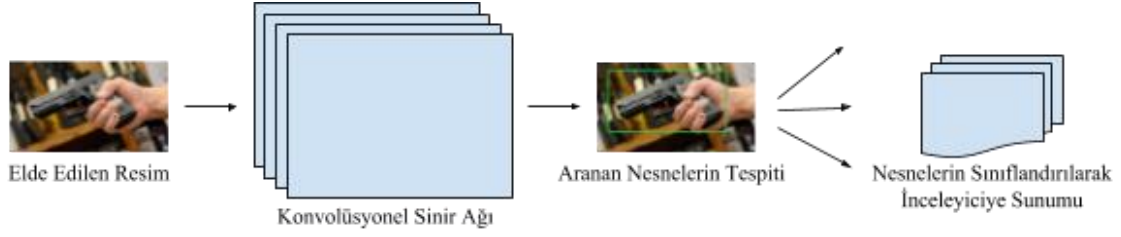
5.4 Metodoloji

Bu çalışmada adli delil incelemelerine yardımcı olması amaçlanan bir adli delil inceleme yöntemi önerilmiştir. Adli delil incelemelerinin hızlandırılması için resimlerin otonom olarak anlamlandırılarak filtrelenmesi incelemecilerin üzerine düşen işi hafifleterek daha sağlıklı bir inceleme süreci sağlayacaktır. Önerilen adli delil inceleme yöntemi Şekil 5.7.'de gösterilmiştir.



Şekil 5.7. Derin öğrenme kullanılarak oluşturulan adli delil inceleme aşamaları

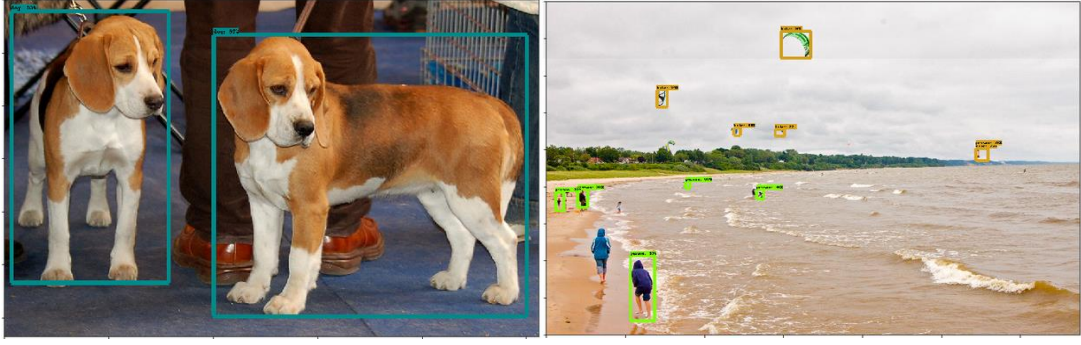
Klasik adli bilişim inceleme adımlarına eklenen otomatik anlamlandırma sınıfı bu tez çalışmasında gerçekleştirilen derin öğrenme yöntemlerini kullanmaktadır. Oluşturulan derin öğrenme modelleri dijital delilden incelenmek üzere elde edilen verileri sınıflandırarak inceleme sürecini hızlandırmaktadır.



Şekil 5.8. Otomatik anlamlandırma adımı

Otomatik anlamlandırma adımının blok diyagramı Şekil 5.8.'de gösterilmiştir. Dijital delillerden elde edilen görüntü verileri derin öğrenme ile aranan nesneler bazında sınıflandırılmakta ve incelemeciye içeriğe göre kategorilere ayrılmış şekilde sunulmaktadır.

Arada kullanılan konvolüsyonel sinir ağı bölüm 5.1'de anlatıldığı gibi önceden eğitilmiş büyük bir ağı yeni veriler için ayarlanması ile oluşturulmuştur. Bunun için ilk olarak tensorflow object detection API [94] bilgisayara indirilerek kurulum adımları takip edilmiştir. Object detection API; kurulum, veri setinin hazırlanması, bulut bilgisayarda çalışma, eğitim ayarlarının yapılması gibi birçok konuyu içeren detaylı bir dokümantasyona sahiptir. API kurulumu tamamlandığında, yapay sinir ağları eğitilirken kullanılacak sınırlayıcı kutu üretimi, resimlerden eğitilmiş modeller kullanılarak özellik çıkarımı, eğitilen sinir ağının kaydedilmesi, resim ön işleme gibi eğitim için gerekli birçok yardımcı kod API tarafından sağlanmaktadır. Kurulumun başarılı bir şekilde tamamlandığının anlaşılması için örnek nesne tespiti programı çalıştırılmıştır. Bu program COCO veri seti üzerinde eğitilmiş bir SSD modelini bilgisayara indirerek API tarafından sağlanan örnek resimdeki nesneleri tespit etmektedir. Programın çıktısı Şekil 5.9'da gösterilmiştir. Programda örnek resimler MobileNet özellik çıkarıcı ağı kullanan SSD ağı ile tespit edilmektedir. Resimde görüldüğü gibi COCO veri setinde bulunan köpek, insan ve uçurtma nesneleri örnek resimlerin içinde başarı ile tespit edilmiştir. İkinci resimdeki bir insanın tespit edilememesi SSD modelinin doğruluktan daha çok hıza odaklı bir model olmasıdır. Doğruluk oranı daha yüksek modellerde tespit sonuçları daha farklı çıkabilmektedir.



Şekil 5.9. Tensorflow object detectidon api test programının çıktısı

Bu noktadan sonra yapılması gereken işlem COCO veri seti üzerinde eğitilmiş olan modellerin tez çalışması için oluşturulan silah veri seti üzerinde yeniden eğitilmesidir. Tez çalışmasında Faster RCNN modeli kullanılarak silahların tespit edilmesi amaçlanmıştır. Bu nedenle ilk olarak kullanılacak olan eğitilmiş modeller tensorflow detection model zoo [85] içerisinde indirilmiştir.

Model name	Speed (ms)	COCO mAP[^1]	Outputs
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_mobilenet_v2_coco	31	22	Boxes
ssdlite_mobilenet_v2_coco	27	22	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_inception_v2_coco	58	28	Boxes
faster_rcnn_resnet50_coco	89	30	Boxes
faster_rcnn_resnet50_lowproposals_coco	64		Boxes
rfcn_resnet101_coco	92	30	Boxes
faster_rcnn_resnet101_coco	106	32	Boxes
faster_rcnn_resnet101_lowproposals_coco	82		Boxes
faster_rcnn_inception_resnet_v2_atrous_coco	620	37	Boxes
faster_rcnn_inception_resnet_v2_atrous_lowproposals_coco	241		Boxes
faster_rcnn_nas	1833	43	Boxes
faster_rcnn_nas_lowproposals_coco	540		Boxes
mask_rcnn_inception_resnet_v2_atrous_coco	771	36	Masks
mask_rcnn_inception_v2_coco	79	25	Masks
mask_rcnn_resnet101_atrous_coco	470	33	Masks
mask_rcnn_resnet50_atrous_coco	343	29	Masks

Şekil 5.10. COCO veri seti ile eğitilmiş modeller ve hız, doğruluk, çıktı bilgileri

Şekil 5.10’da görüldüğü gibi SSD, Faster RCNN, RFCN ve Mask RCNN modellerinin farklı özellik çıkarıcı ağılar kullanılarak eğitilmesinden elde edilen modeller bulunmaktadır. Buradaki hız verisi bir resmin modelde işlenerek çıktının üretilmesine kadar geçen süreyi

göstermektedir. mAP değeri nesne tespitinde doğruluk oranını ve çıktılar ise modelin kullanıcıya sunduğu sonuçları ifade etmektedir. Bu tez çalışması resim bölütleme uygulamalarını içermediğinden Mask RCNN modeli kullanılmamıştır. Hız doğruluk oranı göz önünde bulundurularak özellik çıkarıcı olarak ResNet101 modelini kullanan Faster RCNN modeli tercih edilmiştir. API indirilen bu modelin yeni veri setleri ile tekrar eğitilmesine olanak sağlamaktadır. Bunun için oluşturulan veri setinin model tarafından okunabilir hale getirilmesi gerekmektedir. Veri setinin hazırlanması için daha önce oluşturulmuş olan xml uzantılı dosyalar ve resim verileri birlikte kullanılarak resim ve sınırlayıcı kutular tek bir dosya halinde sıkıştırılmalıdır. Bu işlem sırasında resimlerin piksel değerleri ve nesnelerin resim içindeki yerleri dizi halinde “.record” dosya formatında kaydedilir. “.record” uzantısı ağıın resimleri okuyabileceği dosya formatıdır [95]. Bu dönüşümü gerçekleştirmek için gerekli kod parçası hazırlanarak dönüşüm yapılmıştır. Böylelikle yapay sinir ağı ve veri seti eğitime hazır hale gelmiştir. Ancak eğitime başlamadan önce ayarlanması gereken bazı parametreler bulunmaktadır.

5.5 Eğitimde Kullanılan Hiper Parametreler

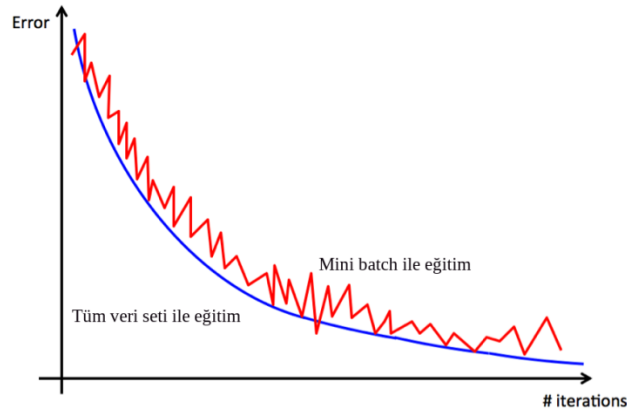
Hiper parametreler yapay sinir ağlarının eğitim sırasında öğrenemediği ancak eğitime başlamadan önce belirlenmesi gereken parametrelerdir. Bazı kümeleme yöntemlerinde küme sayısının belirlenmesi gibi derin öğrenme uygulamalarında da belirlenmesi gereken hiper parametreler bulunmaktadır.

Derin öğrenmede en fazla karşılaşılan hiper parametrelerden birisi veri seti boyutudur. Daha iyi bir genelleme ve dolayısı ile öğrenmenin gerçekleşebilmesi için modelin olabildiğince fazla veri görmesi gerekir. Ancak her zaman bu kadar fazla veri toplamak mümkün olmamaktadır. Bu durumda sentetik veri seti oluşturma yöntemleri kullanılabilir. Örneğin eğitim setindeki bir verinin saat yönünde döndürülmesi, renk düzeninin değiştirilmesi gibi işlemler sonucunda çıkan resimler, yeni eğitim verileri gibi kullanılabilir. Bu tez çalışmasında veri setinde bulunan tüm veriler yatay eksende rastgele yönde döndürülerek veri seti büyüklüğü artırılmıştır. Bu adımda her resmin %50 şans ile aynada yansır hali üretilmiştir. Böylelikle hazırlanan veri setine ek olarak eğitim sırasında yeni resimler eklenmiştir. Şekil 5.11’de yatay eksende çevrilen bir resmin yeni hali gösterilmiştir.



Şekil 5.11. Yatay ekseninde çevrilen örnek resim

Bir diğer önemli parametre bir anda eğitimde kullanılacak veri sayısıdır. Veri setinde bulunan tüm veriler aynı anda kullanılarak eğitimin yapılması genellikle belleğin yetmemesi yüzünden imkânsız olmaktadır. Ayrıca tüm verilerin kullanılması, geri yayılım algoritmasının çalışmasını yavaşlatır ve eğitimin daha uzun sürmesine yol açar bu nedenle eğitim sırasında veri seti belli boyutta parçalara bölünerek yapay sinir ağı eğitilir. Bu parametre mini-batch değeri olarak adlandırılır. Bir sinir ağının bir adımda kaç tane veri üzerinde çalışacağını belirleyen parametredir. Bu değer en az 1 en fazla veri setindeki tüm verilerin sayısı kadar değer alabilir. Bu parametrenin çok büyük ayarlanması gibi çok küçük ayarlanması da öğrenmenin gerçekleşmemesi, veri setinde genelleme ile göz ardı edilmesi gereken gürültülerin de öğrenilmesi gibi problemlere yol açabilir. Bu nedenle dikkatli seçilmesi gereken bir parametredir.

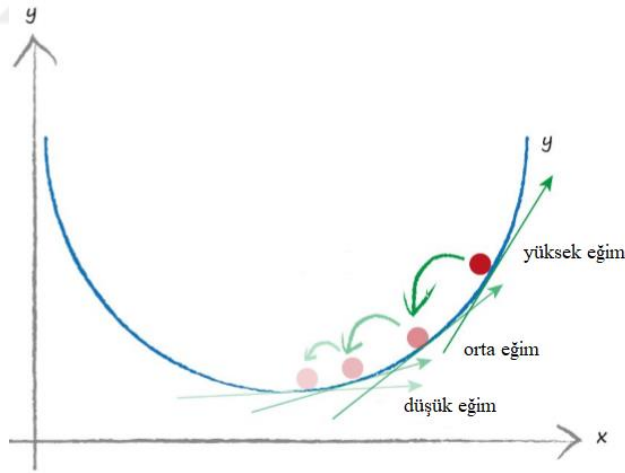


Şekil 5.12. Tüm veri seti ve mini-batch eğitimi hata grafikleri [96].

Şekil 5.12’de görüldüğü gibi mini batch kullanılması durumunda hata grafiğinde dalgalanmalar olmaktadır. Bu durum her iterasyonda başka veriler ile eğitime devam edilmesinden kaynaklanmaktadır. Yani, bir veri grubu için öğrenilen parametreler diğer veri

grubu için tam uygun olmamasından dolayı dalgalanmalar meydana gelmektedir. Ancak grafikte de görüldüğü gibi iki durumda da hata değeri yaklaşık aynı değerlere kadar düşmektedir. Bu parametrenin seçiminde dikkat edilmesi gereken bir nokta; batch değeri eğitimin yapıldığı GPU belleğine sığması gerektiğinden 2 ve katları olarak belirlenmesidir. Tez çalışmasında faster rcnn modeli için batch değeri 1 seçilmiştir.

Öğrenme hızı, yapay sinir ağında öğrenilen parametrelerin güncellenme hızını belirleyen parametredir. Bölüm 3.3.5.3’de gösterildiği gibi geri yayılım algoritmasında güncellenme miktarının bir çarpanıdır. Öğrenme parametrelerinin güncellenmesi için öğrenme hızı hata fonksiyonunun o andaki eğimi ile çarpılarak ağırlık değerinden çıkarılır. Bu parametre eğitim süresince sabit seçilebileceği gibi belli aralıklarla değişecek şekilde de seçilebilir. Örneğin belli bir adım sayısına kadar 10^{-4} seçilirken daha sonra 10^{-3} olarak değişmesi sağlanabilir. Bunun nedeni Şekil 5.13’de gösterildiği gibi hata değerinin düşük değerlerine yaklaşıldıkça eğimden gelen güncelleme oranının azalması ile çok düşük öğrenme hızı katsayısı kullanıldığında öğrenme sürecinin çok yavaşlamasıdır. Ancak minimum noktasının atlandığı durumlarda da tam tersi bir yöntem izlenerek öğrenme hızı katsayısı düşürülmektedir.



Şekil 5.13. Eğim değişimi ve adım sayısı arasındaki bağlantı [43].

Öğrenme hızı parametresi öğrenmenin gerçekleşmesinde önemli rol oynamaktadır. Çok yüksek seçilirse hata grafiğinde minimum noktanın kaçırılmasına ve minimum noktası etrafında salınım yapılmasına yol açabilir. Öğrenme hızının ayarlanmasında en yaygın yöntem daha büyük bir değerden başlayarak eğitimin ilerleyen adımlarında azaltmaktır. Böylelikle eğitimin, başlarında yerel minimuma takılması engellenmekte ve global minimum noktasına yaklaşıldığında daha hassas ilerlemesi sağlanmaktadır. Yapılan

alıřmalar gz nnde bulundurulduėunda ėrenme hızı 0.01 deėerlerinde seilerek eėitimin ilerleyen adımlarında 10 kat dřrlerek eėitime devam edilmesi iyi bir seim olarak grlmektedir [2,57-59]. Ancak, blm 5.1’de anlatılan eėitilmiř modelin tekrar eėitildiėi uygulamalarda, modelin belli bařlı zellikleri ıkarmayı zaten ėrenmiř olduėundan dolayı, ėrenme hızı normalden daha da dřk seilmelidir. řekil 5.14’te ėrenme hızının eėitime etkisi gsterilmiřtir.



řekil 5.14. ėrenme hızı seiminin eėitime etkisi [97].

Bu tez alıřmasında ėrenme hızı daha nceden eėitilmiř modeller zerinde eėitim yapıldıėı iin 3×10^{-4} olarak ayarlanmıřtır.

Tm bu parametrelerin yanında son katmandaki nron sayısı tespit edilecek nesne sayısına eřit olacak řekilde ayarlanır. Bu tez alıřmasında son katmanda sadece bir nesnenin olup olmadıėı sorgulandıėından dolayı 1 nron bulunmaktadır. zellik ıkarıcı konvolsyonel sinir aėına girdi olarak gnderilecek resimlerin boyutlandırılması da diėer bir parametredir. Tez alıřmasında resimler en kk kenar 600 piksel olacak řekilde en boy oranı korunarak yeniden boyutlandırılmıřtır.

Standart sınırlayıcı kutular faster rcnn iin 0.5, 1 ve 2 en boy oranlarına sahip kutuların 0.25, 0.5, 1 ve 2 arpanları ile her alan iin toplam 12’řer adet retilmiřtir.

5.6 akıřmaların Giderilmesi

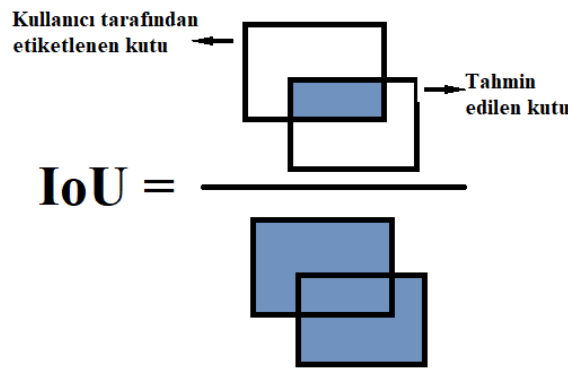
Nesne tespiti algoritmaları nesnelerin bulunma ihtimali olan alanları belli algoritmalar ve standart sınırlayıcı kutular yardımı ile bulmaktadır. Oluřturulan bu alanlara alan teklifleri denmektedir. Ancak her zaman bir nesne zerinde sadece bir kutu oluřmamaktadır. Aynı nesne zerinde birden fazla alan teklifi bulunduėunda, ikisi de nesneyi ierdiėinden dolayı

doğru tahmin olarak değerlendirilebilmektedir. Bu durum nesne tespiti uygulamalarında istenmeyen bir durumdur. Her nesne yalnızca bir sınırlayıcı kutu tarafından çevrelenmelidir. Bu tür çakışmaları önlemek için günümüzde en son geliştirilen nesne tespiti modelleri non-maximum suppression algoritmasını kullanmaktadır [4,55,56].



Şekil 5.15. Non-maximum suppression algoritması öncesi ve sonrası [98].

Şekil 5.15'te non-maximum suppression algoritması ile sadeleştirilen alan tahminleri gösterilmiştir. Algoritma buradaki birçok sınırlayıcı kutu arasından en doğru tahmin edileni seçmek için kullanıcı tarafından etiketlenen kutular ile tahmin edilen kutuları karşılaştırır. Karşılaştırma ölçütü kullanıcı etiketi ile tahmin edilen kutunun kesişimlerinin birleşimlerine oranıdır (Intersection over Union-IoU). Bu oran belli bir değerin üzerinde olduğunda tahmin doğru olarak sınıflandırılır. Birden fazla tahmin edilen kutu bu oranı geçtiğinde içlerinden en büyük değere sahip olan kutu doğru olarak kabul edilir geri kalan tüm kesişen kutular elimine edilir. Bu adımlar tüm kutular doğru ya da yanlış olarak sınıflandırılana kadar devam eder. IoU değeri tez çalışmasında 0.5 olarak ayarlanmıştır.



Şekil 5.16. IoU değerinin hesaplanması

Şekil 5.16.'da IoU değerinin hesaplanması gösterilmiştir. Şekilden de anlaşılacağı gibi IoU değeri en yüksek 1 olurken en düşük 0 olabilir. Bu durumda bir nesne için tahmin edilen sınırlayıcı kutulardan IoU değeri 1'e en yakın olanı doğru tahmin olarak kabul edilmektedir. Geri kalan tahminler de en hatalı olandan başlayarak 5'er tanesi negatif örnek olarak eğitime eklenmektedir. Bu sayede eğitim kümesinde negatif örneklerin oluşturulması gerekmemiştir.

5.7 Eğitimde Hata Fonksiyonunun Hesaplanması

Nesne tespiti uygulamaları resim içinde bulunan bir nesnenin lokasyonunun bulunması ve nesnenin sınıflandırılması gerekmektedir. Bu işlemler eğitilen Faster RCNN modeli için RPN ağında ve sonrasında gerçekleştirilmektedir. Her iki işlemin de doğru bir şekilde yapıldığının anlaşılması amacıyla bir amaç fonksiyonu oluşturulmuştur. Bu amaç fonksiyonu tahmin edilen sınırlayıcı kutunun olması gereken sınırlayıcı kutuya uzaklığı ile tahmin edilen sınırlayıcı kutunun bir nesne veya arka plan içermesinin olasılık skorunun toplamı olarak oluşturulmuştur. Matematiksel olarak hata fonksiyonu 5.1'de verilmiştir.

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (5.1)$$

Eğitim sırasında bu fonksiyonun minimize edilmesi amaçlanmıştır. Burada i , bölüm 4.2.3.3'de bahsedilen anchor kutuların indisini ifade etmektedir. p_i , i . indisdeki sınırlayıcı kutunun bir nesne olma olasılığını ifade etmektedir. p_i^* olması gereken sınırlayıcı kutunun değerini ifade eder ve i . anchor sınırlayıcı kutunun olması gereken sınırlayıcı kutu ile kesişimi 0.7'den büyükse 1, değilse 0 değerini alır. t_i , tahmin edilen sınırlayıcı kutunun koordinatlarını temsil eden bir vektördür ve t_i^* tahmin edilen sınırlayıcı kutunun kesişim oranı 0.7'den olduğu gerçek sınırlayıcı kutunun koordinatlarını temsil eden vektördür. L_{cls} nesne veya arka plan sınıfları arasında logaritmik hata değeridir. Bu hata değeri 0 ve 1 arasında değer almaktadır. $L_{reg}(t_i - t_i^*)$ tahmin edilen ve olması gereken sınırlayıcı kutuların birbirinden farkını göstermektedir. Bu fonksiyonun matematiksel açık hali [48]'de verilmiştir. p_i ile bu hatanın çarpılmasının nedeni sadece doğru tahmin edilen sınırlayıcı kutular için hesaplanmasının sağlanmasıdır. Her iki hata değerinin de çıktıları N_{reg} ve N_{cls}

değerleri ile normalize edilmiştir. Sonuçta elde edilen toplam, modelin hata değerini vermektedir.

5.8 Eğitim Sonucunun Değerlendirilmesi

Nesne tespiti uygulamalarının değerlendirilmesinde birbirinden ayrı iki ölçüt bulunmaktadır. Bunlar; resmin içinde nesnenin bulunup bulunmadığı (sınıflandırma) ve bulunuyor ise resmin hangi alanında olduğu (lokalizasyon) dur. Bunların yanında tipik bir nesne tespiti veri setinde birden fazla nesne bulunmakla birlikte bu nesneler eşit dağılıma sahip değildir. Bu nedenle doğruluk tabanlı basit bir değerlendirme ölçütü sapmalara yol açmaktadır. Doğru tespitlerin yanında yanlış tespitlerin de göz önünde bulundurulması ayrıca önemlidir. Bu nedenle, tespit edilen her bir sınırlayıcı kutu ile bir “güven puanını” ilişkilendirmek ve modeli çeşitli güven düzeylerinde değerlendirmek gerekir. Bu ihtiyaçları karşılamak için, ortalama hassasiyet (Average Precision-AP) algoritması önerilmiştir [99]. Değerlendirme metriklerinin anlaşılması için öncelikler precision ve recall terimleri bilinmelidir. Precision, doğru tespitlerin tüm tespitlere oranı olarak ifade edilir. 1’e yakın bir precision değeri elde edilmesi, tahmin edilen sınırlayıcı kutuların yüksek oranda doğru olduğu anlaşılmaktadır. Recall ise doğru tahminlerin toplam tahmin edilmesi gereken sayıya oranıdır. Yani, 1’e yakın bir recall elde etmek tespit edilmesi gereken nesnelerin yüksek oranda doğru şekilde tespit edildiği anlamına gelir. Precision değeri yanlış tespitlerin artması ile azalırken, recall değeri tespit edilmesi gerekirken tespit edilemeyen nesnelerin sayısı arttıkça azalır. Buradan anlaşılması gereken precision ve recall değerleri arasında bir ters orantı vardır. AP değerinin hesaplanması için modelin tespit çıktılarından yararlanılarak precision-recall grafiği çizilir. Daha sonra grafikte [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1] recall değerlerine karşılık gelen maksimum precision değerlerinin ortalaması alınır ve AP hesaplanmış olur [100]. Bu işlemlerin tamamında bölüm 5.6’da anlatılan IoU değeri ve sınıflandırma sonucu göz önünde bulundurularak tespitin doğru veya yanlış olduğu kabul edilmektedir.

Nesne tespiti algoritmalarında standart değerlendirme ölçütü olarak mean average precision (mAP) değeri kabul edilmiştir [100]. mAP, AP değerinin veri setinde bulunan tüm sınıflar için hesaplanarak ortalamasının alınması ile hesaplanmaktadır. Elde edilen değer nesne tespiti uygulamalarının başarımının ölçülmesinde standart olarak kabul edilmiştir.

Bu tez çalışmasında gerçekleştirilen deneyler 32 GB bellek, Nvidia Quadro M4000 8 GB grafik işlemciye sahip HP Workstation Z840 makinesinde Ubuntu 14.04 işletim sistemi üzerinde gerçekleştirilmiştir. İlk olarak bölüm 5.1’de anlatıldığı gibi eğitilmiş modellerden yararlanılarak elde edilen resim dosyalarının sınıflandırılarak kullanıcıya sunulması amaçlanmıştır. Bu uygulama için farklı hız ve doğruluk oranlarına sahip Faster RCNN, RFCN ve SSD modelleri seçilmiş ve sonuçlar incelenmiştir. Modellerde özellik çıkarıcı olarak hız ve doğruluk arasında denge sağlanması için ResNet101 ve MobileNet Inception v2 modelleri seçilmiştir. Sınıflandırmada kullanılan tüm modeller COCO veri seti ile eğitilmiştir. COCO veri seti 80 farklı sınıfta toplam 200.000’den fazla resimden oluşan bir veri setidir. Veri setinde bulunan sınıflar küçük resimler halinde Şekil 6.1’de gösterilmiştir. Yapılan ilk deneyde, COCO veri setinde bulunan nesnelerin yer aldığı her resim sınıflandırılarak ayrı klasörlerde adli incelemeciye sunulmuştur. Alınan sonuçlar hız ve doğruluk bakımından incelenmiştir. Adli delilden elde edildiği varsayılan test seti Pascal VOC veri setinden rastgele seçilen 8000 resimlerden oluşturulmuştur.

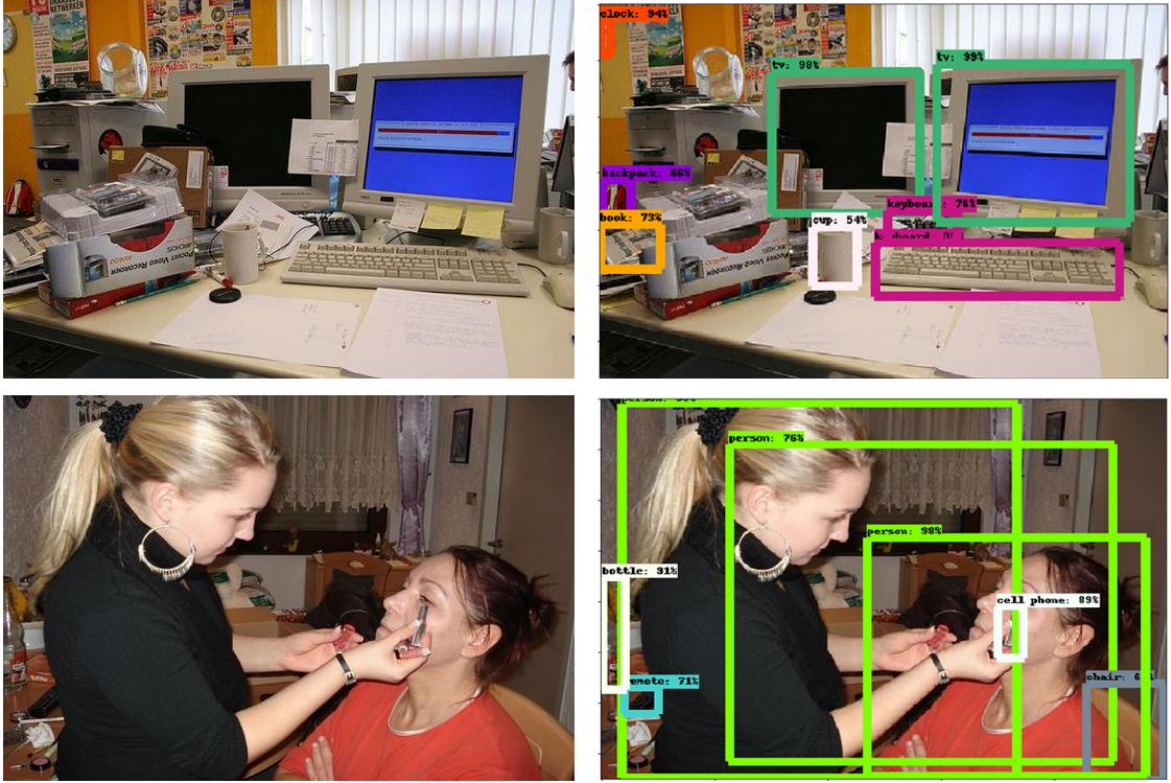


Resimler seçilen modellere tek tek gönderilerek içinde bulunan nesnelere göre klasörlere kaydedilmiştir. Hiçbir nesne bulunamayan resimler de ayrı bir klasöre kaydedilmiştir. Elde edilen veriler Tablo 6.1’de gösterilmiştir.

Tablo 6.1. Modellerin doğruluk oranları ve sınıflandırma hızı

Model	Özellik Çıkarıcı	Doğruluk (mAP)	Zaman (dk)	Sınıflandırılmayan Resim Sayısı
Faster RCNN	ResNet101	32	93.26	39
RFCN	ResNet101	30	69,71	90
SSD	MobileNet Inception v2	24	43,16	419

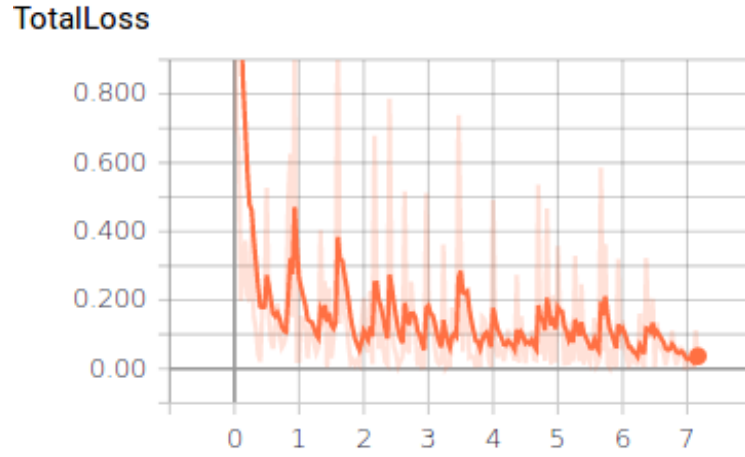
Tablo 6.1’de görüldüğü gibi 8000 görüntü dosyası üzerinde yapılan filtreleme sonucunda geçen zaman ve sınıflandırılmayan nesnelerin sayısı eğitim sırasında alınan doğruluk oranına göre farklılık göstermiştir. En yüksek mAP değerine sahip olan Faster R-CNN en yüksek sınıflandırma performansına sahip olmuştur. RFCN, Faster R-CNN’e yakın bir performans gösterirken, küçük nesnelerin algılanmasında zorlanan SSD en hızlı model olmasına rağmen çok daha fazla sınıflandırılmayan resim üretmiştir. Bu noktada kullanıcıların oluşturdukları sisteme uygun modeli seçmeleri gerekmektedir. Adli bilişim açısından düşünüldüğünde doğruluğu yüksek modellerin kullanılması daha sağlıklı sonuçların alınması açısından önemlidir. Ayrıca, derin öğrenme kullanılmasının incelemeyi hızlandığı açık bir şekilde görülmektedir. Deneyde kullanılan resimlerin her birine manuel olarak 10 sn bakılarak değerlendirilse dahi inceleme süreci 1333,33 dk sürecektir. Buradan görülmektedir ki kullanılan en yavaş model adli delil araçlarından elde edilen resimleri insana göre %93 daha hızlı sınıflandırmaktadır. Günümüzde adli delil incelemelerini bu seviyede hızlandıran bir araç bulunmamaktadır. Bu durumda gözden kaçırılan delillerin olması da olası bir durumdur.



Şekil 6.2. Faster R-CNN tarafından çıkarılan resimler

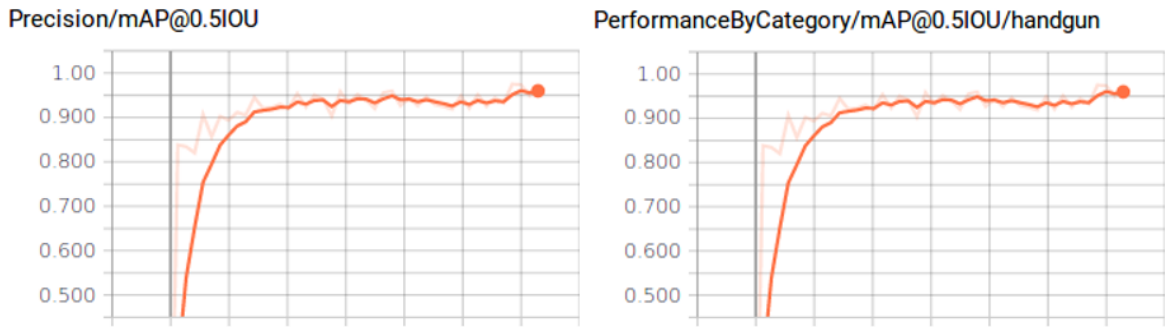
Şekil 6.2’de görüldüğü üzere ilk bakışta algılanması zor ve gözden kaçma olasılığı yüksek nesneler, yapay zekâ tarafından saptanarak kullanıcıya sunulmuştur. Bu, günümüzde geliştirilmeye ihtiyacı olan dijital delil inceleme sürecinin yapay sinir ağları ve derin öğrenme ile kolay bir şekilde iyileştirilebileceğini göstermektedir.

İlk deneyden alınan sonuçlar adli bilişim incelemelerinde derin öğrenmenin kullanılabileceğini göstermiştir. COCO veri seti nesnelerinin sınıflandırılmasından sonra adli bilişim açısından önemli nesnelerin de birçok resim dosyası arasında saptanması amaçlanmıştır. Bu amaçla, ilk uygulamada kullanılan en yüksek başarımlı orana sahip Faster RCNN modeli ile silah nesnelerinin tespit edilmesinde kullanılmıştır. Bunun için model bölüm 5.2’de anlatılan veri seti ve bölüm 5.5’te belirtilen hiper parametreler kullanılarak eğitilmiştir. Eğitim süreci tensorflow’un kendi içerisinde barındırdığı görselleştirme aracı tensorboard aracı ile takip edilmiştir. Daha önce COCO veri seti ile eğitilen model transfer learning kullanılarak sadece silah nesnelerini tespit edecek şekilde, toplamda 7 saat ve 31445 adım boyunca eğitilmiştir. Eğitim sırasında elde edilen hata grafiği Şekil 6.3’te gösterilmiştir.



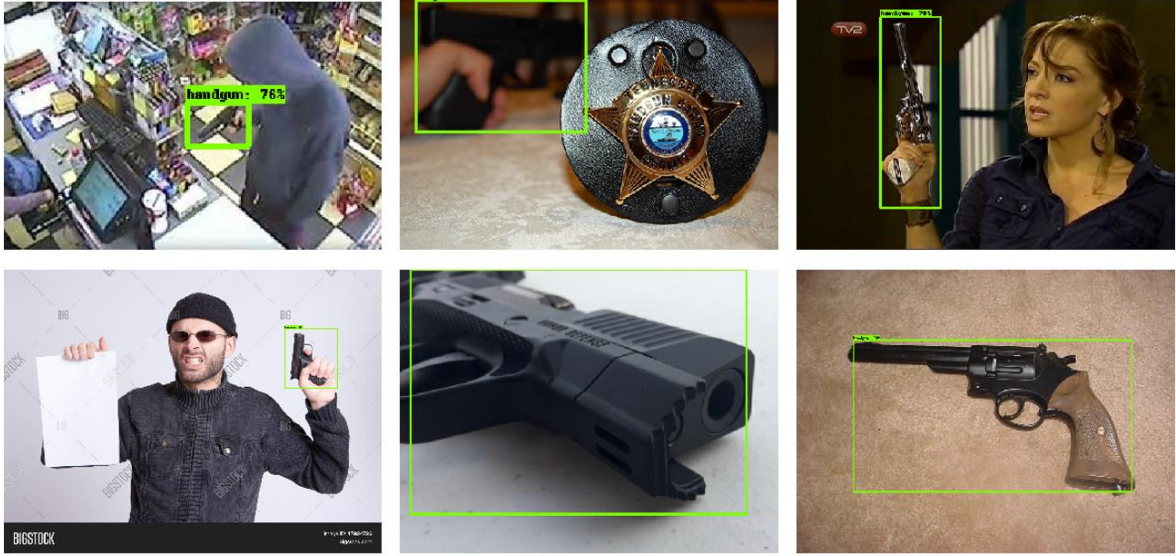
Şekil 6.3. Eğitimde elde edilen hata grafiği (smoothing: 0.7)

Şekilde görülen hata grafiği eğitim kümesi için üretilen hem sınıflandırma hem de lokalizasyon hatalarının toplamını yansıtmaktadır. Toplam hata eğitim sonunda sıfıra yakın (0.016) bir değer almıştır. Bu veriler eğitim kümesinde elde edildiğinden dolayı parametrelerin güncellenmesinde etkisi olmayan resimler kullanılarak da modelin test edilmesi gerekmektedir. Çünkü modelin eğitim kümesini ezberlemesi durumunda hata grafiğindeki değer düşerken, modelin daha önce görmediği resimler üzerindeki başarısı da düşebilir. Böyle bir durum olup olmadığını test etmek için bölüm 5.8’de anlatıldığı gibi daha önce oluşturulan test veri seti üzerinde başarımlar ölçülmüştür. Sonuçlar eğitimde olduğu gibi tensorboard üzerinden takip edilmiştir.



Şekil 6.4. Test seti üzerinde elde edilen başarımlar (smoothing: 0.7)

Şekil 6.4’te test veri setindeki silah nesnelerinin tespit edilmesinde, eğitilen modelin başarısı gösterilmiştir. Modelin son halinde mAP değeri 96.61 olarak ölçülmüştür. Eğitim sonunda test veri setinde bulunan bazı resimlerin çıktısı Şekil 6.5’te gösterilmiştir.



Şekil 6.5. Eğitilen model tarafından işlenen test resimleri

Buradan eğitim süresince aşırı eğitim durumunun olmadığı anlaşılmaktadır. Ancak eğitimin ortasından itibaren başarımda çok artış görülmemektedir. Model küçük artışlar ile eğitime devam etmiştir.

Modelin doğru olarak tespit ettiği resimler gibi tespit edemediği resimler de bulunmaktadır ancak bu resimler oldukça düşük orandadır (%5).



Şekil 6.6. Eğitilen modelin hata yaptığı resimler

Şekil 6.6’da görüldüğü gibi silahın farklı perspektiflerde veya kısmen görülebildiği bazı durumlarda model silahı tespit edememiştir. Böyle yanlışlıklar resmin çözünürlüğü, arka

plan renginden silah nesnesinin zor ayırt edilmesi ve veri setinde yeterince farklı perspektiften resmin olmayışından kaynaklanmaktadır. Bu problem veri setinin genişletilmesi ile ortadan kaldırılabilir. Bazı durumlarda ise fazladan nesneler silah etiketi ile tespit edilmiştir. Bu resimler genellikle eğitim setinde çok az bulunan ancak COCO veri setinde bulunan nesneleri içeren resimlerdir. Modelin eski nesneleri tespit etmeyi iyi öğrenmesinden dolayı bu nesneler tekrar tespit edilmektedir. Bu problemin çözülmesi için eski veri setinde bulunan nesneleri de kapsayan yeni bir veri seti oluşturularak eğitim gerçekleştirilmelidir. Ancak tezde kullanılan bilgisayar ve donanımın COCO veri seti ile birlikte yeni veri setinin eğitilmesi için gerekli güce olmadığından tez çalışmasında COCO veri setinin tamamı kullanılarak model eğitimi gerçekleştirilememiştir. Test veri setinin sınıflandırılması sonucunda 1332 adet resimden 32 tanesi yanlış sınıflandırılmış, 40 tanesi hiç sınıflandırılmamış ve 1260 tanesi doğru olarak sınıflandırılmıştır. Buradan eğitilen modelin yaklaşık olarak %95 oranında bir başarıma sahip olduğu görülmektedir.

7. YORUM VE SONUÇ

Bu tez çalışmasında adli delil inceleme sürecinin hızlandırılması amacıyla derin öğrenme ile nesne tespiti algoritmalarından yararlanılmıştır. İlk olarak yöntemin kullanılabilir olduğunu göstermek amacıyla daha önceden büyük veri setleri ile eğitilmiş olan Faster RCNN, RFCN ve SSD modelleri kullanarak adli delillerden elde edildiği düşünülen resimler içerisinde tespit edilen nesnelere göre sınıflandırılarak farklı klasörlere kaydedilmiştir. Bu deney sonucunda 8000 test resminin sınıflandırılarak 90 farklı klasöre kaydedildiği görülmüştür. Kullanılan en güçlü model olan Faster RCNN için sınıflandırılmayan 39 resmin yanında resimlerin yaklaşık %12' si yanlış sınıflarda bulunmuştur. Ancak bu durum çok fazla sınıfın olduğu COCO veri seti için normaldir. Çünkü bu veri setinde tüm nesnelerden aynı miktarda bulunmamaktadır. Bu durumda özellikle küçük nesnelerin tespitinde yapay sinir ağı daha çok zorlanmaktadır. Yanlış olarak sınıflandırılan nesnelerin çoğu elma, beyzbol eldiveni, cep telefonu gibi küçük nesnelerdir. Bu tip hataların önlenmesi için veri setinde bulunan bu nesnelerin sayısının artırılması gerekmektedir. Yanlışların yanında modeller çok sayıda veriyi doğru sınıflandırmıştır. Deney sonucunda, bölüm 6'da gösterildiği gibi sadece sınıflandırılmış klasörler incelenerek delillerin aranması adli delil inceleme sürecini büyük ölçüde kısaltabilmektedir.

Derin öğrenme ile resim verilerinin sınıflandırılmasının adli delil inceleme sürecini geliştireceğinin gözlenmesinin ardından adli bilişim açısından önemli nesnelerin tespit edilmesi amaçlanmıştır. Örnek nesne olarak birçok suça ilişkin aranan genel bir nesne olan silah nesnesi seçilmiştir. Elde edilen resimler içerisinde silah nesnesinin tespit edilebilmesi için Faster RCNN modeli hazırlanan veri seti ile eğitilmiştir. Eğitim sonuçları eğitilen modelin silah nesnesini yüksek başarımla tespit edebildiğini göstermektedir. Ancak bazı açılardan silahların tespit edilmesinde zorlandığı görülmüştür. Bu hata veri setinin genişletilerek eğitim yapılması ile çözülebilir. Modelin en zorlandığı nokta yanlış nesnelerin silah olarak tespit edilmesidir. Modelin daha önce eğitilmiş olduğu veri setindeki nesneler yeni oluşturulan veri setinde negatif örnekler olarak eklenmediğinden kaynaklanmaktadır. Bu hata için de veri setinin negatif örnekler ile genişletilmesi gerekmektedir. Bunun yanında bu hata adli delil incelemesinin doğruluğuna çok büyük zarar vermezken zamandan yapılan tasarrufu düşürmektedir. Ancak, bu hatalar modelin başarılı sonuçlarının arkasında

kalmaktadır. Önerilen yöntem ile adli delil inceleme sürecinin önemli derecede geliştirildiği gösterilmiştir.

Bu tez çalışmasında nesnelerin tespit edilmesinde kullanılan model hız-doğruluk arasında denge kuran bir model olarak seçilmiştir. Ancak doğruluğun daha yüksek olması istendiğinde daha karmaşık bir model kullanılmalıdır. Tam tersi hızın önemli olduğu gerçek zamanlı alarm sistemleri gibi uygulamalarda ise hata toleransını kabul ederek daha basit bir model kullanılmalıdır. Tez çalışmasında oluşturulan veri seti çok yüksek ve çok düşük çözünürlüklü resimleri de barındırmaktadır. Başarımın daha da artırılması için düşük çözünürlüklü resimlerin temizlenmesi yararlı olacaktır. Ayrıca elde edilen hız verileri incelemenin yapıldığı bilgisayarın donanım özelliklerine göre değişiklik göstermektedir. Daha üst düzey GPU'lar kullanılarak çok daha hızlı sonuçlar alınması mümkündür. İlerleyen çalışmalar ve adli delil inceleme yazılımlarında yapay zekâ algoritmalarının kullanımının yaygınlaşması sonucu adli bilişim incelemelerinde yüksek işlem gücüne sahip gpu kartları da bir gereklilik haline gelecektir.

İlerleyen çalışmalarda önerilen yöntem geliştirilerek genel anlamda adli bilişim incelemelerinde aranan nesneler için genişletilmesi amaçlanmaktadır. Ayrıca görüntü dosyalarının yanında, ses, text ve video verileri için de farklı modeller geliştirilmesi düşünülmektedir. Geliştirilen bu sınıflandırma araçlarının bir adli bilişim aracına entegre edilmesi hatta daha sonraki aşamalarda verilerin kendi yazdığımız programlar ile çıkarılarak yeni açık kaynaklı yapay zeka destekli adli inceleme yazılımlarının üretilmesi amaçlanmaktadır.

KAYNAKLAR

- [1] **Garfinkel, S. L.** 2010. Digital forensics research: The next 10 years. *digital investigation*, 7, S64-S73.
- [2] **He, K., Zhang, X., Ren, S., & Sun, J.** 2016. Deep residual learning for image recognition. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- [3] **Long, J., Shelhamer, E., & Darrell, T.** 2015. Fully convolutional networks for semantic segmentation. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- [4] **Ren, S., He, K., Girshick, R., & Sun, J.** 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Paper presented at the Advances in neural information processing systems.
- [5] **Kanunu, T. C.** Kanun Numarası 5237.
- [6] <https://www.us-cert.gov/sites/default/files/publications/forensics.pdf>, US government organization, Computer Forensics, 14 Ağustos 2017.
- [7] **Wood, C. C., Banks, W. W., Guarro, S. B., Garcia, A. A., Hampel, V. E., & Sartorio, H. P.** 1987. Computer security: a comprehensive controls checklist: Wiley-Interscience.
- [8] **Brinson, A., Robinson, A., & Rogers, M.** 2006. A cyber forensics ontology: Creating a new approach to studying cyber forensics. *digital investigation*, 3, 37-43.
- [9] **Peisert, S., Bishop, M., & Marzullo, K.** 2008. Computer forensics in forensics. Paper presented at the Systematic Approaches to Digital Forensic Engineering, 2008. SADFE'08. Third International Workshop on.
- [10] **Meghanathan, N., Allam, S. R., & Moore, L. A.** 2010. Tools and techniques for network forensics. arXiv preprint arXiv:1004.0570.
- [11] **Social, W. a.** 2017. Digital in 2017: Global Overview.
- [12] **Kapil, S., Kumar, A., Bansal, R., & Victor, N.** SOCIAL NETWORK FORENSICS.
- [13] **Casey, E., Blitz, A., & Steuart, C.** 2004. Digital evidence and computer crime. Academic press, 94.
- [14] **Reith, M., Carr, C., & Gunsch, G.** 2002. An examination of digital forensic models. *International Journal of Digital Evidence*, 1(3), 1-12.
- [15] **Ciardhuáin, S. Ó.** 2004. An extended model of cybercrime investigations. *International Journal of Digital Evidence*, 3(1), 1-22.
- [16] **Baryamureeba, V., & Tushabe, F.** 2004. The enhanced digital investigation process model. Paper presented at the Proceedings of the Fourth Digital Forensic Research Workshop.
- [17] **Henkoğlu, T.** 2011. Adli bilişim: Dijital delillerin elde edilmesi ve analizi: Pusula.
- [18] **Berber, L. K.** 2004. Adli bilişim (computer forensic): İstanbul Bilgi Üniversitesi.
- [19] **Klieiman, D., Timothy, K., & Cross, M.** 2007. The Official CHFI Study Guide for Forensic Investigators.
- [20] **Ballou, S.** 2010. Electronic crime scene investigation: A guide for first responders: Diane Publishing.
- [21] **Hitchcock, B., Le-Khac, N.-A., & Scanlon, M.** 2016. Tiered forensic methodology model for Digital Field Triage by non-digital evidence specialists. *digital investigation*, 16, S75-S85.
- [22] **Raghavan, S.** 2013. Digital forensic research: current state of the art. *CSI Transactions on ICT*, 1(1), 91-114.

- [23] **Wazid, M., Katal, A., Goudar, R., & Rao, S.** 2013. Hacktivism trends, digital forensic tools and challenges: A survey. Paper presented at the Information & Communication Technologies (ICT), 2013 IEEE Conference on.
- [24] **Karie, N. M., & Venter, H. S.** 2015. Taxonomy of challenges for digital forensics. *Journal of forensic sciences*, 60(4), 885-893.
- [25] **Conti, M., Dehghantanha, A., Franke, K., & Watson, S.** 2018. Internet of Things security and forensics: Challenges and opportunities: Elsevier.
- [26] **Franke, K., Årnes, A., Flaglien, A., Sunde, I. M., Dilijonaite, A., Hamm, J., . . . Axelsson, S.** 2018. Challenges in Digital Forensics. *Digital Forensics*, 313-317.
- [27] **Roussev, V., Quates, C., & Martell, R.** 2013. Real-time digital forensics and triage. *digital investigation*, 10(2), 158-167.
- [28] **Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y.** 2016. Deep learning (Vol. 1): MIT press Cambridge.
- [29] **Christopher, M. B.** 2016. PATTERN RECOGNITION AND MACHINE LEARNING: Springer-Verlag New York.
- [30] **Bengio, Y.** 2012. Deep learning of representations for unsupervised and transfer learning. Paper presented at the Proceedings of ICML Workshop on Unsupervised and Transfer Learning.
- [31] **MacQueen, J.** 1967. Some methods for classification and analysis of multivariate observations. Paper presented at the Proceedings of the fifth Berkeley symposium on mathematical statistics and probability.
- [32] **Cover, T., & Hart, P.** 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1), 21-27.
- [33] **Dasarathy, B. V.** 1991. Nearest neighbor ({NN}) norms:{NN} pattern classification techniques.
- [34] **Cortes, C., & Vapnik, V.** 1995. Support-vector networks. *Machine learning*, 20(3), 273-297.
- [35] **Vapnik, V.** 1964. A note one class of perceptrons. *Automation and remote control*.
- [36] **Chand, N., Mishra, P., Krishna, C. R., Pilli, E. S., & Govil, M. C.** 2016. A comparative analysis of SVM and its stacking with other classification algorithm for intrusion detection. Paper presented at the Advances in Computing, Communication, & Automation (ICACCA)(Spring), International Conference on.
- [37] **Quinlan, J. R.** 1986. Induction of decision trees. *Machine learning*, 1(1), 81-106.
- [38] **Quinlan, J. R.** 1990. Decision trees and decision-making. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2), 339-346.
- [39] **Rojas, R.** 2013. Neural networks: a systematic introduction: Springer Science & Business Media.
- [40] **Long, L. N., & Gupta, A.** 2008. Scalable massively parallel artificial neural networks. *Journal of Aerospace Computing, Information, and Communication*, 5(1), 3-15.
- [41] **Hornik, K.** 1991. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2), 251-257.
- [42] **Rumelhart, D. E., Hinton, G. E., & Williams, R. J.** 1986. Learning representations by back-propagating errors. *nature*, 323(6088), 533.
- [43] **Rashid, T.** 2016. Make your own neural network: CreateSpace Independent Publishing Platform.
- [44] **Wilson, D. R., & Martinez, T. R.** 2003. The general inefficiency of batch training for gradient descent learning. *Neural networks*, 16(10), 1429-1451.
- [45] **Ramachandran, P., Zoph, B., & Le, Q. V.** 2017. Swish: a Self-Gated Activation Function. arXiv preprint arXiv:1710.05941.

- [46] **Haykin, S., & Network, N.** 2004. A comprehensive foundation. *Neural networks*, 2(2004), 41.
- [47] **Steinkraus, D., Buck, I., & Simard, P.** 2005. Using GPUs for machine learning algorithms. Paper presented at the Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on.
- [48] **Girshick, R.** 2015. Fast r-cnn. *arXiv preprint arXiv:1504.08083*.
- [49] **LeCun, Y., Bengio, Y., & Hinton, G.** 2015. Deep learning. *nature*, 521(7553), 436.
- [50] **Toderici, G., Vincent, D., Johnston, N., Hwang, S. J., Minnen, D., Shor, J., & Covell, M.** 2016. Full resolution image compression with recurrent neural networks. *arXiv preprint*.
- [51] **Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., . . . Bernstein, M.** 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3), 211-252.
- [52] **Szeliski, R.** 2010. *Computer vision: algorithms and applications*: Springer Science & Business Media.
- [53] **Cipolla, R., Battiato, S., & Farinella, G. M.** 2013. *Machine Learning for Computer Vision (Vol. 5)*: Springer.
- [54] **Huang, T.** 1996. *Computer vision: Evolution and promise*.
- [55] **Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C.** 2016. Ssd: Single shot multibox detector. Paper presented at the European conference on computer vision.
- [56] **Redmon, J., Divvala, S., Girshick, R., & Farhadi, A.** 2016. You only look once: Unified, real-time object detection. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- [57] **Krizhevsky, A., Sutskever, I., & Hinton, G. E.** 2012. Imagenet classification with deep convolutional neural networks. Paper presented at the Advances in neural information processing systems.
- [58] **Simonyan, K., & Zisserman, A.** 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- [59] **Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A.** 2015. Going deeper with convolutions.
- [60] **Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A.** 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. Paper presented at the AAAI.
- [61] **Jung, H., Choi, M.-K., Jung, J., Lee, J.-H., Kwon, S., & Jung, W. Y.** 2017. ResNet-Based Vehicle Classification and Localization in Traffic Surveillance Systems. Paper presented at the Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on.
- [62] **Girshick, R., Donahue, J., Darrell, T., & Malik, J.** 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- [63] **Lowe, D. G.** 1999. Object recognition from local scale-invariant features. Paper presented at the Computer vision, 1999. The proceedings of the seventh IEEE international conference on.
- [64] **Dalal, N., & Triggs, B.** 2005. Histograms of oriented gradients for human detection. Paper presented at the Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on.

- [65] **Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W.** 2013. Selective search for object recognition. *International Journal of Computer Vision*, 104(2), 154-171.
- [66] **LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D.** 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 541-551.
- [67] **LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P.** 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.
- [68] **Fukushima, K.** 1988. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks*, 1(2), 119-130.
- [69] **Marr, D., & Hildreth, E.** 1980. Theory of edge detection. *Proc. R. Soc. Lond. B*, 207(1167), 187-217.
- [70] **LeCun, Y.** 1998. The MNIST database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>, 20 Mayıs 2018.
- [71] **Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R.** 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.
- [72] **Zeiler, M. D., & Fergus, R.** 2014. Visualizing and understanding convolutional networks. Paper presented at the European conference on computer vision.
- [73] **Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S.** 2014. CNN features off-the-shelf: an astounding baseline for recognition. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition workshops.
- [74] **Athiwaratkun, B., & Kang, K.** 2015. Feature representation in convolutional neural networks. *arXiv preprint arXiv:1507.02313*.
- [75] **Oquab, M., Bottou, L., Laptev, I., & Sivic, J.** 2014. Learning and transferring mid-level image representations using convolutional neural networks. Paper presented at the Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.
- [76] **Mahendran, A., & Vedaldi, A.** 2015. Understanding deep image representations by inverting them.
- [77] **Wang, L., Guo, S., Huang, W., & Qiao, Y.** 2015. Places205-vggnet models for scene recognition. *arXiv preprint arXiv:1508.01667*.
- [78] **Xie, S., & Tu, Z.** 2015. Holistically-nested edge detection. Paper presented at the Proceedings of the IEEE international conference on computer vision.
- [79] **Maninis, K.-K., Pont-Tuset, J., Arbeláez, P., & Van Gool, L.** 2016. Convolutional oriented boundaries. Paper presented at the European Conference on Computer Vision.
- [80] **Karpathy, A.** 2014. What I learned from competing against a convnet on ImageNet.
- [81] **Wu, Z., Shen, C., & Hengel, A. v. d.** 2016. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv preprint arXiv:1611.10080*.
- [82] **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.
- [83] **Endres, I., & Hoiem, D.** 2010. Category independent object proposals. Paper presented at the European Conference on Computer Vision.
- [84] **Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., . . . Isard, M.** 2016. TensorFlow: A System for Large-Scale Machine Learning. Paper presented at the OSDI.

- [85] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md, Tensorflow Object Detection Model Zoo, 12 Haziran 2018.
- [86] **Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., . . . Zitnick, C. L.** 2014. Microsoft coco: Common objects in context. Paper presented at the European conference on computer vision.
- [87] **Geiger, A., Lenz, P., Stiller, C., & Urtasun, R.** 2013. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11), 1231-1237.
- [88] **Google**, 2018, Google Open Images Dataset, <https://github.com/openimages/dataset>, 20 Haziran 2018.
- [89] **Yosinski, J., Clune, J., Bengio, Y., & Lipson, H.** 2014. How transferable are features in deep neural networks? Paper presented at the Advances in neural information processing systems.
- [90] **Fei-Fei Li, A. K., Justin Johnson.** Stanford University CS231n Lecture-7 Notes. http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture7.pdf, 21 Haziran 2018.
- [91] **Olmos, R., Tabik, S., & Herrera, F.** 2018. Automatic handgun detection alarm in videos using deep learning. *Neurocomputing*, 275, 66-72.
- [92] <https://github.com/tzutalin/labelImg>, LabelImg Annotation Tool. 21 Haziran 2018.
- [93] **Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., & Shelhamer, E.** 2014. cudnn: Efficient primitives for deep learning. arXiv preprint arXiv:1410.0759.
- [94] https://github.com/tensorflow/models/tree/master/research/object_detection, Tensorflow Object Detection API. 22 Haziran 2018.
- [95] **Tran, D.** 2017. How to train your own Object Detector with TensorFlow's Object Detector API. <https://medium.com/towards-data-science/how-to-train-your-own-object-detector-with-TensorFlows-object-detector-api-bec72ecfe1d9>. 25 Haziran 2018.
- [96] **Cavaioni, M.** DeepLearning series: Deep Neural Networks tuning and optimization. <https://medium.com/machine-learning-bites/deeplearning-series-deep-neural-networks-tuning-and-optimization-39250ff7786d>. 25 Haziran 2018
- [97] <http://cs231n.github.io/neural-networks-3/>, Stanfor University CS231n Convolutional Neural Networks for Visual Recognition. 1 Temmuz 2018.
- [98] **Prisacariu, V., & Reid, I.** 2009. fastHOG-a real-time GPU implementation of HOG. *Department of Engineering Science*, 2310(9).
- [99] **Zhu, M.** 2004. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2, 30.
- [100] **Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A.** 2010. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2), 303-338.
- [101] <http://cocodataset.org/#explore>, Coco Dataset Categories. 29 Haziran 2018.

ÖZGEÇMİŞ

Mustafa ERİŞ

Teknoloji Fakültesi

Adli Bilişim Mühendisliği Bölümü

GSM: +90 531 669 30 70

E- mail: meris@firat.edu.tr

Eğitim

YÜKSEK LİSANS (2015–2018) Fırat Üniversitesi /Elazığ, Türkiye
*Mühendislik Fakültesi Bilgisayar Mühendisliği
Bölümü, Yazılım ABD*

LİSANS (2010–2014) Erciyes Üniversitesi /Kayseri, Türkiye
*Mühendislik Fakültesi Bilgisayar Mühendisliği
Bölümü*